Think Big, Teach Small: Do Language Models Distil Occam's Razor?

Anonymous Author(s)
Affiliation
Address
email

Abstract

Large language models have recently shown a remarkable ability for few-shot learning, including patterns of algorithmic nature. It is now time to ask what kind of patterns these models can capture and how many examples they need in their prompts. We frame this question as a teaching problem with strong priors, and study whether language models can identify simple algorithmic concepts from small witness sets. In particular, we explore how several GPT architectures, program induction systems and humans perform in terms of the complexity of the concept and the number of additional examples, and how much their behaviour differs. This first joint analysis of language models and machine teaching can address key questions for artificial intelligence and machine learning, such as whether some strong priors, and Occam's razor in particular, can be *distilled* from data, making learning from a few examples possible.

1 Introduction

An effective and reliable interaction between humans and machines depends on a proper understanding of their priors. The alignment between their priors is especially critical for communication and learning (Sun et al., 2020). For instance, after a human utters the *prompt* "go stop go", another human would expect "stop" to be a more likely continuation than "go". On the contrary, after the prompt "ready steady" we would expect "go" to be more likely. The former depends on an abstract pattern *intrinsic* to the sequence (a simple algorithmic sequence alternating 'go' and 'stop'), while the latter depends on some language pattern *extrinsic* to the sequence (common use in the English language).

Language models are the most representative AI systems extrapolating sequences of symbols, and are very good at exploiting the extrinsic patterns that are borrowed from humans after training over massive natural language corpora. However, it is only recently, with systems such as BERT (Devlin et al., 2018), GPT-2 (Radford et al., 2019), and GPT-3 (Brown et al., 2020), that we see how these models can also discover some intrinsic patterns in the prompts provided. This kind of inference is called 'few-shot learning', even if there is no further training or fine-tuning of the model. Many of these language models are based on transformers (attention-based architectures, Vaswani et al. (2017)), which do not feature recursion or loops, unlike some approaches based on recurrent neural networks. Some particularly well-chosen and optimised prompts have shown remarkable extrapolations (Xu et al., 2020; Izacard and Grave, 2020; Hendrycks et al., 2021b), illustrating this new kind of *abstraction on the go*. However, it is unclear how elaborate these abstractions can be and whether they can even have an algorithmic nature. A more systematic analysis is needed to determine the optimal prompts that make the model capture a range of different patterns.

Precisely, machine teaching is the area of artificial intelligence that focuses on determining the optimal examples or curriculum that a teacher should use to make a learner capture a concept (Zhu, 2015). This is of enormous importance today in applications such as digital assistants, where we

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

would like to teach them new concepts or routines with a few examples, in the same way humans teach or communicate with other humans (Cakmak and Thomaz, 2014; Goodman and Frank, 2016; Degen et al., 2020). It is also fundamental for explainable AI, as the dual problem appears: how a machine can teach (explain) a concept to a human in an efficient way (Khan et al., 2011; Basu and Christensen, 2013). While machine teaching has a long story of theoretical results about what can be taught with a finite number of examples, it is only recently that the problem has been set in terms of the interactive alignment between the priors of learner and teacher (Goodman and Frank, 2016; Yang and Shafto, 2017; Melo et al., 2018). When dealing with rich representation languages (e.g., regular languages or Turing-complete languages), closer or similar to human language in expressive power, it has recently been shown that strong simplicity priors can make effective teaching possible with very few examples: the overall *size* of the example set is usually shorter than the algorithmic expression of the concept itself (Telle et al., 2019). For instance, teaching the concept of what an odd number is becomes *shorter* by using a few examples than by giving the full definition of what an odd number is.

Language models and machine teaching can now meet because of this new few-shot learning ability of language models. We can explore the question of how many examples are needed (or how long they have to be) to *teach* something to a language model. These are seen as *teaching prompts*: we want the language model to continue the sequence as if it had captured the pattern that has been expressed in the prompt. From the machine teaching perspective, language models are general systems, as they have not been trained for simplicity or for any particular domain, other than a partial sample of human knowledge expressed as natural language.

With their conjunction, we can now ask key questions for AI such as the relevance of simplicity or acquired priors to make learning from a few examples possible (Chater and Vitányi, 2003). In this paper, we examine how a machine teaching setting inspired by Occam's razor can help us understand how language models 'learn' and choose prompts more efficiently. We use the term prior 'distillation', as opposed to the concept known as *knowledge distillation* within the field of machine learning, as we analyse whether a simplicity prior is acquired *as the result* of training large language models such as GPT-2 or GPT-3. To complete the picture, we also evaluate some other AI systems (e.g., program induction systems) and humans on the same task, which are known to have this prior.

