Position: Transformers Have the Potential to Achieve AGI

Anonymous Authors[1]

Abstract

As large language models (LLMs) based on the Transformer architecture continue to achieve impressive performance across diverse tasks, this paper explores whether Transformers can ultimately achieve artificial general intelligence (AGI). We argue that Transformers have significant potential to achieve AGI, supported by the following insights and arguments. (1) A Transformer is expressive enough to simulate a programmable computer equipped with random number generators and, in particular, to execute programs for meta-tasks such as algorithm design. (2) By the Extended Church-Turing thesis, if some realistic intelligence system (say, a human with pencil and paper) achieves AGI, then in principle a single Transformer can replicate this capability; Besides, we suggest that Transformers are well-suited to approximate human intelligence, because they effectively integrate knowledge and functions represented in network form (e.g. pattern recognition) with logic reasoning abilities. (3) We argue that Transformers offer a promising practical approximation of Hutter's AIXI agent, which is an ideal construction to achieve AGI but is uncomputable.

1. Introduction

Large language models (LLMs) (Achiam et al., 2023; Gemini et al., 2023; Anthropic, 2024; Dubey et al., 2024) have demonstrated remarkable capabilities across a broad range of challenging tasks. For example, OpenAI's o-series (OpenAI, 2024) model achieves 71.7% accuracy on the software engineering benchmark SWE-bench (Jimenez et al., 2023), 87.7% on the graduate-level question answering task GPQA (Rein et al., 2023), and 96.7% on a competition-level mathematics reasoning task (Hendrycks et al., 2021). Notably, these results surpass human-expert performance. As LLMs evolve, their capabilities are expected to advance further. These successes are grounded in the Transformer architecture (Vaswani et al., 2017), which has proven to be highly effective across a wide range of domains, extending beyond natural language processing to areas such as computer version (Dosovitskiy, 2020) and decision-making (Chen et al., 2021). Given the impressive achievements of Transformers in tackling challenging tasks across various domains, a fundamental question arises:

Question 1: Can Transformers ultimately achieve artificial general intelligence (AGI)?

To answer this question, we must first establish a rigorous definition of intelligence. Intelligence is multifaceted, encompassing abilities such as creativity, problem-solving, pattern recognition, classification, and reasoning. However, formulating a single, comprehensive definition that captures all these aspects is challenging. As pointed out by Hutter (2005), most, if not all, aspects of intelligence can be framed in terms of goal-driven behavior, or more precisely, as the maximization of some (often unknown) utility (reward) function. This aligns with the "reward is enough" hypothesis (Silver et al., 2021), which suggests that the pursuit of maximizing reward alone is sufficient to drive behaviors that exhibit a wide range of capabilities, many of which are traditionally studied in both natural and artificial intelligence.

In this paper, we follow the definition that intelligence can be broadly categorized into two types of reasoning abilities:

- Learning an unknown utility function (inductive reasoning): This involves drawing generalizations from specific observations, where the conclusions are probable but not certain. This type of reasoning is extensively explored in the context of inverse reinforcement learning (Ng et al., 2000; Hadfield-Menell et al., 2017). Examples of inductive reasoning include pattern recognition, natural language processing, prediction, and scientific research, where repeated observations lead to hypotheses or theories.
- Maximizing a known utility function (deductive reasoning): In this case, the solution depends entirely on the explicit, provided information. Successful applications includes AlphaGo (Silver et al., 2016), Muzero (Schrittwieser et al., 2020), AlphaProof (AlphaProof

[1] Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

& Teams, 2024), and OpenAI-o1 (OpenAI, 2024).

In this paper, we argue in favor of Question 1, **supporting the potential of Transformers to achieve AGI** with the following insights and arguments.

1. A single Transformer can simulate a probabilistic programmable computer. Prior works (e.g. (Merrill & Sabharwal, 2024)) have shown that Transformers (with chain-of-thoughts) can efficiently simulate deterministic Turing machines (DTMs). We extend this result to the potentially more powerful probabilistic Turing machine (PTM) model, proving that Transformers can efficiently simulate PTMs as well (Theorem 2.2).

At first glance, Theorem 2.2 may suggest adherence to a one-model-one-task paradigm, where different tasks require different transformers. This misaligns with the current practice of training a single general-purpose transformer to perform various tasks. In fact, Theorem 2.2 provides deeper insights: as also observed in related work (e.g. (Qiu et al., 2024)), it implies that a *single* Transformer can simulate a probabilistic universal Turing machine (UTM), a formalization of a general-purpose programmable computer equipped with random number generators.

Furthermore, while Transformers do not follow the one-model-one-task paradigm, they appear to adhere to a one-prompt-one-task paradigm, where different tasks require different PTMs (or equivalently, programs) to be specified in the prompt or pre-injected during training. We argue that this is *not* the case. Specifically, beyond algorithms for specific tasks, a PTM T can also serve as a program for meta-tasks, such as designing other algorithms (meta-algorithms), or even higher-order tasks, such as meta-meta-algorithms.

2. Implication of the Extended Church-Turing thesis. The *Extended Church-Turing thesis* (ECT) (Yao, 2003; Aharonov & Vazirani, 2013), an extension of the Church-Turing thesis in the modern computer science literature from a complexity-theoretic perspective, asserts that the PTM model is not as expressive as but also as efficient as any realistic physical device (say, a human brain, a society, or a future neural network). Specifically, any function that can be computed by a realistic finite physical system can also be computed by a PTM with at most a polynomial slowdown. Consequently, if some realistic intelligence system (say, a human brain with pencil and paper) achieves AGI, then in principle, a single Transformer can achieve AGI as well (Thesis 1).

In particular, Thesis 1 suggests that a single Transformer has the potential to achieve human-level intelligence. Moreover, we suggest that Transformers are particularly well-suited as approximations of human intelligence, because they effectively integrate knowledge and functions represented in network form with logical reasoning abilities, and thus can leverage results from both connectionism AI and symbolicism AI.

3. Algorithmic approximations of general intelligence: Besides mimicking the human reasoning process, another line of research, inspired by algorithmic information theory, seeks to reach or even outperform human intelligence by establishing a formal theory of general intelligence. Several constructions have been proposed to address meta-tasks, including:

Levin's universal search algorithm: Many deductive reasoning tasks, such as theorem proving, planning, and general NP-complete problems, can be effectively modeled as search problems. Levin's universal search is an algorithm that can solve all search problems as quickly as the fastest algorithm for each, up to a large constant factor (Levin, 1973; 1984). The basic idea is to run all programs p in parallel with relative computation time $2^{-\ell(p)}$; i.e. a time fraction $2^{-\ell(p)}$ is dedicated to executing p. Here, we describe programs as Boolean strings using a prefix-free encoding, where $\ell(p)$ denotes the length of the description of p. Note that the sum of all these time fractions satisfies $\sum_p 2^{-\ell(p)} \leq 1$.

