Supplementary Materials

Anonymous Author(s)
Affiliation
Address
email

Appendices

A Overview and Further Background

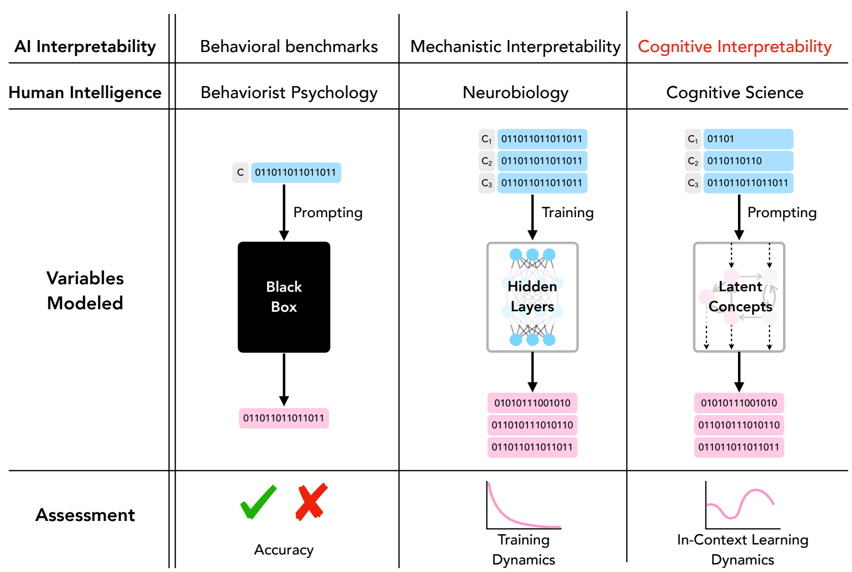

Figure 1: **Cognitive Interpretability in the context of prior work** Cognitive Interpretability studies in-context learning dynamics in LLMs, positing theories of the latent concepts enabling behavioral capabilities. It is a middle ground between behavioral benchmarks, which treat models as black boxes and evaluate hit-or-miss accuracy, and mechanistic interpretability, which studies toy models and training loss dynamics, analogous to how cognitive science is a middle ground between behaviorist psychology and neurobiology in the study of human intelligence.

A multi-layer neural networks may possess multiple distributed circuits implementing computational primitives, such as basic mathematical operations like addition and sequence copying [1–6]. In state of the art LLMs with hundreds of billions (even trillions) of parameters, there may be sub-networks implementing various computations, and a wide variety of emergent behaviors corresponding to those computations. A similar situation occurs in research focused on understanding the human brain, where sub-networks of neuronal cells have been shown to localize specific capabilities. In cognitive science, researchers study such aspects of cognition without fully understanding the underlying neural circuitry, often modeling behavior without observing brain activity. Analogously, we seek to understand the structure of behaviors in large language models in the wild, without a full mechanistic understanding of their circuitry (Figure 1).

The reliance on benchmarks to evaluate LLM capabilities loosely parallels early behaviorist psychology, when theories of human and animal learning only assumed stimulus-response associations, without positing theories about mental processes or neural substrates [7]. Circuit-level *mechanistic* interpretations of neural networks parallel neurobiology, which offers physical models of information processing in biological neurons, but typically does not account for the structure of high-level behavior. Cognitive interpretability, like cognitive science, is aimed at predicting behavioral outcomes over a potentially infinite space of possible tasks. It defines high-level specifications of behaviors performed by LLMs, which, we argue, should be the first step before mechanistic understanding of circuits existent in a model is pursued.

Cognitive scientists have used Bayesian predictive and posterior distributions to model learning dynamics in human adults and children, as well as in non-human animals [8–10]. A discrete hypothesis space in a probabilistic model can give clear and meaningful explanations to learning patterns analogous to mode collapse and phase transitions in deep learning. When behavior suddenly shifts from one pattern, or mode, of behavior to another, this can be understood as one hypothesis coming to dominate the posterior $p(h|x)$ as x grows in scale [11]. Such cognitive analysis of behavioral shifts as generated from a shift in posterior probability between one hypothesis to another parallels recent work on learning dynamics in training and prompting LLMs. E.g., sharp phase changes have been observed when training neural networks, corresponding to the formation of identifiable mechanisms such as modular addition circuits or induction heads [1, 2, 4, 12]. ICL research has explored how few-shot prompting can significantly boost LLM performance in various domains, and how the particular exemplars provided in context determine its overall effectiveness [13–19]. For further discussion of related work, see Appendix C.

Learning Dynamics in Model Selection

Cognitive scientists have used Bayesian predictive and posterior distributions to model learning dynamics in human adults and children, as well as in non-human animals [8–10]. A discrete hypothesis space in a probabilistic model can give clear and meaningful explanations to learning patterns analogous to mode collapse and phase transitions in deep learning. When behavior suddenly shifts from one pattern, or mode, of behavior to another, this can be understood as one hypothesis coming to dominate the posterior $p(h|x)$ as x grows in scale [11]. For example, children between the ages of 3.5—5 years old learning to count undergo a dramatic conceptual shift from knowing the meanings of only a few number words ("one", "two") to a full inductive understanding of counting, which can be modeled as Bayesian model selection with a simplicity prior over models [20]. Such cognitive analysis of behavioral shifts as generated from a shift in posterior probability between one hypothesis to another parallels recent work on learning dynamics in training and prompting LLMs. E.g., sharp phase changes have been observed when training neural networks, corresponding to the formation of identifiable mechanisms such as modular addition circuits or induction heads [1, 2, 4, 12]. ICL research has explored how few-shot prompting can significantly boost LLM performance in various domains, and how the particular exemplars provided in context determine its overall effectiveness [13–19]. Work on chain-of-thought reasoning in LLMs demonstrates how a few exemplars of detailed solutions or even a simple prompt like "let's think this through step-by-step" can dramatically impact model performance [21–23]. For further discussion of related work, see Appendix C.

B Additional Experimental Details

All calls were made with the OpenAI API, using default parameters including — important to our analysis — a temperature parameter of 1.0. We use token-wise log probabilities $p(y_t|y_{0...t-1})$ from the OpenAI API where available for cost efficiency and since this is equivalent to drawing repeated token samples and computing the fraction of samples, e.g. $N_{\texttt{Tails}}/(N_{\texttt{Heads}} + N_{\texttt{Tails}})$.

In the Generation task, the context x includes the prompt question, as well as an initial set of coin flips that follow the beginning of the "answer" section of the context. Prompt context in these experiments includes a specific probability, shown in Fig. 2, where the last __ marks where the model begins generating tokens y. In subjective randomness experiments, an initial flip '`Heads`' is used

Q: Generate a sequence of 1000 random samples from a weighted coin, with {1 - p}% probability of Heads and {p}% probability of Tails.