The key contributions of this paper are:

- To our knowledge, this is the first paper analysing language models in the context of machine teaching.
- This is also the first paper comparing the acquired priors of language models with humans and program induction systems, when facing patterns that can be taught efficiently with a simplicity prior.
- We find that language models such as GPT-3 have similar or better performance than humans as the number of parameters increase. The performance negatively correlates with the complexity of the concept.
- Language models and humans differ in their capability for producing explanations. The length of human explanations correlates with the complexity of the concept.

Having a better understanding of language models and its relation to the way other systems, including humans, make inferences is fundamental research that can make the use of these systems safer and more effective, through more robust prompts to solve a variety of tasks. However, this better understanding can also be used maliciously to cunningly modify prompts that alter the results of some applications built on top of language models. A query may fail when using dates, numbers or other strings that have intrinsic patterns. For instance, GPT-3 usually continues "Mary's father has 5 daughters, Lucy, Anne, Alice, Jane and" correctly, but fails on "Mary's father has 5 daughters, Ba, Baba, Bababa, Babababa and", systematically continued with 'Babababaa'.

The rest of the paper is organised as follows. In the next section, we present related work in machine teaching, language models and the comparison of machine and human learning. In Section 3, we introduce our theoretical framework, the notion of the teaching size and the importance of the alignment of priors between teacher and learner. Section 4 contains the experimental design, and introduces the agents we consider. Section 5 presents and analyses the results. Finally, we close the paper with a discussion of the implications of this work.

2 Related Work

It is only recently that deep language models are starting to show some minimal capabilities for few-shot learning. The abstraction abilities of language models have been empirically shown by systems such as GPT-3 that solve "tasks which require to perform simple on-the-fly computational reasoning, recognize a novel pattern that is unlikely to have occurred in training, or adapt quickly to an unusual task" (Brown et al., 2020). Examples of these tasks are simple arithmetic operations and the induction of symbolic manipulations such as anagrams or reversed words. But, are deep language models just memorising a huge corpus, or do they have the capacity of learning abstract concepts from data? This is still an open debate. On the one hand, recurrent neural networks, such as long-short-term-memory (LSTM) architectures, have shown the capacity of capturing syntax-sensitive generalisations such as long distance number agreement (Lakretz et al., 2019), or language modelling for the game of chess (Toshniwal et al., 2021). From a more general perspective, the abstract reasoning capacity of transformers is shown by Hupkes et al. (2020). On the other hand, there are important limitations of language models on simple generalisation tasks (Bender and Koller, 2020; Marcus and Davis, 2020).

The comparison between humans and AI systems has usually been performed with systems that are trained for a task, but not for a variety of tasks that are introduced with a few words or examples. For instance, Kühl et al. (2020) present an experiment that compares the performance of several machine learning algorithms and 44 human participants recognising patterns in 3x3 matrices. The results in this case show that humans, as expected, tend to achieve better performance when few examples are provided. This work highlights that many AI systems are still not ready for a direct and fair comparison with humans when limited examples are given for a range of tasks. Further insight can be obtained when we try to look into the extracted models through explanations, and analyse them in the context of concept complexity. For instance, the relation between complexity of concepts and explanations is discussed by Ai et al. (2021).

Regarding language models, Hendrycks et al. (2021b) compare the performance of a text model with humans on 57 tasks including elementary mathematics, US history, law, and more. Only GPT-3 improves significantly over random chance, but the model is still very far from human performance. Poor performance is also shown for code generation when using text descriptions and a few examples (Hendrycks et al., 2021a). Sinha et al. (2019) show that language models obtain poor robustness and generalisation capabilities when compared to graph neural networks that work with symbolic inputs. The experiments also show that humans (with limited time) performed well in simple problems but the performance decreases with complex examples. Motivated by these limitations, McClelland et al. (2019) propose a modification of the language models architecture to extend the ability of these systems to consider context information inspired in the way humans use languages.

Machine teaching has been previously used to understand how humans teach machines, comparing a machine teaching setting with a curriculum learning setting (Khan et al., 2011; Basu and Christensen, 2013). Many of these studies use feature vector concepts instead of rich representations (Turing-complete) such as natural language. Only recently has machine teaching focused on universal languages, and how relevant simplicity priors become when teaching concepts with very few —or very short— examples (Telle et al., 2019). Machine teaching can determine the minimal set of examples (in number or size) that a learner needs to identify a concept, provided teacher and learner share the same prior. In this paper we propose to do this conversely: by generating examples going from these minimal sets to more redundant sets, we can investigate the priors that the learners have.

3 Machine Teaching and Priors

Machine teaching is often understood as an inverse problem to machine learning (Zhu, 2015). In machine learning, the aim is to induce a concept from a training set. In machine teaching, given a concept, the goal is to obtain the optimal training set from which a learner is able to learn the concept.