Solomonoff's universal induction: Every inductive reasoning task, such as continuing a number of series in an IQ test, classification in machine learning, stock-market forecasting, or scientific research, can be described as a sequence prediction problem, more precisely, predicting future data from past observations (Hutter, 2005). Solomonoff's universal induction (Solomonoff, 1964; 1978) is an optimal approach for all sequence prediction problems, where the data is sampled from a computable probability distribution, or equivalently generated by a realistic physical system according to the physical version of Church-Turing thesis. The basic idea is to do Bayesian prediction, using Solomonoff prior as the prior belief, which assigns higher probabilities to simpler hypotheses with shorter descriptions, aligning with Occam's razor.

Hutter's AIXI agent: AIXI is a theoretical agent that achieves AGI (Hutter, 2005). AIXI is somehow a combination of Solomonoff's universal induction and Levin's universal search. Specifically, AIXI replaces the unknown environment in the Bellman equation with a generalized Solomonoff prior and then invokes $M^{\epsilon}_{p^*}$ (Hutter, 2005), an enhancement of Levin's universal search, to solve the Bellman equation. Like Solomonoff induction, AIXI tends to hypothesize the environment as shortest possible programs, in line with Occam's razor.

While these constructions are theoretically optimal, they are often intractable in practice or even uncomputable. However, we argue that Transformers provide a promising tractable approximation of these universal constructions. Specifically,

Universal search: By Theorem 2.2, a single Transformer, by simulating Levin search, can theoretically solve all search problems as efficiently as the fastest algorithm for each problem. To enhance tractability, Transformers can leverage prior knowledge embedded during training to assign the relative computation time proportion in a more adaptive and efficient way. In addition, Transformers can continually refine search strategies by learning from past experiences.

Universal induction: Recent works (Müller et al., 2022; Hollmann et al., 2023; Grau-Moya et al., 2024; Goldblum et al., 2024; Young & Witbrock, 2024) have demonstrated that Transformers align with Occam's razor, the core principle of Solomonoff induction. Specifically, Transformers tend to output sequences generated by shorter programs (a.k.a. with lower Kolmogorov complexity). The alignment with Occam's razor enables Transformers to generalize effectively across diverse tasks and data modalities, making them good approximations of general-purpose predictors. Furthermore, Young & Witbrock (2024) put forth and explore a hypothesis that Transformers approximate Solomonoff induction better than any other extant sequence prediction method, highlighting their potential as practical implementations of universal induction.

AIXI agent: we suggest that Transformers have the potential to offer a practical approximation of AIXI for the following reasons: as we just discussed, (i) Transformers have the potential to approximately implement Solomonoff universal induction; (ii) Transformers has potential to implement universal search in practice, enabling efficient solutions for a wide range of deductive reasoning tasks; and (iii) Transformers integrate prior knowledge effectively, leveraging human experience to enhance their practical applicability.

2. Transformers Can Efficiently Simulate Probabilistic Programmable Computers

We assume that the reader is familiar with the definitions of the Transformer architecture, Turing machine (TM), probabilistic Turing machine (PTM), and universal Turing machine (TUM). For the convenience of readers, we present a background of TM in the appendix.

2.1. Transformers Can Efficiently Simulate PTMs

There is a line of theoretical works (Pérez et al., 2019; Bhattamishra et al., 2020; Pérez et al., 2021; Schuurmans, 2023; Giannou et al., 2023; Merrill & Sabharwal, 2024; Hou et al., 2024; Liu et al., 2024; Qiu et al., 2024) studying the expressive power of Transformer with chain of thought (CoT) by connecting them with Turing machines. It turns out that decoder-only Transformers with t CoT steps can simulate t DTM steps.

Theorem 2.1 (Merrill & Sabharwal (2024))**.** *Let T be a deterministic Turing machine that, on input x of length n, runs for at most $t(n)$ steps. There is a constant-depth decoder-only Transformer that, on input x, takes $t(n)$ CoT steps and then outputs $T(x)$.*

In this paper, we extend Theorem 2.1 to PTMs. In particular, it implies that Transformers with polynomial CoT steps can solve all problems in BPP, the class of decision problems solvable by a PTM in polynomial time, which is strictly larger than the P class, which consists of all decision problems solvable by a DTM in polynomial time, unless BPP = P.

Theorem 2.2. *Let T be a probabilistic Turing machine that, on input x of length n, runs for at most $t(n)$ steps. There is a constant-depth decoder-only transformer that, on input x, takes at most $2t(n)$ CoT steps and returns the same (randomized) output as T.*

Proof. We first adapt T by introducing a lazy sampling of the coin tape. The coin tape is initially empty, filled with blank symbols $\perp$, and will be assigned random coins on the fly during execution. At one step, if T reads a blank symbol $\perp$ from the coin tape, it first tosses a fair coin and writes the result on the coin tape at the current head position. Note that the adapted T, denoted by T', runs for at most $2t(n)$ steps, since each original step may include an additional coin-tossing operation.

Next, we demonstrate how a transformer can simulate T' with at most $2t(n)$ CoT steps. We adapt the proof of Theorem 2 in (Merrill & Sabharwal, 2024). For the i-th step of T', let $h_i^\tau \in \mathbb{Z}$ and $\gamma_i^\tau \in \Sigma$ denote the head position and the content on tape τ, and let $q_i \in Q$ denote the state. Let $\Delta := Q \times \Sigma^2 \times \{L, S, R\}^3$, and $\delta_i \in \Delta$ denote the log at the i-th step, indicating the state entered, symbols written, and directions moved. The crucial observation is that the tape contents at the current head positions can be reconstructed from the input x and the previous logs $\delta_0, \delta_1, \cdots, \delta_{i-1}$.

As shown in (Merrill & Sabharwal, 2024), a Transformer can first obtain all arguments $(q_{i-1}, \gamma_i^1, \gamma_i^2, \gamma_i^3)$ for the transition function. Suppose tape 3 is the coin tape. If $\gamma_i^3 \neq \perp$, which means that the i-th step of T' will be deterministic rather than the coin-tossing operation, then the Transformer computes $\delta_i = \delta(q_{i-1}, \gamma_i^1, \gamma_i^2, \gamma_i^3)$ with a feedforward net outputting the one-hot encoding of δ_i. If $\gamma_i^3 = \perp$, which means that the i-th step is a coin-tossing operation, then the Transformer outputs the equally weighted linear combination of the one-hot encodings of $(q_{i-1}, \gamma_i^1, \gamma_i^2, 0, S, S, S)$ and $(q_{i-1}, \gamma_i^1, \gamma_i^2, 1, S, S, S)$. The vector outputted by the feedforward net is then processed by the finial token classification head, which is a softmax function. One can check that the Transformer exactly simulates the i-th step of T'. □

Remark 2.3. The Transformer architecture in Theorem 2.1 is not exactly the same as real-world transformers, as it makes several assumptions, such as using saturated attention (a.k.a. hardmax attention) instead of softmax attention and allowing log-precision, that each token has $O(\log(n + t(n))$ bits. These assumptions carry over to Theorem 2.2. An important direction is to remove these assumptions and explore how real-world Transformers can efficiently simulate PTMs.