A: [{sequence}, ________

Q: Is the following sequence of coin flips generated by a random process with no pattern, or are they generated by a non-random algorithm? [{sequence}]

A: The sequence was generated by a ________

Figure 2: **Prompt templates used for the Randomness Generation (Top) and Judgment (Bottom) tasks.** $\{\texttt{sequence}\}$ is substituted with a list of context flips x, e.g. `Heads`, `Tails`, `Tails`, `Heads`. $\{\texttt{p}\}$ is substituted with the probability of Tails, and $\{\texttt{1} - \texttt{p}\}$ with the probability of Heads.

so the model's flips will be formatted consistently. For these experiments, we collected 200 output sequences y for each LLM at each $P(Tails) \in [.05, .1, .2, .3, .4, .49, .5, .51, .60, .70, .80, .90, .95]$, cropping output tokens to $|y| = 50$ to limit cost and for simplicity.

In the Judgment task, x includes the prompt question and the full sequence of flips. We systematically vary prompt context x by varying the number of flips, denoted $|x|$. We test few-shot learning in LLMs by evaluating output behavior on specific bit sequences, for example $(\texttt{01})^n$ with varying n (e.g. "`Heads`, `Tails`, `Heads`, `Tails`, . . ."), as well as zero-shot language generation dynamics when x is empty and $|x| = 0$ (in practice, we initialize x with a single flip to help the LLM match the correct format).

Chat-Formatted Prompts

Since ChatGPT (not including `gpt-3.5-turbo-instruct`) and GPT-4 use a ChatCompletions API instead of Completions, we re-formatted the prompts in Fig. 2 to follow user/assistant/system prompt format. The following prompts were used:

System	Your responses will only consist of comma-separated "Heads" and "Tails" samples. Do not repeat the userś messages in your responses.
User	Generate a sequence of 1000 random samples from a weighted coin, with {1 - p}% probability of Heads and { p }% probability of Tails.
Assistant	[{ sequence }

Although results are not shown here, for Randomness Judgment experiments, we also tested `text-davinci-003` with prompts other than the one in Fig. 2, including specifying *a non-random algorithm* instead of *a weighted coin, with { 1 - p }% probability* . . . , and found similar results of concept learning dynamics as in Fig. 9.

Models Used

The models we use together capture a variety of behavioral patterns with varying complexity: Bernoulli processes serve as a baseline for a true random coin flip, where all elements in a sequence are independent and identically distributed; Markov chains are a minimal model capable of representing serial correlation patterns; the Gambler's Fallacy bias in cognitive science (described below) emerges from a simple memory-limited model that draws the sequence towards a specified probability dependent based on previous samples; and Regular languages are well studied in theoretical computer science [24, 25], representing a simple class of non-random programs with known mechanisms of finite automata. We use a subset of regular languages $(x)^n$, where (x) is a short sequence of values, e.g., $(\texttt{010})^n$, where 0 maps to Heads and 1 to Tails.

C Related Work

Formal languages and transformers. A number of recent works explore how transformers and other neural language models learng formal languages [26–36]. One common theme is that neural networks often learn 'shortcuts', degenerate representations of formal languages that fail out-of-samples.

In-Context Learning as Bayesian inference A number of recent works frame ICL as Bayesian model selection [12, 37–40]. Two key differences in our work are: first, we analyze state-of-the-art LLMs based on behaviors alone, whereas prior work trains models from scratch on synthetic data and analyzes model parameters directly. Second, we consider Bayesian inference as an empirical modeling framework, as well as a theory, whereas these works only do the latter.

Mechanistic interpretability of transformer models Prior work has characterized specific circuit-level implementations of simple high-level behaviors such as sequence copying, modular addition, and other primitive computational operations [2, 4, 41–52]. Our work differs in that we model hypothetical algorithms to characterize LM output behavioral patterns, without observing underlying activation patterns. We see this as analogous to cognitive science complementing neuroscience in the understanding of human cognition. We characterize the high-level "cognitive" representations in LLMs as a step towards connecting low-level explanations of neural circuits, such as induction heads, with sophisticated high-level behaviors that are characteristic of LLMs.

Language model evaluations Our work resembles evaluation benchmarks such as BIG-Bench [23] that use behavior alone to evaluate LM understanding and reasoning. However, as described later, the domain of subjective randomness is fundamentally different in that there is no "correct" answer. Linguistic probing attempts to characterize the structure of LM representations, but unlike our work, is a function of hidden unit activations rather than output behavior.

LLM Text Generation Dynamics Work on chain-of-thought reasoning in LLMs demonstrates how a few exemplars of detailed solutions or even a simple prompt like "let's think this through step-by-step" can dramatically impact model performance [21–23], but typically only the model's final answer is analyzed, not the trajectory of its intermediate steps. Our memory-constrained Window Average model, inspired by Hahn and Warren [53], is similar in spirit to the claim of Prystawski and Goodman [54], that '[chain-of-thought] reasoning emerges from the locality of experience'. Zhang et al. [55] demonstrate that invalid reasoning can snowball in LLMs, where hallucinations during intermediate steps lead to hallucinations in the final answer.

Random number generation in LLMs Renda et al. [56] explore random number generation in LLMs, in addition to cursory explorations by [57, 58]. These investigations do not analyze dynamics of sequence generation, nor do they ground their analysis, as we do, in theories of ICL as Bayesian model selection and the cognitive science of subjective randomness. Ortega et al. [59] uses a similar domain as ours with random binary sequences and has a similar binary tree visualization over possible sequences, but they train models from scratch and analyze model hidden states, rather than behavioral trajectories as we do.

Bayesian program learning in cognitive science Our work is inspired by computational cognitive science work that theoretically treats concepts as programs, and empirically uses structured Bayesian models to understand human cognition in various domains [8, 10, 11, 60]. We use models based on the cognitive science of subjective randomness [61], drawing particularly on the Bayesian program induction definitions of subjective randomness in Griffiths and Tenenbaum [62, 63], Griffiths et al. [64]. Our method of studying learning as probabilistic inference over formal languages with varying $|x|$ is also similar to Goodman et al. [9], Piantadosi et al. [20], Yang and Piantadosi [65], Bigelow and Piantadosi [66], who use more sophisticated grammar-based models of concept learning.