Most of the early research in machine teaching was based on the idea of the *teaching dimension*, the minimum number of examples the learner needs to identify a concept. However, the teaching dimension approach is too narrow. For instance, in the case of teaching concepts from richer languages, it has been demonstrated that even if the teaching dimension is small (low cardinality), the examples might be enormously large. Telle et al. (2019) propose a machine teaching framework that is based on the size of examples instead of the dimension of the witness set. While the teaching

dimension is related to algorithmic learning theory, the teaching size is more appropriate for language models and constructive examples. For instance, considering two witness sets of the same teaching dimension, the example '(010101,1)' contains more information than '(0,1)'.

Following Telle et al. (2019), we consider a possibly infinite example space X and a possibly infinite concept class C consisting of concepts over X. Examples are defined as pairs $\langle i, o\rangle$ and concepts as functions mapping inputs to outputs. An example set $S = \{\langle i_1, o_1\rangle, \ldots, \langle i_k, o_k\rangle\}$ is just a finite set of input-output pairs, used as witness for the teaching process. Given a concept $c \in C$ and an example set S, we say that c satisfies S, denoted by $c \vDash S$, if $c(i) = o$ for all the pairs $\langle i, o\rangle$ in S. The empty set is satisfied by all the concepts. We define an encoding function δ as the number of bits needed to encode S, then $\delta(S)$ represents the size of S. The teaching size is defined as:

$$TS(c) = \min_S \{\delta(S) : \{c\} = \{c' \in C : c' \vDash S\}\}$$

A concept c can be implemented in a language L by zero, one or more programs. A program p captures concept c iff for every S we have that $p \vDash S$ iff $c \vDash S$. The equivalence class for c is $Class_L(c) = \{p : \forall S, p \vDash S \iff c \vDash S\}$. The learner Φ can be seen as a function mapping sets to programs. After this, the teaching size for Φ can be reformulated as follows:

$$TS(c) = \min_S \{\delta(S) : \Phi(S) \in Class_L(c)\}$$

Priors can be used as more than one concept might be consistent with one given set S. One natural choice is simplicity, which can be formally defined as the size of the concepts. Let $l(p)$ be the length in bits of a program p in L using an appropriate encoding. Let $\prec$ be the total order of programs ordered by l, where shorter programs precede longer ones. In case of equal l, ties are broken lexicographically. So the teaching size and the learner can now be expressed as:

$$TS_l(c) = \min_S \{\delta(S) : \Phi_l(S) \in Class_L(c)\} \quad \text{with} \quad \Phi_l(S) = arg \min_p^{\prec} \{l(p) : p \vDash S\}$$

From these equations the concept of the *teaching book* derives easily. The *teaching book* is a list consisting of entries in the form $\langle w, p\rangle$ being w the optimal witness set and p the smallest program compatible with w. The book is ordered by the encoding size δ for w. The algorithm enumerates on the size of w and looks for the shortest program p in each case, and does not include $\langle w, p\rangle$ if p has appeared before. All this ensures that no w is repeated and no p is repeated. This setting allows to prove theoretically and empirically that it is possible to find witness sets for explaining concepts whose size is smaller than the programs they identify, "an illuminating justification of why machine teaching from examples makes sense at all" (Telle et al., 2019).

In this work, we build on the increasing relevance of the alignment of priors between learner and teacher (Goodman and Frank, 2016; Yang and Shafto, 2017; Melo et al., 2018) to investigate how long witness sets have to be for different learners (language models, program induction systems and humans), when devising the 'teaching examples' using simplicity (Occam's razor) as a prior.

4 Experimental Setting

The questions we want to explore experimentally are:

- Q1 - Abstraction capability: Is a language model such as GPT able to capture algorithmic patterns from very few examples selected with a simplicity prior?
- Q2 - Learning curves: Does accuracy increase from the theoretically minimal witness sets needed for teaching the concept with additional examples?
- Q3 - Characteristic curves: How do language models, program induction approaches and humans behave in terms of the complexity of the patterns?
- Q4 - Comparison: How do language models compare with humans and program induction approaches in terms of performance curves?
- Q5 - Architecture: Apart from being more capable, which has already been shown before, do larger models have a stronger preference for simple concepts than smaller models?
- Q6 - Explanations: Can we get explanations from GPT for these algorithmic patterns? Is the accuracy of human explanations related to performance, and their size related to complexity?

Before introducing the experimental setting, we present the algorithmic language we will use and the machine teaching algorithm that selects the optimal examples.

4.1 The Machine Teaching Setting

The language L we will use to express programs will be the simple but Turing-complete programming language P3, a variation of P" (Böhm, 1964) with only 7 instructions. We employ P3 as a concept language because it is completely unrelated to the way language models work internally (transformers) and it is different from any natural language and the languages used by the program induction systems.

P3 programs operate on an input tape and a write-only output tape, divided into cells, which can hold any of the three characters in the alphabet: $\Sigma = \{$'0', '1', '.'$\}$. The pointer is moved using the instructions '$<$' (left) and '$>$' (right). The instructions '+' and '$-$' are used to respectively increase or decrease the value of the pointed cell, in the order they are defined in Σ. The instructions to perform loops are '[' and ']', being '[' the start and ']' the end of the loop, exiting the loop when the pointer is on '.'. Finally, the instruction 'o' is used to output the value of a pointed cell.