2.2. Transformers Can Efficiently Simulate UTMs

At first glance, Theorem 2.2 appears to follow the one-model-one-task paradigm: different tasks require different Transformers. This misaligns with the current practice of training a single general-purpose transformer to perform various tasks. In fact, Theorem 2.2 provides deeper insights: as also observed in related works (e.g. (Qiu et al., 2024)), Theorem 2.2 implies that a *single* Transformer can simulate a UTM, or equivalently a programmable computer equipped with random number generators.

Though Transformers do not have to follow the one-model-one-task paradigm, they appear to follow the one-prompt-one-task paradigm: for different tasks, different PTMs should be loaded into the prompt or pre-injected during training. We argue that this is not the case. Specifically, beyond algorithms for specific tasks, the PTM T can also be taken as a program that performs meta-tasks, such as a meta-algorithm that designs algorithms, or even meta-meta-algorithms. For example, the Transformer can implement the following meta procedure: it takes a problem description and an input x as input, and then

1. run some prescribed meta-algorithm to initialize or update a program p;
2. run p on input x, and obtain $p(x)$;
3. evaluate $p(x)$. If not good enough, then go to Step 1.

3. The Extended Church-Turing Thesis and Its Implication

3.1. The Extended Church-Turing Thesis

The physical version of *Church-Turing thesis* (CT), as known as Deutsch-Wolfram thesis, asserts (Wolfram, 1985; Deutsch, 1985; Copeland & Shagrir, 2018) that every finite physical system (say, a modern personal computer, a human brain, a society, or a future neural network) can be simulated to any specified degree of accuracy by a PTM. Furthermore, there is also a strengthening, referred to as the *Extended Church-Turing thesis* (ECT), of the physical Church-Turing thesis in the modern computer science literature (Yao, 2003; Aharonov & Vazirani, 2013) from a complexity-theoretic perspective, asserting that the probabilistic Turing machine model is also as efficient as any computing device can be. That is, if a function is computable by some hardware device in time $T(n)$ for the input of size n, then it is computable by a PTM in time $O(T(n)^k)$ for some constant k.

Remark 3.1. The physical version of CT and ECT are very different from the original version proposed by Church and Turing in the 1930s (Church, 1936; Turing, 1937), which asserts that every algorithmic process can be carried out by a PTM. Specifically, if a task can solved by a human being with paper and pencil by following a finite number of exact instructions, then the original CT asserts that it can also be solved by a PTM. Notably, no insight, intuition, or ingenuity is demanded on the part of the human being carrying out the method, which is very different from the physical version. The original Church-Turing thesis is something between a theorem and a definition. And the physical version and ECT are neither mathematical theorems nor definitions. If they are true, then the truth is a consequence of the laws of physics (of Philosophy, 2023).

By combining Theorem 2.2 and ECT, we obtain the following thesis:

Thesis 1*: If some realistic intelligence system (say, a human brain with pencil and paper, or a future neural network) achieves AGI, then a single Transformer can also achieve AGI with at most a polynomial slowdown.*

Remark 3.2. There is ongoing debate as to whether quantum computers falsify ECT. In particular, it is a central problem in quantum computational complexity theory, well-known as the BQP =?BPP problem, whether all decision problems solvable by a polynomial-time quantum computer can also be solved by a polynomial-time PTM. If ECT is falsified by quantum computers, then a quantum variant of Transformer that can simulate universal quantum computers (Benioff, 1980; Deutsch, 1985; Yao, 1993; Bernstein & Vazirani, 1993) might be necessary to achieve AGI.

Remark 3.3. It is widely accepted that a human brain can be modeled as a complex computational system (say, a huge neural network) following classical physical laws, and thus can be simulated by a PTM (Searle, 1992; Guttenplan & Guttenplan, 1994). However, this traditional view of the brain as a classical system was challenged by Penrose (1994); Hameroff & Penrose (2014): they argued that the brain utilizes quantum mechanical effects (e.g., quantum coherence or entanglement) for reasoning and recognition, and human consciousness is even non-algorithmic, though still lack empirical validation.

3.2. Algorithmically Description of Human Reasoning

If we accept that (a) the Extended Church-Turing thesis applies to the human's reasoning process, meaning that the reasoning process of humans can be efficiently simulated by a PTM, and (b) a human brain, or a group of human brains (say, a research community) with paper and pencil can achieve AGI, then by Theorem 2.2, we should also accept that in principle a single Transformer or a group of Transformers can also achieve AGI as well. The related challenge lies in algorithmically describing the human reasoning process, including cognitive functions like intuition or creativity.

Question 2: *How to algorithmically describe human reasoning process?*

There are two kinds of general approaches to this challenge: connectionism and symbolicism.

1. Connectionism: Simulating the Brain at the Physical Level. Connectionism posits that human reasoning arises from the emergent properties of biologically inspired neural networks. By modeling the brain's physical and biological substrates—specifically, the interactions of neurons through synaptic connections—this approach seeks to replicate cognitive processes via distributed, parallel computation. Modern artificial neural networks, such as deep learning architectures, exemplify this paradigm. These systems learn hierarchical representations from data, mirroring how the brain processes sensory input and abstracts patterns (Hinton et al., 2006; LeCun et al., 2015).

For instance, Transformer architectures (Vaswani et al., 2017) model sequential reasoning by leveraging temporal dependencies and attention mechanisms, achieving expert-level performance in complex mathematical reasoning and code generation tasks (OpenAI, 2024; Guo et al., 2025). Connectionist models excel at pattern recognition and probabilistic reasoning but often lack explicit symbolic representations, leading to critiques about their interpretability and inability to handle structured, rule-based logic (Marcus, 2018). Recent advances in neuro-symbolic integration, however, aim to bridge this gap by combining neural networks with symbolic reasoning modules (Besold et al., 2021; Bhuyan et al., 2024).

2. Symbolicism: Abstracting General Principles of Human Thought. Symbolicism adopts a top-down perspective, seeking to formalize the universal principles and logical structures that underpin human reasoning. Rooted in classical AI and influenced by philosophy, linguistics, and formal logic, this approach abstracts cognition into discrete symbols and rules, independent of biological implementation. Unlike connectionism, which emulates neural substrates, symbolicism prioritizes computational-level explanations of thought—asking what problems cognition solves and why, rather than how the brain physically solves them (Pylyshyn, 1989; Newell & Simon, 2007).

At its core, symbolicism assumes that reasoning can be modeled as manipulation of explicit representations through deterministic or probabilistic rules. For example: Occam's razor, a heuristic for inductive reasoning, is formalized in algorithmic frameworks like Bayesian model selection (Jefferys & Berger, 1992), where simpler hypotheses are assigned higher prior probabilities. Deductive reasoning is captured by logic-based systems (e.g., Prolog, theorem provers) that apply syllogistic rules (e.g., modus ponens) to derive conclusions from premises (Russell & Norvig, 2016).