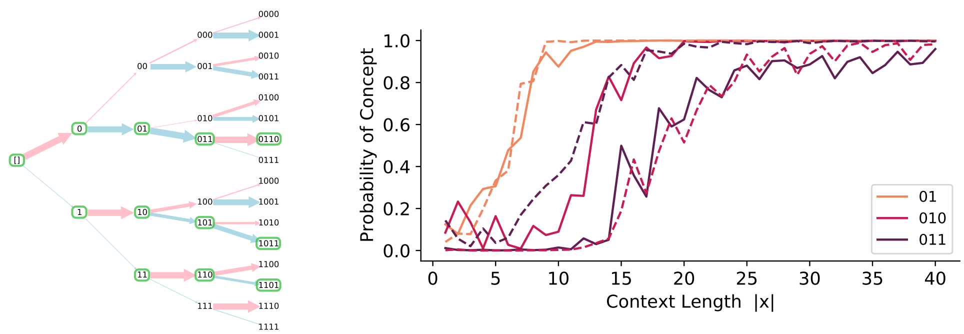

Figure 3: **With enough context, GPT-3.5 learns simple formal language concepts, transitioning from generating pseudo-random numbers to only generating values from the concept.** (Left) Visualization of predictive distribution $p(y|x)$ as a weighted tree, with trajectories matching the concept $C = (\texttt{011})^n$ highlighted in green. (Right) Corresponding probabilities $p(y|x)$ sharply transition from pseudo-random sequence generation, to deterministic repetition of the formal language concept.

D Formal Language Learning

In Figure 3, the GPT-3.5 next-token predictive distribution $p(y|x)$ for `text-davinci-003` visualized as a binary tree, where red arrows correspond to *Heads*, blue arrows to *Tails*, and nodes matching the target concept $C = (011)^n$ are green. The probability table for $p(y_t|y_{0,\ldots t-1})$ is a weighted binary tree with depth d, where edges represent the next-token probability $p(y_t|y_{t-1})$, and paths represent the probability of a sequence of tokens. $p(y|x)$ changes with varying $|x|$, here $|x| = 39$ and next-token predictions strongly follow the concept C. Also see Fig 6, 7.

In the right side of Figure 3, we show in-context learning dynamics for simple formal languages $x = (\texttt{HTH})^n$ (`010`) and $x = (\texttt{HTT})^n$ (`011`) as a function of context length $n = |x|$ (note: this figure is repeated in the main text). Prediction accuracy computed as the total probability mass assigned to valid continuations of the formal language x, as a function of prediction depth $d = |y|$ and context length $|x|$. Curves shown are for $d = 6$, where only 3 out of 64 total paths y match concept C. Solid lines correspond to `text-davinci-003` and dashed lines to `gpt-3.5-turbo-instruct`; note that learning curves for `010` and `011` flip between the two models. Also see Fig. 4, 5.

Varying Prediction Depth d

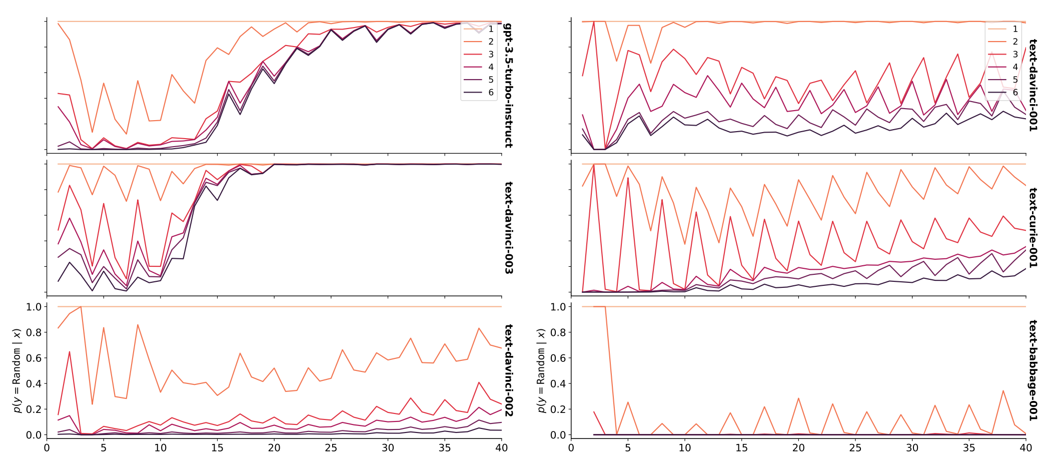

Figure 4: **Predictive distributions $p(y|x)$ by each LLM for Concept $(\texttt{010})^n$, at each prediction depth** d. Colors correspond to different prediction depths, also refer to Figure 3. Note: `text-ada-001` results are not shown since results did not follow the required format (*'Heads, Tails, . . .'*) adequately to be analyzed.

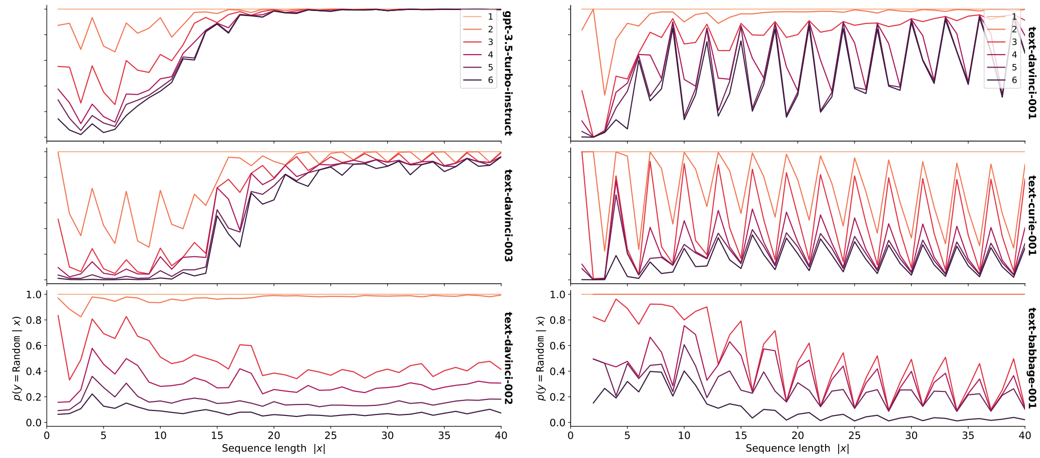

Figure 5: **Predictive distributions $p(y|x)$ by each LLM for Concept $(\texttt{011})^n$, at each prediction depth** d. Colors correspond to different prediction depths, also refer to Figure 3.