We follow Telle et al. (2019) to generate P3 concepts and the associated witness sets as teaching examples. The example set is restricted to binary strings, with '.' used as delimiter, padding cells before and after the string. We use a maximum teaching size of 7 (total number of bits in all the inputs and outputs in the witness set). Removing repeated sets and contradictory examples, the example set space is hence comprised of 17,252 example sets. Note that the teaching procedure and its teaching book assumes a perfect model of the learner (the teacher knows the learner prior).

4.2 Choice of Concepts and Examples

We extract eight different concepts $c_1, \ldots, c_8$, each c_i being randomly selected from all programs in the Teaching Book with i instructions. We bin the concepts into four levels: the programs with 1-2 operators (very low complexity, VL-C), those with 3-4 operators (low complexity, L-C), those with 5-6 operators (high complexity, H-C) and those with 7-8 operators (very high complexity, VH-C).

The minimal witness sets for these concepts in the teaching book are all between 1 and 3 examples. This will be the first batch (the witness set, WS) used as training set. As stated before, in this paper we want to test if language models (and other learners, including humans) are capable of capturing patterns when a strong simplicity prior is assumed. As the learner prior might differ from the teacher prior, we want to analyse how learning improves as more examples are provided. Therefore, we will use 5 additional examples in two different batches, a first batch with 2 extra examples (additional set one, AS I) and a second batch with 3 extra examples (additional set two, AS II).

For the test examples, we generate 5 examples per concept, totalling $8 \times 3 \times 5 = 120$ score values. This number of concepts, batches and test examples aims to find a trade-off for a minimally robust evaluation and the limitations of conducting the experiments with humans.

With all this, the proposed teaching protocol for each 'concept lesson' is divided into three phases. In all cases we ask the learner to complete the test examples. In the first phase, we present only the witness set (WS). In the second phase, we also provide the learner the examples of AS I (WS + AS I). Finally, in the third phase, we show all the available examples (WS + AS I + AS II).

Table 1: Experimental setting for the learning of concepts c_1 and c_4.

Id	P3	Description (not used)	WS	AS I	AS II	Test
c_1	o	*Print the first character of a string*	{('0','0')}	{('111001','1'), ('110101','1')}	{('100110','1'), ('01010','0'), ('111100','1')}	{('00000','0'), ('11100','1'), ('00111','0'), ('11010','1'), ('0010', '0')}
c_4	o+oo	*If input is '', print '00'; else if input starts with '0', print '001'; else print '1'*	{('0','011')}	{('10','1'), ('001','011')}	{('00','011'), ('0001','011'), ('000','011')}	{('01011','011'), ('0101','011'), ('0010','011'), ('100','1'), ('1','1')}

4.3 Learners

The following learner families have been considered.[1]

[1]The code and experimental data used in this paper can be found at `https://bit.ly/Code3357`

Humans: We devised a form that was completed by 30 human participants with ages between 18 and 54, and 28 of them having university-level education. Apart from the answers to each test example, humans had to write an explanation in natural language for each of the 8 concepts after the third phase. All data collected in the experiments were anonymised, informing the participants of the data treatment and how they could apply their rights, complying with the current data protection laws.[2]

MagicHaskeller (**MH**): A general-purpose inductive functional programming system based on systematic exhaustive search (Katayama, 2011). It infers programs expressed in Haskell from input-output examples. A strong simplicity prior and a library of Haskell primitives allow MagicHaskeller to solve many problems using only one example. When MH returns more than one hypothesis, we take the first one.

Louise: An inductive logic programming system based on Top program construction (Patsantzis and Muggleton, 2021). It learns Prolog programs from finite examples, with the capability of learning recursive programs in polynomial time. Two different approaches are considered, where the left and right recursive calls to the target predicate are omitted (as they end up in infinite loops):

- **Louise-ens.** The most frequent result returned by the totality of the Top program will be used as the final answer. Ties are broken randomly.
- **Louise-exp.** The expected score is calculated considering all the possible solutions returned by the Top program.

GPT-2: A transformer-based language model trained on a dataset of 8 million web pages (Radford et al., 2019). The model returns the next predicted word or token considering the previous tokens as the given context. We use the 774M-parameters version of GPT-2. The default hyperparameters of the model are preserved, with the exception of the 'length' of the output returned, which is modified to return 3 tokens. By doing so, the experiment execution performance is improved, without asking GPT-2 for more information than what we require for our analysis. Deterministic results can be obtained by setting the 'top-k' parameter to 1, but we can get better performance if we use several continuations. Accordingly we will use the default 'top-k' value, which is 0. With this, two different approaches will be considered by running the model extrapolations n=40 times for the same query:

- **GPT-2-ens.** The most frequent result (out of the n results obtained for the same query) will be used as the final answer. Ties are broken randomly. This approach can be understood as a majority ensemble setting.
- **GPT-2-exp.** Here, the n results for the same query are averaged. This approach can be understood as the expected value for GPT-2.