Here, we argue that an effective solution requires a combination of these two approaches, since (i) abstracting general principles offers a more tractable and generalizable framework for intelligence and (ii) part of knowledge and functions, such as pattern recognition and cognitive functions, may have no representation more concise than a huge, analog neural network (Graham, 2007), thus are not suitable to be represented as logic or symbolic.

In particular, since Transformers can effectively integrate knowledge and functions represented in network form (since they are neural networks) with logical reasoning abilities (Theorem 2.2), and thus can leverage benefits from both connectionism and symbolicism, we suggest that Transformers are particularly well-suited as approximations of human intelligence.

4. Algorithmic approximations of general intelligence

Besides mimicking the human reasoning process, another line of research, inspired by algorithmic information theory (Li et al., 2008), aims to achieve or even surpass human-level intelligence by establishing a formal theory of general intelligence, such as Levin's universal search algorithm (Levin, 1973; 1984), Solomonoff's universal induction (Solomonoff, 1964; 1978), and Hutter's AIXI agent (Hutter, 2005). While these constructions are theoretically optimal, they are often intractable in practice and even uncomputable. However, we argue that Transformers provide a promising and tractable approximation of these universal constructions.

4.1. Levin's Universal Search

Many deductive reasoning tasks, such as theorem proving, planning, and general NP-complete problems, can be effectively modeled as search problems.

Search problems. Let $\phi : \{0,1\}^* \to \{0,1\}^*$ be a function where $\phi(\cdot)$ can be computed quickly (say, in polynomial time). The search problem is defined as: given y, find an x such that $\phi(x) = y$.

For example, in the Boolean satisfiability problem (SAT), the function $\phi : \{0,1\}^* \to \{0,1\}$ can be defined as a verifier that checks whether a given assignment satisfies the Boolean formula in conjunctive normal form.

Levin search. The algorithm is simple to describe: just run and verify the output of all algorithms p in parallel with relative computation time $2^{-\ell(p)}$; i.e. a time fraction $2^{-\ell(p)}$ is devoted to executing p (Levin, 1973; 1984). Here, programs are described as Boolean strings in a prefix-free encoding, where $\ell(p)$ denotes the length of the description of p. Note that $\sum_p 2^{-\ell(p)} \leq 1$.

Theorem 4.1 (Levin (1973; 1984); Hutter (2005))**.** *The computation time of Levin search is upper bounded by* $\min_p\{2^{\ell(p)} \cdot \mathrm{time}^+_p(y)\}$*, where* $\mathrm{time}^+_p(y)$ *is the runtime of* $p(y)$ *plus the time to verify the correctness of the result (*$\phi(x) = y$*) by a known implementation for* ϕ*.*

By Theorem 2.2, we conclude that in principle, a single Transformer can solve all search problems as quickly as the fastest algorithm for each, up to a constant factor.

We note that Levin's universal search—which optimally allocates computational effort across candidate solvers according to their algorithmic probability (Theorem 4.1)—may provide a theoretical foundation for the emerging paradigm of inference-time scaling in LLMs (Brown et al., 2024; Snell et al., 2024; OpenAI, 2024; Guo et al., 2025). This framework structures LLM reasoning into two synergistic phases: generating diverse candidate solutions (or algorithms) and efficiently prioritizing their execution and evaluation, mirroring Levin's time-optimal balance between exploration and exploitation. By prescribing a focus on programs with minimal description length (i.e., favoring simpler, valid solutions), Levin's principles offer guidance for designing compute-efficient strategies.

Though Levin search is theoretically optimal for all search problems, the large constant overhead $2^{\ell(p)}$ renders it impractical. A line of research (Solomonoff, 1986; Schmidhuber et al., 1997; Schmidhuber, 1997; 2002b; 2004) has explored adaptations of Levin's search that leverage past experience to improve its efficiency. We note that the key lies in generating highly successful algorithms p with the shortest description length, ensuring they are prioritized during the search process. Such knowledge can be acquired from experience. For instance, a Transformer could maintain a parameterized model (e.g., a neural network or program) within its context and employ bootstrap methods—such as search-and-learn processes (Arfaee et al., 2011)—to iteratively refine its performance. By repeatedly solving increasingly challenging instances and updating the model based on successfully solved examples, the system could incrementally improve its problem-solving efficiency.

4.2. Solomonoff's Universal Induction

Every inductive reasoning task, such as continuing a number of series in an IQ test, classification in machine learning, stock-market forecasting, or scientific research, can be described as a sequence prediction problem, more precisely, predicting future data from past observations (Hutter, 2005). Without loss of generality and for simplicity, we assume the data $x_i \in \{0, 1\}$ is binary.

We first introduce some notations and definitions. Given a subset S of $\{0, 1\}^\star$, let $\lfloor S \rfloor$ denote the set obtained from S by deleting all elements that have a prefix in S. A *monotone Turing machine* is a Turing machine with one unidirectional input tape, one unidirectional output tape, and some bidirectional work tapes. The input tape is read-only, and the output tape is write-only. We say a tape is unidirectional if its head can only move from left to right, and bidirectional if its head can move in both directions.

Definition 4.2 (Measure)**.** We say a function $\mu : \{0, 1\}^* \to [0, 1]$ is a measure if $\mu(\emptyset) = 1$ and $\mu(x) = \mu(x1) + \mu(x0)$. Here, $\emptyset$ denotes the empty string.

A measure μ defines a random process generating an infinitely long binary sequence: start with an empty string and repeatedly select the next bit $x_n \in \{0, 1\}$ according to the probability $\mu(x_n \mid x_{<n}) := \mu(x_{<n}x_n)/\mu(x_{<n})$ conditioned on the past data $x_{<n} := x_1x_2 \cdots x_n$.

We say μ is estimable if there exists a TM that, given $x \in \{0, 1\}^*$ and a precision ϵ, computes an ϵ-approximation of $\mu(x)$. By the physical version of the Church-Turing thesis, any μ implemented on a finite, realistic physical device is estimable. Moreover, by Theorem 4.5.2 in (Li et al., 2008), for any estimable μ, there is a monotone TM T that takes an infinitely long uniformly random binary string as input and generates an infinitely long binary sequence according to μ. Let $K(\mu)$ denote the shortest description of such a T.

Sequence prediction problem. Having observed the past data $x_{<n} := x_1x_2 \cdots x_{n-1}$, the task is to predict the next bit x_n. More precisely, let μ denote the unknown underlying mechanism generating the sequence $x_1x_2 \cdots$. The task is to estimate the conditional probability $\mu(x_n \mid x_{<n}) := \mu(x_{<n}x_n)/\mu(x_{<n})$.