Additional Predictive $p(y|x)$ Trees

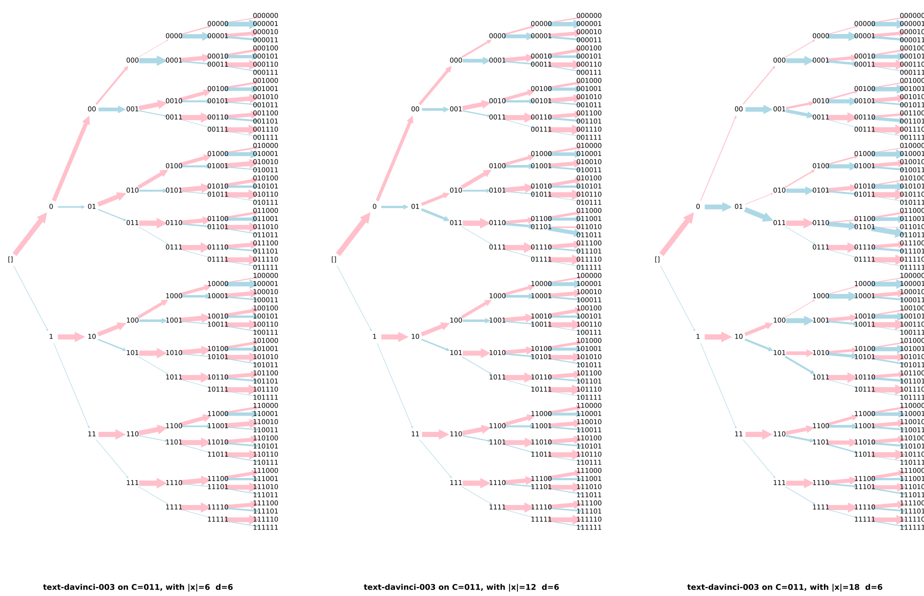

Figure 6: **Predictive distribution $p(y|x)$ trees with $d = 6$ for concept $C = (011)^n$ with** $|x| \in \{6, 12, 18\}$. Since $|x|$ is increasing by the same depth as the tree $\Delta_{|x|} = d = 6$, the transition from generating pseudo-random numbers to deterministically repeating 011 is visibly apparent. Also see Figure 3.

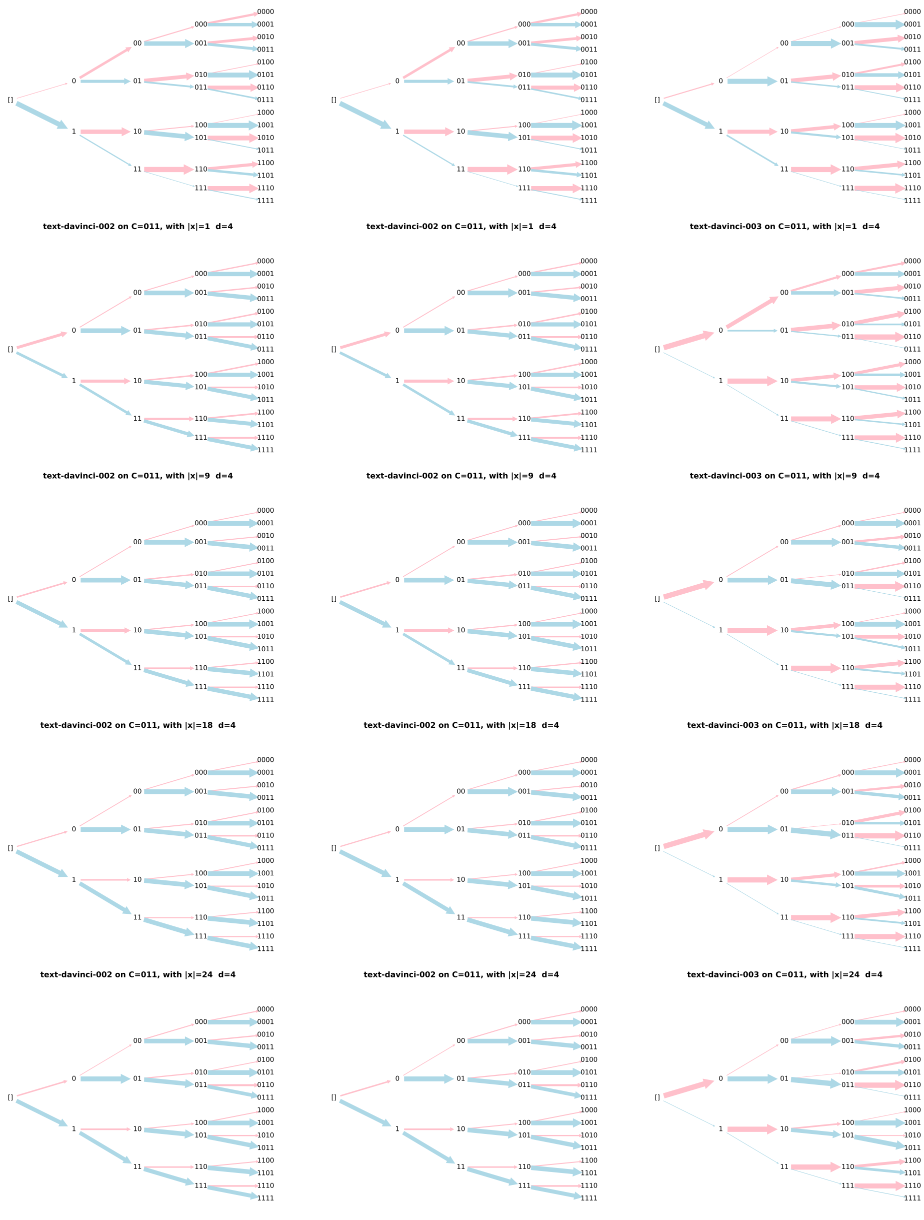

Figure 7: **Predictive distribution $p(y|x)$ trees with $d = 4$ for concept $C = (011)^n$ with** $|x| \in \{1, 9, 18, 24, 39\}$. Models shown are `text-davinci-002`, `text-davinci-003`, and `gpt-3.5-turbo-instruct`. Also see Figure 3.