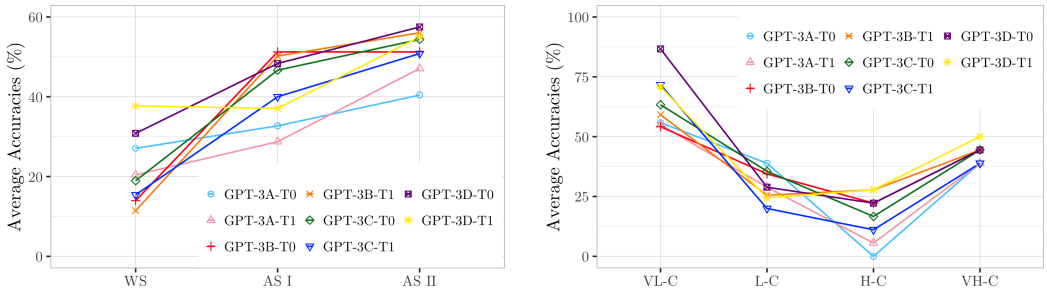
GPT-3: The latest approach from the GPT family (Brown et al., 2020). We used different models available, with an increasing number of parameters (Ada, Babbage, Curie and Da Vinci), as recent work (Bender et al., 2021) demonstrate that larger models outperform smaller ones. We want to show whether this trend holds for abstract concepts too, with Occam's razor emerging more strongly for larger models. We will employ acronyms GPT-3A, GPT-3B, GPT-3C, GPT-3D correspondingly. We performed the experiment considering two different settings: the hyperparameter temperature set to 0 and to 1. When the temperature is set to 1, the results are non-deterministic, and the same approach used in GPT-2-ens is applied. We will use T0 or T1 as shortcuts depending on the value of this parameter. For instance, GPT-3 Da Vinci with temperature 0 will be denoted by GPT-3D-T0. Prompts were designed similarly to GPT-2. Finally, we also asked GPT-3 to provide an explanation of the concepts, just after each batch (8×3 explanations).[3]

5 Results

We take the results of each of the learners in the previous section. For humans, we aggregate the results into a single value per concept, batch and test example, by calculating the average of the 30 human subjects. Performance is represented using learning curves and characteristic curves. A learning curve depicts the variation of accuracy according to the number of examples, which in our

[2]The experiment form used in the human experiments can be found at `https://bit.ly/LearningTest3357`.

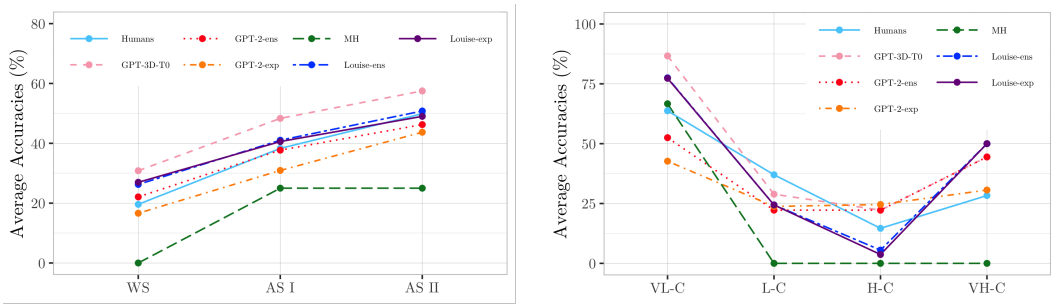
[3]Complete details of the prompts are included in the supplementary material.



(a) Mean GPT-3 accuracy by teaching batch.

(b) Mean GPT-3 accuracy by concept complexity level.

Figure 1: Figure (a) shows the average accuracies for eight GPT-3 configurations and the three teaching set batches. WS refers to the first (optimal) teaching set, AS I refers to these plus two additional examples and AS II refers to all these plus the final three additional examples. Figure (b) shows the average accuracies for eight GPT-3 configurations and the four concept complexity levels (VL-C, L-C, H-C and VH-C refer to programs with 1-2, 3-4, 5-6 and 7-8 instructions, respectively).



(a) Mean accuracy by teaching batch.

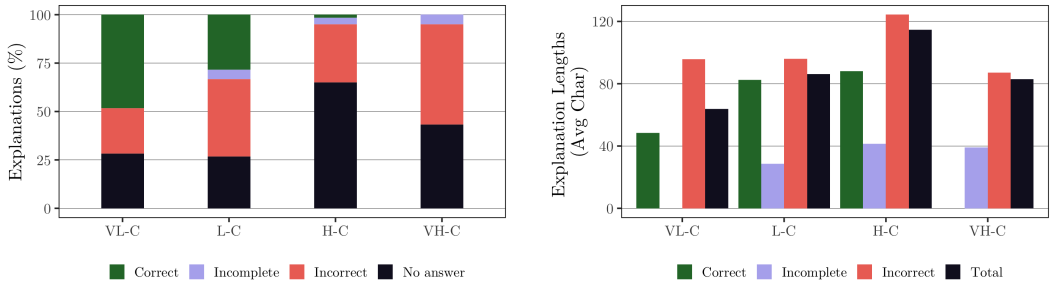
(b) Mean accuracy by concept complexity level.