Solomonoff's universal induction. Bayesian prediction provides a framework for sequence prediction problems, which repeatedly employs Bayes' rule to update its beliefs about each hypothesis based on newly observed data. The primary challenge is how to select the prior beliefs. Solomonoff (Solomonoff, 1964; 1978) addressed this challenge by introducing a universal prior, rooted in the simplicity of hypotheses. His approach leverages the fact that simpler hypotheses, represented by shorter programs, are more likely to generalize well—a concept aligned with Occam's Razor. Solomonoff showed that the Bayesian prediction with the Solomonoff prior as the prior belief is an optimal way for the sequence prediction problem, provided that the underlying μ is estimable.

Definition 4.3 (Solomonoff prior (Solomonoff, 1964; 1978))**.** Let U be a monotone UTM. The Solomonoff prior is defined as

$$M_U(x) := \sum_{\lfloor p \in \{0,1\}^* : U(p) = x\star \rfloor} 2^{-\ell(p)}.$$

Here, $U(p) = x\star$ means x is a prefix of $U(p)$. Intuitively, $M_U(x)$ is the probability that the output starts with x when the input is an infinite-long uniformly random binary string.

Remark 4.4. For different monotone UTMs U_1 and U_2, the associated Solomonoff priors are equivalent up to multiplicative constants: there exist two constants $0 < c_1 < c_2$ such that $c_1 \cdot M_{U_1}(x) \le M_{U_2}(x) \le c_2 \cdot M_{U_2}(x)$ for any $x \in \{0,1\}^*$ (Solomonoff, 1978; Wood et al., 2013).

Solomonoff's universal induction is simple to describe: use $M_U(x_n \mid x_{<n}) = M_U(x_n)/M_U(x_{<n})$ as an estimate of the true conditional probability $\mu(x_n \mid x_{<n})$.

Theorem 4.5 (Solomonoff central theorem (Solomonoff, 1964; 1978))**.** *For any estimable μ, we have*

$$\sum_{n=1}^{+\infty} \sum_{x_n \in \{0,1\}} \mu(x_{<n}) \left(M(x_t \mid x_{<t}) - \mu(x_n \mid x_{<n})\right)^2 \le \ln 2 \cdot K(\mu) + O(1).$$

For any estimable μ, the upper bound $\ln 2 \cdot K(\mu)$ is finite, so the difference $M(x_t \mid x_{<t}) - \mu(x_n \mid x_{<n})$ tends to zero as $n \to \infty$ with μ-probability 1. Consequently, $M(x_t \mid x_{<t})$ converges rapidly to the true underlying generating process.

Unfortunately, Solomonoff prior $M_U(x)$ is inestimable: there is no TM that, given $x \in \{0,1\}^*$ and a precision ϵ, can compute an ϵ-approximation of $M_U(x)$ in finite time. To address this uncomputability issue, several approximations have been proposed (Schmidhuber, 2002a; Veness et al., 2012; Filan et al., 2016; Grau-Moya et al., 2024).

In particular, observing that Transformers are naturally suited for sequence prediction tasks, a line of work (Müller et al., 2022; Hollmann et al., 2023; Grau-Moya et al., 2024; Goldblum et al., 2024; Young & Witbrock, 2024) has explored whether the Transformer model can approximate Solomonoff induction. Specifically, Hollmann et al. (2023); Müller et al. (2022) showed that transformers can do Bayesian inference. Grau-Moya et al. (2024) used Transformers to approximate Solomonoff induction by training on UTM data, and showed that increasing model size leads to improved performance, demonstrating that model scaling helps learning increasingly universal prediction strategies. Young & Witbrock (2024) proposed and investigated the hypothesis that Transformers approximate Solomonoff induction better than any other extant sequence prediction method. This hypothesis was further supported by (Goldblum et al., 2024; Delétang et al., 2024). Specifically, they showed that like Solomonoff induction, transformers also align with Occam's Razor: transformers prefer generating data with low Kolmogorov complexity. Occam's razor provides transformers with good generalization on many different problems and modalities of data, and makes them powerful general-purpose predictors.

4.3. Hutter's AIXI agent

In this subsection, we briefly introduce Hutter's AIXI agent, which is claimed to be universal in that it is independent of the true environment (model-free) and is able to solve any solvable problem and learn any learnable task. The main idea of AIXI is simple to describe: just replace the unknown environmental distribution in the Bellman equations with a suitably generalized Solomonoff prior (Hutter, 2005).

Setting. The agent and the environment interact chronologically as follows: in each cycle k, the agent performs an action $y_k \in \mathcal{Y}$ (output), and then receives a perception $x_k \in \mathcal{X}$ from the environment. The perception x_k consists of a regular part o_k and a reward r_k. Given the history $y_1 x_1 \cdots x_{k-1} y_k$, the probability that the environment produces perception x_k is denoted $\mu(x_k \mid y_1 x_1 \cdots x_{k-1} y_k)$. Here, we make no assumptions about μ other than it is estimable. In particular, μ is allowed to depend on the complete history $y_1 x_1 \cdots x_{k-1} y_k$.

We use p to denote the agent's policy, which can be described as a monotone Turing machine that takes $x_1 x_2 \cdots$ as input and outputs $y_1 y_2 \cdots$. As the optimal policy can always be chosen to be deterministic, we assume p is a deterministic monotone TM. In addition, we say $y_{1:k} = p(x_{<k})$ if $y_i = p(y_1 x_1 y_2 x_2 \cdots x_{i-1})$ for $i \le k$. We also use $\mu(x_{k:m} \mid y_{1:m} x_{<k})$ as an abbreviation for $\Pi_{i=k}^m \mu(x_i \mid x_{<i}, y_{\le i})$. We define the value of policy p in environment μ as

$$V_\mu^p := \sum_{x_{1:m}} (r_1 + \cdots + r_m) \mu(x_{1:m} \mid y_{1:m})_{|y_{1:m} = p(x_{<m})}$$

where m is the lifespan of the agent.

The goal of the agent is to maximize the total reward $\sum_{i=1}^m r_i$. Formally, the agent aims to find a policy p^μ that maximizes V_μ^p.

The AIXI agent. If the environment μ is known, then the optimal policy is

$$y_k := \arg\max_{y_k} \sum_{x_k} \cdots \max_{y_m} \sum_{x_m} \left(\sum_{i=k}^m r_i \right) \mu(x_{k:m} \mid y_{1:m}, x_{<k})$$

with total reward

$$\max_{y_1} \sum_{x_1} \cdots \max_{y_m} \sum_{x_m} (r_1 + \cdots + r_m) \mu(x_{1:m} \mid y_{1:m}) := V_\mu^*$$

The AIXI agent replaces the true but unknown μ with a generalized Solomonoff prior. Specifically, the AIXI policy is

$$y_k := \arg\max_{y_k} \sum_{x_k} \cdots \max_{y_m} \sum_{x_m} \left(\sum_{i=k}^{m} r_i \right) \xi(x_{k:m} \mid y_{1:m}, x_{<k})$$

where

$$\xi(x_{1:k} \mid y_{1:k}) := \sum_{\text{monotone TM } q: q(y_{1:k}) = x_{1:k}} 2^{-\ell(q)}.$$

Intuitively, the agent continually updates its belief about hypotheses of the unknown environment μ by Bayes' rule. Similar to Solomonoff universal induction, environments with lower Kolmogorov complexity are preferred, in line with Occam's razor. Hutter (2005) shows that AIXI's environment model converges rapidly to the true environment, and its policy is Pareto-optimal and self-optimizing. Here, we say a policy Pareto-optimal if there is no other agent that performs at least as well as AIXI in all environments while performing strictly better in at least one environment, and self-optimizing if $\frac{1}{m} V_\mu^{\text{AIXI}} \to \frac{1}{m} V_\mu^*$ for horizon $m \to +\infty$ for all estimable μ.