E Randomness Judgments

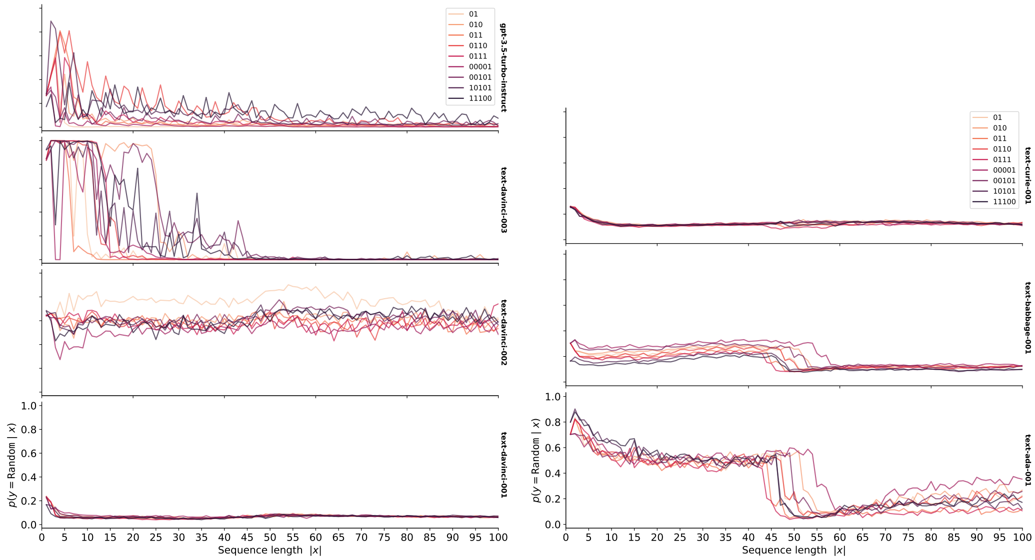

Figure 8: **Randomness judgments across GPT models for 9 concepts**.

(Fig 8) `text-davinci-003` shows a stable pattern of being highly confident (high token probability) in the process being random up to some amount of context $|x|$, at which point it rapidly transitions to being highly confident in the process being non-random, with transition points varying substantially between concepts. `chat-gpt-3.5-instruct` does not go through a stable high-confidence random period like `text-davinci-003`, and stable high-to-low confidence dynamics are observed for only a subset of concepts. The majority of earlier GPT models (`text-davinci-002`, `text-davinci-001`, `text-curie-001`, `text-babbage-001`) show no 'formal language learning', at all. However, surprisingly OpenAI's smallest available GPT model `text-ada-001` shows S-shaped in-context learning dynamics, with the peak close to .5 instead of 1.0 as in `text-davinci-003`. Additionally, the learning dynamics and transition points for all concepts appear nearly identical, approximately at $|x| = 50$, and some concepts show less stable "non-random" patterns for larger $|x|$.

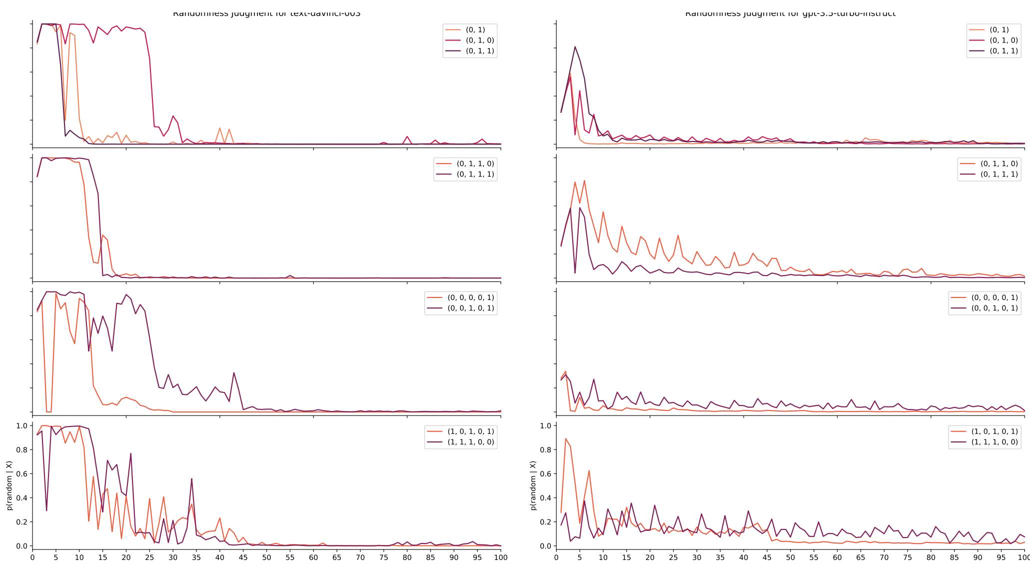

Figure 9: Randomness Judgment $p(y = \texttt{random}|x)$ dynamics for each concept tested, for `text-davinci-003` and `gpt-3.5-turbo-instruct`

F Random Sequence Generation by GPT Model

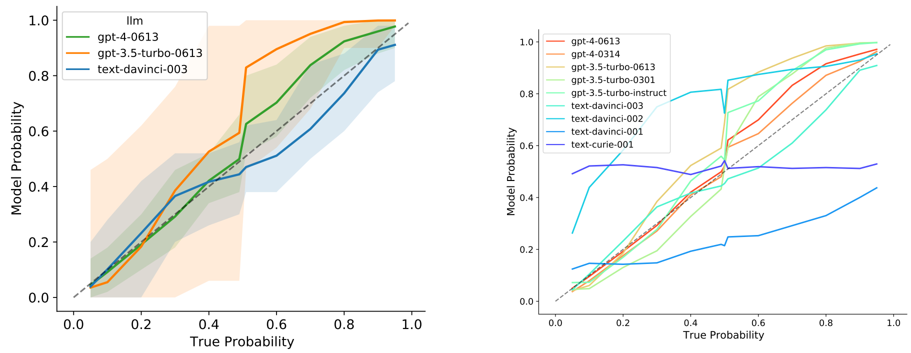

Figure 10: **Probability** $p(\texttt{Tails})$ **bias across LLMs** `text-davinci-003` and GPT-4 models are least biased relative to the specified $p(\texttt{Tails})$ (x-axis). In the left figure, error bars represent the maximum and minimum sequence means $\overline{y}$ for each $p(\texttt{Tails})$.

.

Our cross-LLM analysis (Fig. 10, 11) shows that `text-davinci-003` is controllable with $P(\texttt{Tails})$, with a bias towards $\overline{y} = .50$ and higher variance in sequence means (though lower variance than a true Bernoulli process). ChatGPT (`gpt-3.5-turbo-0301` and `0613`) demonstrate similar behavior for $P(\texttt{Tails}) < 50\%$, but behave erratically with higher $P(\texttt{Tails})$ and the majority of sequences y converge to repeating '`Tails`'. GPT-4 (*0301*, *0613*) show stable, controllable subjective randomness behavior, but with lower variances than sequences generated by `text-davinci-003`. Earlier models do not show subjective randomness behavior, with `text-davinci-002` and `text-davinci-001` being heavily biased and uncontrollable, and `text-curie-001` generates sequences with $\overline{y} = .50$ regardless of $P(\texttt{Tails})$.