Figure 2: Figure (a) presents the average accuracies for seven learning agents shown for the three teaching set batches. Interpretation as in Figure 1a. Figure (b) presents the average accuracies for seven learning agents shown for the four complexity levels. Interpretation as in Figure 1b

case is grouped into the batches WS, AS I and AS II. Characteristic curves depict the variation of accuracy in terms of the complexity level of the concepts: VL-C, L-C, H-C and VH-C.

We start with the analysis for the eight GPT-3 parameter configurations. The learning curves in Figure 1a are clearly increasing in general. The first observation is that the systems with the temperature set to 0 obtain better performance than those with the temperature set to 1. There is no clear evidence that larger models in terms of the number of parameters present a better performance for this kind of patterns, which are solely based on simplicity and not other priors. Regarding the characteristic curves in Figure 1b, the GPT-3 systems show a decreasing trend in performance as the complexity increases. Once again, there is no clear pattern in terms of the number of the parameters. Nevertheless, GPT-3D-T0 is the best method, and both GPT-3D versions (the largest models) show their advantage in the very few shot case (WS) and low complexity (VL-C). In other words, more parameters seem to make a difference for easier concepts with fewer examples.

Figure 2a shows the learning curves for humans, MH, the two variations for Louise and a selection of GPT methods. GPT-3D-T0 turns out to be the system with the highest average performance for every teaching batch. Louise approaches follow, with a similar performance between them. Humans and GPT-2-ens go next, with similar general performance in the experiment. GPT-2-exp remains slightly under the average performance of the ensemble (majority) approach. Finally, MH is the poorest system, especially for WS, the case with very few examples. For all systems, the accuracy increases as additional examples are provided (AS I and AS II).



(a) Types of human explanations.

(b) Size of human explanations.

Figure 3: Figure (a) shows a breakout of the quality of human explanations after batch AS II. Correct answers are those that explain the whole concept correctly, incomplete answers explain part of the concept correctly and incorrect answers explain any part of the concept incorrectly. Figure (b) shows the average lengths of the different explanations provided by the human participants after batch AS II.

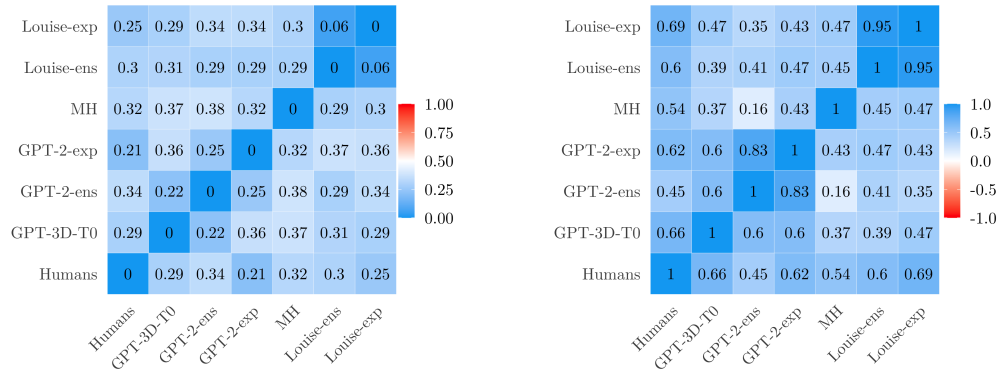
Figure 2b shows the characteristic curves. We see curves cross more often, but overall the order of the learners in terms of performance is similar to the learning curves. The evolution of performance in terms of complexity is generally decreasing, with the best results clearly being for concepts of very low complexity (VL-C). Looking into the results in more detail (see supplementary material). we explain the small increase in the last group (VH-C) as a misalignment between the systems and the teacher, whose representational language is P3. The prior (Occam's razor) depends on the representational language and the background knowledge in these systems. For instance, if one concept is 'reverse' and we have the reverse function in the background knowledge, the program using it is going to be short. As P3 does not have background knowledge, the concept is categorised as complex (it has 8 instructions) even if GPT-3 and Louise, which rely more on background knowledge, can solve it.