Unfortunately, like Solomonoff's universal induction, AIXI is uncomputable. To address this issue, several computable approximations have been proposed (Hutter, 2005; Pankov; Veness et al., 2010; 2011; 2012; 2013; Bellemare et al., 2013; 2014; Yang-Zhao et al., 2022; 2024). One such approximation is AIXI$t\ell$, which performs at least as well as any other agent bounded time t and length ℓ. Some approximations focus on restricted environment classes and have been successfully implemented (Veness et al., 2011). Yang-Zhao et al. (2024) studied how to inject knowledge into the AIXI agent.

We suggest that the Transformer model has the potential to approximate AIXI for the following reasons: (i) as previously discussed, Transformers might serve as a good approximation of Solomonoff induction, and provide a good estimation of ξ; (ii) with an estimation of ξ, Transformers can solve the Bellman equation using an enhanced Levin search or an enhanced $M_{p^*}^{\epsilon}$ algorithm (Hutter, 2005), which solves all well-defined problems as quickly as the fastest algorithm for each problem; and (iii) Transformers effectively integrate prior knowledge, leveraging human experience to further enhance their practical applicability.

5. Alternative Views

This section discusses alternative views arguing that Transformers are not a sufficient path to AGI.

Alternative view 1: Transformers miss essential capabilities for intelligent beings, such as understanding and reasoning about the physical world. Specifically, Transformers cannot anchor their understanding in reality: They cannot perform actions in the real world or learn through embodied experiences, and they lack the capability for hierarchical planning, a crucial element for understanding and interacting with the world at multiple levels of abstraction (e.g. (Lecun, 2024)).

We acknowledge that current Transformers lack the capabilities to interact with the physical world directly and employ embodied learning. However, we do not think this represents an inherent limitation of Transformers. With minor enhancements, Transformers could be embedded within agent models. Specifically, such agents could utilize Transformers as an approximation of universal induction (Section 4.2) to learn about the unknown environment, and subsequently apply them as an approximation of universal search (Section 4.1) to perform deductive reasoning.

Besides, we argue that current Transformers really do understand. In our definition of intelligence (Section 1), understanding can be equated to inductive reasoning–the ability to uncover the underlying general mechanism from specific observations. As we argued in Section 4.2, Transformers provide a promising practical approximation of Solomonoff's universal induction, which is a universal and optimal way to do inductive reasoning.

In addition, as we argued in Section 2, Transformers can execute any meta-process, such as algorithm design, when an algorithmic description is provided. In particular, as argued in Section 4.1, Transformers provide a promising practical approximation of Levin's universal search algorithm, enabling them to efficiently perform various deductive reasoning tasks, including planning and theorem proof.

Alternative view 2: Transformers are limited by their expenditure of bounded compute per input instance, e.g. the finite context window and finite precision, thus cannot simulate a UTM, whose tapes are infinitely long (e.g. (Lecun, 2024; Goldblum et al., 2024; Upadhyay & Ginsberg, 2023)).

First, no finite physical system, such as a human brain or a personal computer, can solve problems of infinite size. This limitation naturally extends to the simulation of a UTM, which assumes infinitely long tapes. Therefore, when discussing whether a Transformer can simulate a UTM, the correct interpretation should follow the framework of the logical circuit model (Arora & Barak, 2009), specifically: "a uniform family of Transformers can simulate a UTM." In other words, for any arbitrarily large tape length ℓ, there exists an efficiently constructible Transformer capable of simulating a UTM with tape length ℓ.

In this context, while an exact simulation of a UTM is impossible, a sufficiently large Transformer can approximate its behavior to an arbitrarily high degree of accuracy. This scalability ensures that Transformers, much like circuits, can address increasingly complex problems within practical computational limits.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Aharonov, D. and Vazirani, U. V. *Is Quantum Mechanics Falsifiable? A Computational Perspective on the Foundations of Quantum Mechanics*, pp. 329–349. 2013.

AlphaProof, D. and Teams, A. AI achieves silver-medal standard solving international mathematical olympiad problems, 2024. URL https://deepmind.google/discover/blog/ai-solves-imo-problems-at-silver-medal-level.

Anthropic. The claude 3 model family: Opus, sonnet, haiku, 2024.

Arfaee, S. J., Zilles, S., and Holte, R. C. Learning heuristic functions for large state spaces. *Artificial Intelligence*, 175(16-17):2075–2098, 2011.

Arora, S. and Barak, B. *Computational complexity: a modern approach*. Cambridge University Press, 2009.

Bellemare, M. G., Veness, J., and Bowling, M. Bayesian learning of recursively factored environments. In *International Conference on Machine Learning*, volume 28, pp. 1211–1219, 2013.

Bellemare, M. G., Veness, J., and Talvitie, E. Skip context tree switching. In *ICML*, volume 32, pp. 1458–1466, 2014.

Benioff, P. The computer as a physical system: A microscopic quantum mechanical hamiltonian model of computers as represented by turing machines. *Journal of Statistical Physics*, 22:563–591, 1980.

Bernstein, E. S. and Vazirani, U. V. Quantum complexity theory. *ACM Symposium on Theory of Computing*, 1993.

Besold, T. R., d'Avila Garcez, A., Bader, S., Bowman, H., Domingos, P., Hitzler, P., Kühnberger, K.-U., Lamb, L. C., Lima, P. M. V., de Penning, L., et al. Neural-symbolic learning and reasoning: A survey and interpretation 1. In *Neuro-Symbolic Artificial Intelligence: The State of the Art*, pp. 1–51. IOS press, 2021.

Bhattamishra, S., Patel, A., and Goyal, N. On the computational power of transformers and its implications in sequence modeling. In *Conference on Computational Natural Language Learning*, pp. 455–475, 2020.

Bhuyan, B. P., Ramdane-Cherif, A., Tomar, R., and Singh, T. Neuro-symbolic artificial intelligence: a survey. *Neural Computing and Applications*, pp. 1–36, 2024.

Brown, B., Juravsky, J., Ehrlich, R., Clark, R., Le, Q. V., Ré, C., and Mirhoseini, A. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*, 2024.

Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., Abbeel, P., Srinivas, A., and Mordatch, I. Decision transformer: Reinforcement learning via sequence modeling. *Advances in Neural Information Processing Systems*, 34:15084–15097, 2021.

Church, A. An unsolvable problem of elementary number theory. *American Journal of Mathematics*, 58:345, 1936.

Copeland, B. J. and Shagrir, O. The Church-Turing thesis: logical limit or breachable barrier? *Commun. ACM*, 62 (1):66–74, 2018.

Delétang, G., Ruoss, A., Duquenne, P., Catt, E., Genewein, T., Mattern, C., Grau-Moya, J., Wenliang, L. K., Aitchison, M., Orseau, L., Hutter, M., and Veness, J. Language modeling is compression. In *International Conference on Learning Representations*, 2024.

Deutsch, D. Quantum theory, the Church-Turing principle and the universal quantum computer. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, 400:117 – 97, 1985.

Dosovitskiy, A. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Yang, A., Fan, A., et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Filan, D., Leike, J., and Hutter, M. Loss bounds and time complexity for speed priors. In *International Conference on Artificial Intelligence and Statistics*, volume 51, pp. 1394–1402, 2016.

Gemini, T., Anil, R., Borgeaud, S., Alayrac, J.-B., Yu, J., Soricut, R., Schalkwyk, J., Dai, A. M., Hauth, A., Millican, K., et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Giannou, A., Rajput, S., Sohn, J.-y., Lee, K., Lee, J. D., and Papailiopoulos, D. Looped transformers as programmable computers. In *International Conference on Machine Learning*, pp. 11398–11442, 2023.

Goldblum, M., Finzi, M. A., Rowan, K., and Wilson, A. G. Position: The no free lunch theorem, kolmogorov complexity, and the role of inductive biases in machine learning. In *International Conference on Machine Learning*, 2024.

Graham, P. How to do philosophy. http://www.paulgraham.com/philosophy.html, 2007.

Grau-Moya, J., Genewein, T., Hutter, M., Orseau, L., Delétang, G., Catt, E., Ruoss, A., Wenliang, L. K., Mattern, C., Aitchison, M., and Veness, J. Learning universal predictors. In *International Conference on Machine Learning*, 2024.

Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Guttenplan, S. and Guttenplan, S. D. *A Companion to the Philosophy of Mind*. Blackwell Oxford, 1994.

Hadfield-Menell, D., Milli, S., Abbeel, P., Russell, S. J., and Dragan, A. Inverse reward design. *Advances in neural information processing systems*, 30, 2017.

Hameroff, S. and Penrose, R. Consciousness in the universe: A review of the 'orch or'theory. *Physics of life reviews*, 11(1):39–78, 2014.

Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., and Steinhardt, J. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Hennie, F. C. and Stearns, R. E. Two-tape simulation of multitape turing machines. *Journal of the ACM*, 13(4): 533–546, 1966.

Hinton, G. E., Osindero, S., and Teh, Y.-W. A fast learning algorithm for deep belief nets. *Neural computation*, 18 (7):1527–1554, 2006.

Hollmann, N., Müller, S., Eggensperger, K., and Hutter, F. Tabpfn: A transformer that solves small tabular classification problems in a second. In *International Conference on Learning Representations*, 2023.

Hou, K., Brandfonbrener, D., Kakade, S. M., Jelassi, S., and Malach, E. Universal length generalization with turing programs. *CoRR*, abs/2407.03310, 2024.

Hutter, M. *Universal artificial intelligence: Sequential decisions based on algorithmic probability*. Springer Science & Business Media, 2005.

Jefferys, W. H. and Berger, J. O. Ockham's razor and bayesian analysis. *American scientist*, 80(1):64–72, 1992.

Jimenez, C. E., Yang, J., Wettig, A., Yao, S., Pei, K., Press, O., and Narasimhan, K. Swe-bench: Can language models resolve real-world github issues? *arXiv preprint arXiv:2310.06770*, 2023.

Lecun, Y. Meta ai, open source, limits of llms, agi & the future of ai, 2024. https://www.youtube.com/watch?v=5t1vTLU7s40.

LeCun, Y., Bengio, Y., and Hinton, G. Deep learning. *nature*, 521(7553):436–444, 2015.

Levin, L. A. Universal sequential search problems. *Problemy peredachi informatsii*, 9(3):115–116, 1973.

Levin, L. A. Randomness conservation inequalities; information and independence in mathematical theories. *Information and Control*, 61(1):15–37, 1984.

Li, M., Vitányi, P., et al. *An introduction to Kolmogorov complexity and its applications*, volume 3. Springer, 2008.

Liu, Z., Liu, H., Zhou, D., and Ma, T. Chain of thought empowers transformers to solve inherently serial problems. In *International Conference on Learning Representations*, 2024.

Marcus, G. Deep learning: A critical appraisal. *arXiv preprint arXiv:1801.00631*, 2018.

Merrill, W. and Sabharwal, A. The expressive power of transformers with chain of thought. In *International Conference on Learning Representations*, 2024.

Müller, S., Hollmann, N., Pineda-Arango, S., Grabocka, J., and Hutter, F. Transformers can do bayesian inference. In *International Conference on Learning Representations*, 2022.

Newell, A. and Simon, H. A. Computer science as empirical inquiry: Symbols and search. In *ACM Turing award lectures*, pp. 1975. 2007.

Ng, A. Y., Russell, S., et al. Algorithms for inverse reinforcement learning. In *International Conference on Machine Learning*, volume 1, pp. 2, 2000.

of Philosophy, S. E. The Church-Turing thesis. https://plato.stanford.edu/entries/church-turing/, 2023.

OpenAI. Learning to reason with LLMs. OpenAI Blog, Feb 2024. https://openai.com/index/learning-to-reason-with-llms.

Pankov, S. A computational approximation to the AIXI model. In *Artificial General Intelligence*, volume 171, pp. 256–267.

Penrose, R. *Shadows of the Mind*, volume 4. Oxford University Press Oxford, 1994.

Pérez, J., Marinkovic, J., and Barceló, P. On the turing completeness of modern neural network architectures. In *International Conference on Learning Representations*, 2019.

Pérez, J., Barceló, P., and Marinkovic, J. Attention is turing-complete. *Journal of Machine Learning Research*, 22 (75):1–35, 2021.

Pippenger, N. and Fischer, M. J. Relations among complexity measures. *Journal of the ACM*, 26(2):361–381, 1979.

Pylyshyn, Z. W. Computing in cognitive science. *Foundations of cognitive science*, pp. 51–91, 1989.

Qiu, R., Xu, Z., Bao, W., and Tong, H. Ask, and it shall be given: Turing completeness of prompting. *CoRR*, abs/2411.01992, 2024.

Rein, D., Hou, B. L., Stickland, A. C., Petty, J., Pang, R. Y., Dirani, J., Michael, J., and Bowman, S. R. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

Russell, S. J. and Norvig, P. *Artificial intelligence: a modern approach*. Pearson, 2016.