Fig. 11 and the left side of Fig. 10 demonstrate that `text-davinci-003` and GPT-4 models not only are more controllable, following the correct probability more closely on average, but also have substantially lower variance than ChatGPT, which is both less controllable and has more variability in its distribution of responses. Further, GPT-4 is lower variance than `text-davinci-003`, with sequences staying even closer to their means $\overline{y}$.

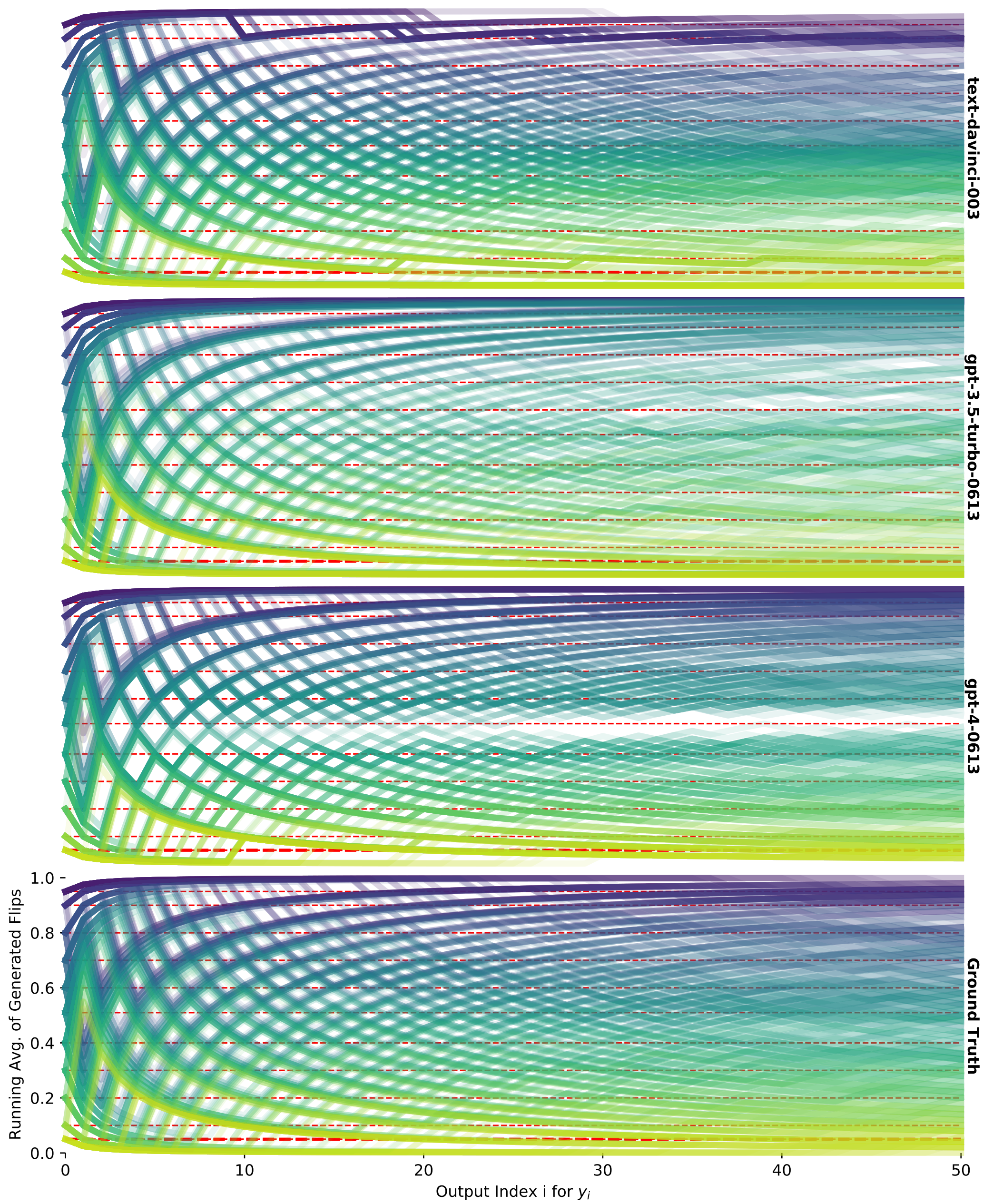

Figure 11: 50 sequences sampled by each GPT model, for each $p(Tails)$. Color is assigned according to specified $p(Tails)$. Red dotted lines are drawn for each $p(Tails)$.

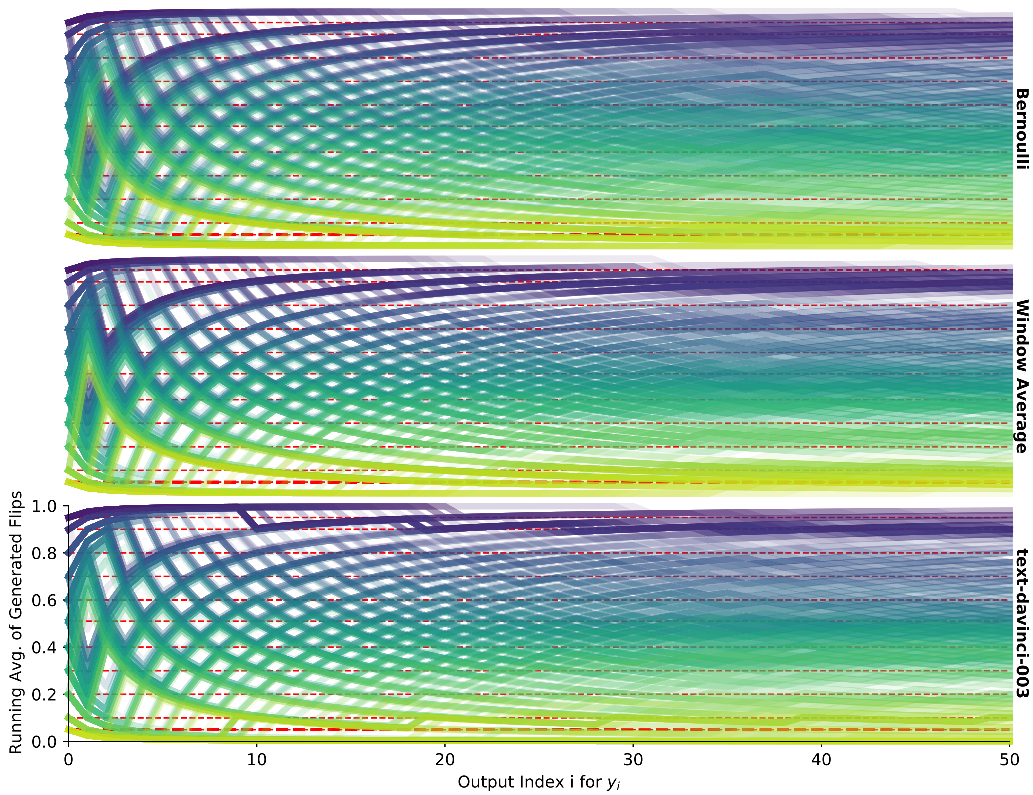

Figure 12: 50 sequences sampled by `text-davinci-003`, for each $p(Tails)$, compared with samples from Bernoulli and Window Average models fit to y_{LLM} for each $p(Tails)$. Color is assigned according to specified $p(Tails)$.