When looking at the explanations, only those given by MH and humans were minimally acceptable. Despite having tried many kinds of prompts for the GPT systems, as discussed in the supplementary material, we did not get good explanations consistently. For Louise, despite being interpretable, the solutions usually had too many clauses to serve as explanations. As observed in Figure 3a, the percentage of correct natural language explanations decreases as the complexity of the concepts to learn increases. On the other hand, Figure 3b shows how the average size of the correct explanations increases with the complexity of the concepts (size is measured as the number of characters in the natural language explanation given by humans). Remarkably, the average size of the different types of explanations present the opposite behaviour to performance considering the concept complexity bins in Figure 2b. Even for incorrect explanations, the perceived complexity in terms of the size of the explanation inversely corresponds to the accuracy results.

Finally, comparing learners by batches of examples or complexity of the concepts as we have seen in the previous plots does not really provide a fully fine-grained metric of how much similar the systems are. This is what we see in Figure 4, showing the error disagreement (proportion of cases where one method is right and the other is wrong, or vice versa) (Figure 4a) and the Phi coefficient (Figure 4b) taking all the examples individually. The matrix for error disagreement strongly depends on performance, so the results for Phi coefficient are more informative. We see that the two Louise approaches have the highest correlation between them. When compared with humans, GPT-3D-T0 and Louise-exp are the most similar ones.

6 Discussion

We can now answer the questions we formulated in Section 4. We see that only for easy concepts (VL-C), the simplicity prior leads to example sets that humans, Louise and some GPT models can identify consistently (Q1, Q3). For more complex concepts the behaviour is more erratic, and may depend on spurious factors, as the explanations are incorrect for those very hard cases having some accuracy. The learning curves are increasing for all systems, as more examples are given (Q2).



(a) Error disagreement matrix.

(b) Phi coefficient matrix.

Figure 4: (a) Error disagreement matrix for performance and (b) Phi coefficient matrix for the behaviour of seven learning agents.

The minimal set is rarely sufficient, which means that no system prior is perfectly aligned with the simplicity prior that derives from P3 program size. This is expected as the internal representations of these systems are different. Both humans and GPT have priors to capture both extrinsic and intrinsic patterns, as language models have been trained using natural language texts created by humans. This seems consistent with large language models having a comparable performance to humans. However, this would not explain cases in which GPT-3 outperforms humans, unless (1) our population behaves differently (worse) than the population behind the training corpora used for learning them, and/or (2) GPT-3 has generalised some very abstract latent concepts that are applicable for simple intrinsic patterns, where GPT-3 seems to excel (VL-C). Finally, MH is the system that behaves most like an ideal learner in our setting, but with very limited capability. The difference in the performance between MH and Louise (both being inductive programming systems) might be due to the differences in their background knowledge.

A more detailed analysis in terms of correlation of errors shows that GPT-3D-T0 is one of the most similar systems to humans (Q4), so the extra performance is not explained by being more deviated from human priors, but perhaps from higher robustness. When we look at architecture size, we do not see a monotonic relation between results and number of parameters (Q5), but the correlation is clear and GPT-3D, for the two variants that we have studied, achieves the best results consistently. These results may suggest that *larger models can be better at capturing patterns that depend on simplicity priors, distilling some kind of Occam's razor for few-shot inference*. However, some other experiments in the supplementary material with concepts including loops suggest that larger models in the future may not be able to solve complex algorithmic concepts, with this distillation being limited to sequential (if-then-else) algorithmic patterns. New paradigms beyond transformers may be needed in the future.

Explanations should give us more insight about what these systems are doing. Unfortunately, we were not able to find a working prompt so that GPT-3 could generate good explanations for the learnt concepts (Q6). Regarding the explanations provided by humans, the results suggest that the higher the complexity, the higher the size of the correct explanations to cover the learnt concepts. Furthermore, as complexity increases, the percentage of correct explanations decreases.

There are some limitations in this work. First, while we use MH as a system that would only be dominated by the simplicity priors, it is not very capable, and it is only useful to disentangle the behaviour for very easy concepts. Louise adds robustness to this analysis but it has more dependency on background knowledge. Second, the use of humans for comparisons limits the number of concepts and test examples that we can do for a reasonable questionnaire, and more concepts were tried only for GPT-3 (see supplementary material). Overall, we think that the machine teaching approach can bring very important insights about what language models can do, what their acquired priors are, and how many examples they need.

References

Ai, L., Muggleton, S. H., Hocquette, C., Gromowski, M., and Schmid, U. (2021). Beneficial and harmful explanatory machine learning.

Basu, S. and Christensen, J. (2013). Teaching classification boundaries to humans. In *AAAI*.

Bender, E. M., Gebru, T., McMillan-Major, A., and Shmitchell, S. (2021). On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, page 610–623.

Bender, E. M. and Koller, A. (2020). Climbing towards NLU: On meaning, form, and understanding in the age of data. In *ACL*, pages 5185–5198.

Böhm, C. (1964). On a family of Turing machines and the related programming language. *ICC Bulletin*, 3(3):187–194.

Brown, T. B., Mann, B., Ryder, N., et al. (2020). Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Cakmak, M. and Thomaz, A. L. (2014). Eliciting good teaching from humans for machine learners. *Artificial Intelligence*, 217:198–215.