Schmidhuber, J. Discovering neural nets with low kolmogorov complexity and high generalization capability. *Neural Networks*, 10(5):857–873, 1997.

Schmidhuber, J. The speed prior: A new simplicity measure yielding near-optimal computable predictions. In *Annual Conference on Computational Learning Theory*, volume 2375, pp. 216–228, 2002a.

Schmidhuber, J. Bias-optimal incremental problem solving. *Advances in Neural Information Processing Systems*, 15, 2002b.

Schmidhuber, J. Optimal ordered problem solver. *Machine Learning*, 54:211–254, 2004.

Schmidhuber, J., Zhao, J., and Wiering, M. Shifting inductive bias with success-story algorithm, adaptive levin search, and incremental self-improvement. *Machine Learning*, 28:105–130, 1997.

Schnorr, C.-P. The network complexity and the turing machine complexity of finite functions. *Acta Informatica*, 7: 95–107, 1976.

Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., Guez, A., Lockhart, E., Hassabis, D., Graepel, T., et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839): 604–609, 2020.

Schuurmans, D. Memory augmented large language models are computationally universal. *arXiv preprint arXiv:2301.04589*, 2023.

Searle, J. R. The rediscovery of the mind. *A Bradford Book*, 1992.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

Silver, D., Singh, S., Precup, D., and Sutton, R. S. Reward is enough. *Artificial Intelligence*, 299:103535, 2021.

Snell, C., Lee, J., Xu, K., and Kumar, A. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

Solomonoff, R. A formal theory of inductive inference. part i and ii. *Information and Control*, 7(1):1–22, 1964.

Solomonoff, R. Complexity-based induction systems: comparisons and convergence theorems. *IEEE Transactions on Information Theory*, 24(4):422–432, 1978.

Solomonoff, R. The application of algorithmic probability to problems in artificial intelligence. In *Machine Intelligence and Pattern Recognition*, volume 4, pp. 473–491. 1986.

Turing, A. M. Computability and λ-definability. *The Journal of Symbolic Logic*, 2(4):153–163, 1937.

Upadhyay, S. K. and Ginsberg, E. J. Turing complete transformers: Two transformers are more powerful than one. 2023.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.

Veness, J., Ng, K. S., Hutter, M., and Silver, D. Reinforcement learning via AIXI approximation. In *Proceedings National Conference on Artificial Intelligence*, pp. 605–611, 2010.

Veness, J., Ng, K. S., Hutter, M., Uther, W. T. B., and Silver, D. A monte-carlo AIXI approximation. *J. Artif. Intell. Res.*, 40:95–142, 2011.

Veness, J., Sunehag, P., and Hutter, M. On ensemble techniques for AIXI approximation. In *Artificial General Intelligence*, volume 7716, pp. 341–351, 2012.

Veness, J., White, M., Bowling, M., and György, A. Partition tree weighting. In *Data Compression Conference*, pp. 321–330, 2013.

Wolfram, S. Undecidability and intractability in theoretical physics. *Physical Review Letters*, 54 8:735–738, 1985.

Wood, I., Sunehag, P., and Hutter, M. (non-) equivalence of universal priors. In *Algorithmic Probability and Friends*, pp. 417–425. Springer, 2013.

Yang-Zhao, S., Wang, T., and Ng, K. S. A direct approximation of AIXI using logical state abstractions. In *Advances in Neural Information Processing Systems*, 2022.

Yang-Zhao, S., Ng, K. S., and Hutter, M. Dynamic knowledge injection for AIXI agents. In *Proceedings National Conference on Artificial Intelligence*, pp. 16388–16397, 2024.

Yao, A. C. Classical physics and the Church-Turing thesis. *Journal of the ACM*, 50(1):100–105, 2003.

Yao, A. C.-C. Quantum circuit complexity. In *IEEE Annual Symposium on Foundations of Computer Science*, 1993.

Young, N. and Witbrock, M. Transformers as approximations of solomonoff induction. *CoRR*, abs/2408.12065, 2024.

A. Background on Turing Machines

A.1. Background on Turing Machines

Turing machines. Turing machines (TMs) are a mathematical model of computation. A k-tape TM is defined as a tuple $\langle \Sigma, \perp, Q, q_{start}, F, \delta \rangle$ where (i) Σ is a finite tape alphabet including a blank symbol $\perp$, (ii) Q is the finite set of states containing initial state q_{start}, (iii) $F \subseteq Q$ is a set of halting states, and (iv) δ is a transition function $(Q \setminus F) \times \Sigma^k \to Q \times (\Sigma \times \{L, S, R\})^k$.

Throughout this paper, we will assume $\Sigma = \{0, 1, \perp\}$ for simplicity and with loss of generality.

Probabilistic Turing Machines. A probabilistic Turing Machine (PTM) is a Turing machine with an additional read-only coin tape full of independent and uniformly random coins.

The PTM model is potentially more powerful than the deterministic Turing machine (DTM) model. An example of a computational problem that can be solved in polynomial time by a PTM but still not known how by a DTM is the polynomial identity testing problem (PIT) (see, e.g. (Arora & Barak, 2009)). In fact, it is a central question in complexity theory, well-known as the $\mathsf{BPP} =? \mathsf{P}$ problem, whether any decision problem solvable by a polynomial-time PTM can also be solved by a polynomial-time DTM.

We say that a (deterministic or probabilistic) TM T is *oblivious* if the tape head movements of T running on input x depend only on the input length $|x|$. That is, T makes the same sequence of head movements for all inputs x of the same length. Hennie & Stearns (1966); Pippenger & Fischer (1979) proved that: for every multitape DTM T running in $O(t(n))$ time, there is an equivalent oblivious two-tape DTM T' that runs in $O(t(n) \log t(n))$ time. Furthermore, as observed by Schnorr (1976), this result also holds for all relative Turing machines, including PTMs. Specifically, for any multitape PTM T running in $O(t(n))$ time, there is an equivalent oblivious two-tape PTM T' running in $O(t(n) \log t(n))$ with an additional ready-only coin tape. So, w.l.o.g., in this paper unless otherwise specified, whenever we refer to DTMs or PTMs, we refer to two-tape oblivious DTMs or PTMs respectively.

Universal Turing Machines. A universal Turing machine (UTM) is a TM that can simulate the execution of every other (deterministic or probabilistic) TM T given T's description as input. Specifically, we encode PTMs as Boolean strings in a prefix-free way. A UTM is a PTM U that takes the concatenations of the encoding of a PTM T and an input x, and outputs the (possibly randomized) $T(x)$. UTMs capture the notion of a "general-purpose programmable computer", which is a single machine that can be adapted to any arbitrary task provided an appropriate program is loaded. We remark that the parameters of a UTM, such as alphabet size, number of states, and number of tapes are fixed, though the TM being simulated could have much more parameters.