G Gambler's Fallacy Metrics by GPT Model

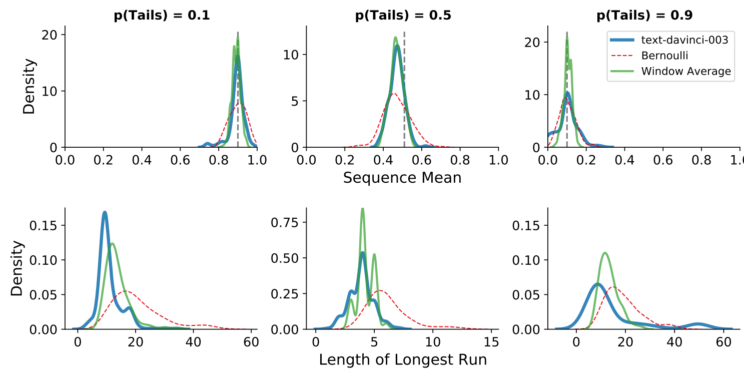

Figure 13: **GPT-3.5 shows a Gambler's fallacy bias of avoiding long runs.** (Top) Distribution of mean values of flip sequences ($\mu = \frac{1}{T}\sum_t y_t$) generated by GPT-3.5 (`text-davinci-003`) with the specified $p(Tails)$, compared with a Bernoulli process and our Window Average model with the same mean as the GPT-3.5 flips. Flips generated by GPT approximately follow the expected mean $p(Tails)$, but have lower variance than a Bernoulli distribution. (Bottom) Length of the longest run for each sequence, where a run is a sub-sequence of the same value repeating. In this case, we see a clear bias in GPT-3.5 to avoid long runs, with a similar pattern across all values of $p(Tails)$ despite the x-axis changing in scale.

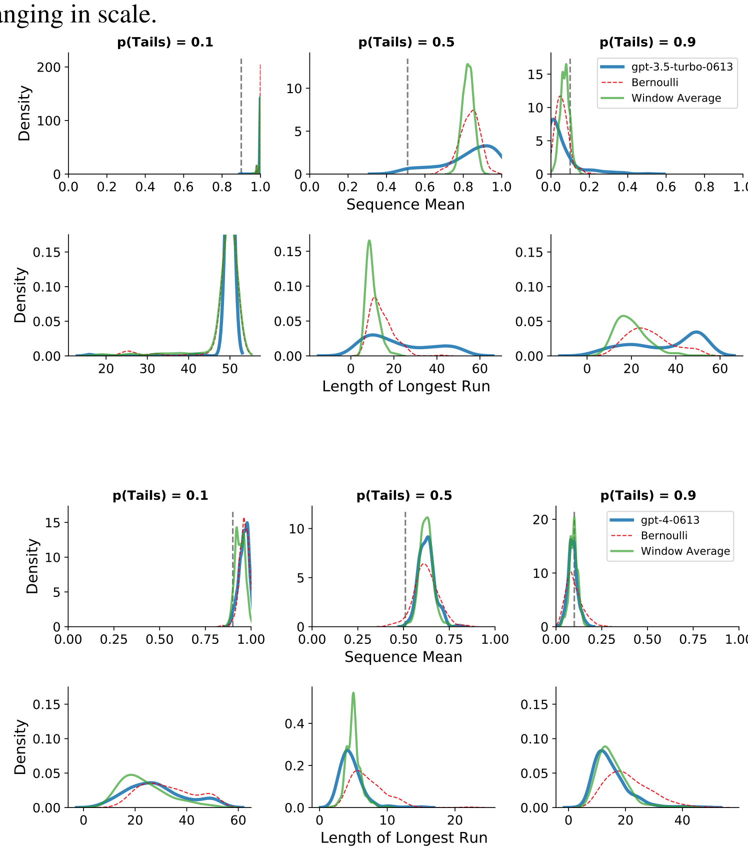

Figure 14: **Gambler's Fallacy histograms for ChatGPT (Top) and GPT-4 (Bottom)**. Also see Fig. 13.

ChatGPT shows no clear Gambler's Fallacy bias, whereas GPT-4 does show this pattern, but is less pronounced than `text-davinci-003` (Fig. 14).

In both plots of Fig. 15, we observe that `text-davinci-003` shows a Gambler's Fallacy bias across $p(\texttt{Tails})$, of higher-than-chance alternation rates and shorter runs; ChatGPT

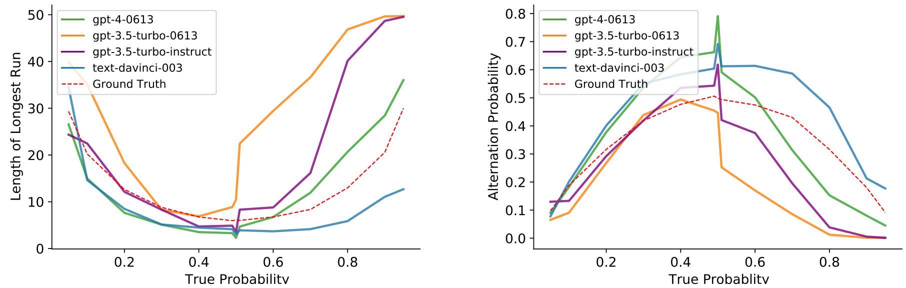

Figure 15: **Comparing metrics of Gambler's Fallacy across probabilities and LLMs** (Left) The mean longest run for each sequence y, at each specified probability $p(\texttt{Tails})$, where a run is a consecutive sub-sequence of the same flip repeating multiple times in a row. (Right) The mean alternation rate for each LLM, where alternation rate is the fraction of consecutive flips that are not equal $p(y_t \neq y_{t-1})$.