Chater, N. and Vitányi, P. (2003). Simplicity: A unifying principle in cognitive science? *Trends in Cognitive Sciences*, 7(1):19–22.

Degen, J., Hawkins, R. D., Graf, C., Kreiss, E., and Goodman, N. D. (2020). When redundancy is useful: A bayesian approach to "overinformative" referring expressions. *Psychological Review*.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Goodman, N. D. and Frank, M. C. (2016). Pragmatic language interpretation as probabilistic inference. *Trends in Cognitive Sciences*, 20(11):818–829.

Hendrycks, D., Basart, S., Kadavath, S., Mazeika, M., Arora, A., Guo, E., Burns, C., Puranik, S., He, H., Song, D., and Steinhardt, J. (2021a). Measuring coding challenge competence with APPS.

Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., and Steinhardt, J. (2021b). Measuring massive multitask language understanding. *ICLR*.

Hupkes, D., Dankers, V., Mul, M., and Bruni, E. (2020). Compositionality decomposed: How do neural networks generalise? *Journal of Artificial Intelligence Research*, 67:757–795.

Izacard, G. and Grave, E. (2020). Leveraging passage retrieval with generative models for open domain question answering. *arXiv preprint arXiv:2007.01282*.

Katayama, S. (2011). Magichaskeller: System demonstration. *AAIP 2011*, pages 63–70.

Khan, F., Mutlu, B., and Zhu, J. (2011). How do humans teach: On curriculum learning and teaching dimension. In *NIPS*, pages 1449–1457.

Kühl, N., Goutier, M., Baier, L., Wolff, C., and Martin, D. (2020). Human vs. supervised machine learning: Who learns patterns faster? *arXiv preprint arXiv:2012.03661*.

Lakretz, Y., Kruszewski, G., et al. (2019). The emergence of number and syntax units in LSTM language models. In *NAACL 2019*, pages 11–20.

Marcus, G. and Davis, E. (2020). GPT-3, bloviator: OpenAI's language generator has no idea what it's talking about. *Technology Review*.

McClelland, J. L., Hill, F., Rudolph, M., Baldridge, J., and Schütze, H. (2019). Extending machine language models toward human-level language understanding. *arXiv:1912.05877*.

Melo, F. S., Guerra, C., and Lopes, M. (2018). Interactive optimal teaching with unknown learners. In *27th International Joint Conference on Artificial Intelligence*, pages 2567–2573.

Patsantzis, S. and Muggleton, S. (2021). Top program construction for polynomial meta-interpretive learning. *Machine Learning*.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Sinha, K., Sodhani, S., Dong, J., Pineau, J., and Hamilton, W. L. (2019). CLUTRR: A diagnostic benchmark for inductive reasoning from text. *arXiv preprint arXiv:1908.06177*.

Sun, W., Nasraoui, O., and Shafto, P. (2020). Evolution and impact of bias in human and machine learning algorithm interaction. *Plos one*, 15(8):e0235502.

Telle, J. A., Hernández-Orallo, J., and Ferri, C. (2019). The teaching size: computable teachers and learners for universal languages. *Machine Learning*, 108(8-9):1653–1675.

Toshniwal, S., Wiseman, S., Livescu, K., and Gimpel, K. (2021). Learning chess blindfolded. `https://openreview.net/pdf?id=DGIXvEAJVd`.

Vaswani, A., Shazeer, N., et al. (2017). Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Xu, S., Semnani, S. J., Campagna, G., and Lam, M. S. (2020). AutoQA: From databases to QA semantic parsers with only synthetic training data. *EMNLP*.

Yang, S. C.-H. and Shafto, P. (2017). Explainable artificial intelligence via Bayesian teaching. In *Workshop on Teaching Machines, Robots, and Humans*.

Zhu, X. (2015). Machine teaching: An inverse problem to machine learning and an approach toward optimal education. In *AAAI*, pages 4083–4087.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See **Discussion.**
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See **Introduction.**
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] See **Introduction** and **Learners**.
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See **Learners** links and **Supplementary Material**.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See **Supplementary Material**.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] We calculated the error bars for those systems that present variability on the results (GPT-3) or that represent a population (differently parameterised GPT-2 or humans), but not for the rest of the cases as the systems are deterministic or there are very few instances. See **Supplementary Material**.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See **Supplementary Material**.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [Yes] See **Learners**.[1]

(b) Did you mention the license of the assets? [Yes] See **Supplementary Material**.

(c) Did you include any new assets either in the supplemental material or as a URL? [Yes] See **Learners** links.

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] See **Learners** experiment form link.

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] See **Learners**.

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [Yes] See **Learners** experiment form link.

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [No] There were no potential participant risks identified.

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [Yes] See **Learners** experiment form link.

[1]We thank OpenAI for giving us early access to GPT-3 through their API.