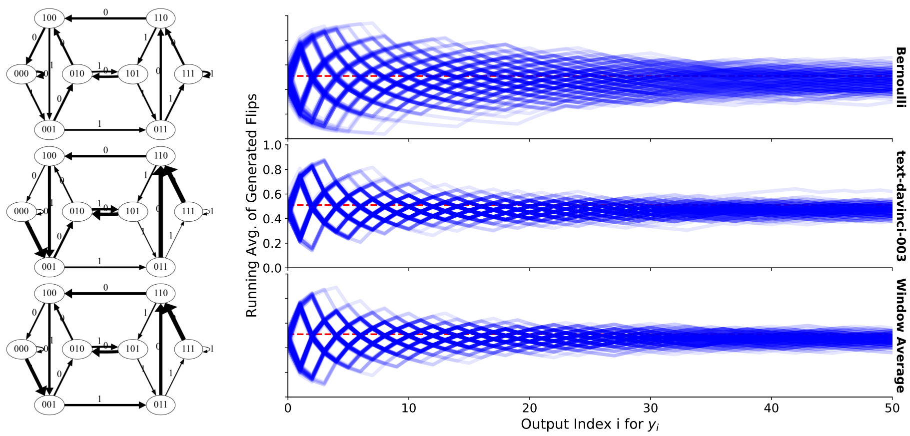

Figure 16: **GPT-3.5 generates pseudo-random binary sequences that deviate from a Bernoulli process.** (Left) Empirical conditional probabilities for a third-order Markov Chain fit to sequences y generated by GPT-3.5 `text-davinci-003`, a Bernoulli process centered at the mean of GPT sequence $\overline{y}$, and our Window Average model ($w = 5$). In the simulated Bernoulli process, edges are fairly uniform; the conditional probabilities for GPT-3.5 and the Window Average model demonstrate a similar non-uniform bias. (Right) Running averages for flip sequences generated by each model, where 0 denotes '`Heads`' and 1 denotes '`Tails`'. Compared to a Bernoulli process (top), sequences generating using GPT (middle) and those of our Window Average model (bottom) stay closer to the mean, repeating the same patterns more often.

(`gpt-3.5-turbo-0613`) produces more tails-biased and higher-variance sequences y when $p(\texttt{Tails}) > 50\%$; GPT-4 and `gpt-3.5-turbo-instruct` interpolate between the two distinct trends of `text-davinci-003` and ChatGPT. The red dotted line represents a Bernoulli process with mean $p(\texttt{Tails})$.

It is unclear how the capabilities we identify are implemented at a circuit level, or why they only seem to emerge in the most powerful and heavily tuned GPT models. For the latter, one hypothesis is that internet corpora contain text with human-generated or human-curating subjectively random binary sequences, and fine-tuning methods such as instruction fine-tuning, supervised fine-tuning, and RLHF make LLMs more controllable, enabling them to apply previously inaccessible capabilities in appropriate circumstances. Another hypothesis is that these fine-tuning methods bias LLMs towards non-repetitiveness, or induce some other general bias that plays a role in the in-context learning dynamics we observe in our particular domain. We hope that future work in cognitive and mechanistic interpretability will shed further light on these questions.

H Memorization, Compression, and Complexity

Across three metrics of sequence complexity — number unique sub-sequences, Gzip file size, and inter-sequence Levenshtein distance (see Fig. 20, 19 in Appendix) — we find that *GPT-3.5+ models, with the exception of ChatGPT, generate low complexity sequences*, showing that structure is repeated across sequences and supporting Goldblum et al. [67], Delétang et al. [68]. By the metrics of mean Levenshtein distance and number of unique sub-sequences, ChatGPT generates higher complexity sequences than chance. We speculate that this phenomenon might explained by a cognitive model that avoids sampling with replacement.

For the Generation task, we note that with a specification of $P(Tails) = 50\%$, but not 49%, 51% or other values, sequences y generated by GPT-3.5+ are dominated by repeating *'Heads, Tails, Heads, Tails, ...'*. This pattern is consistent across variations of the prompts listed in Fig. 2, including specifying 'fair' or 'unweighted' instead of a 'weighted coin', and produces a visible kink in many cross-p(`Tails`) metrics (Fig. 20, 19, 15). For this reason, in Fig. 13 we show results for $P(Tails) = 51\%$.

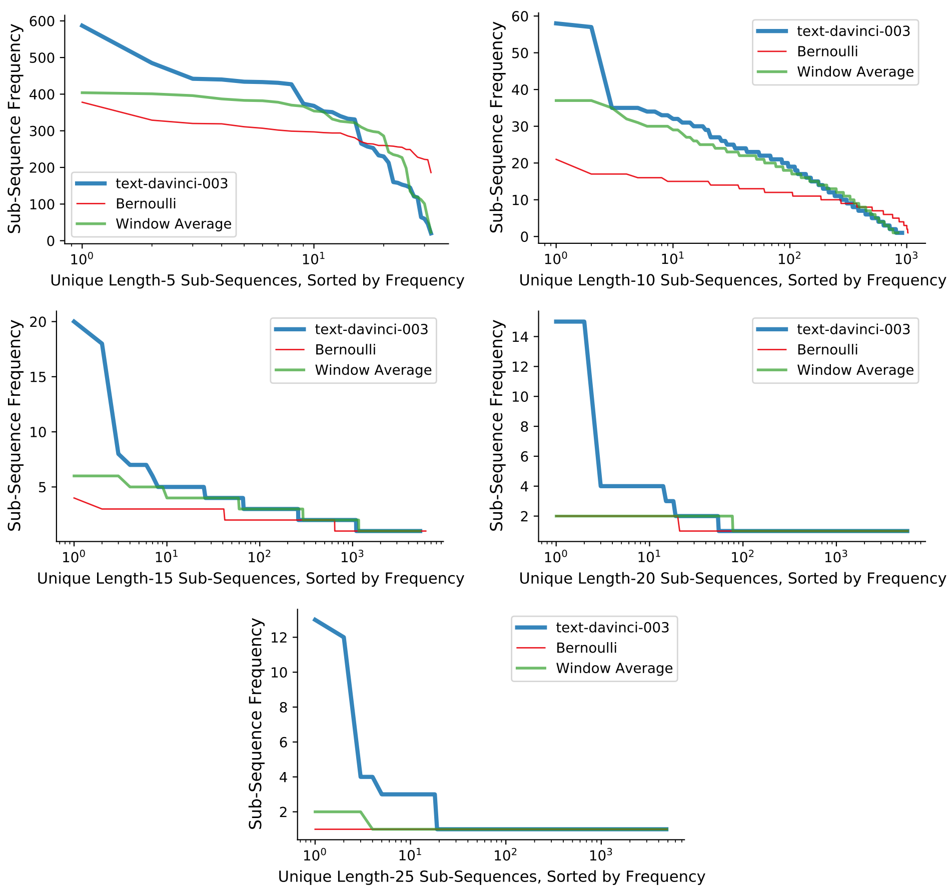

Figure 17: **Distribution of unique sub-sequences for `text-davinci-003` for varying sub-sequence lengths**

.

In Figure 17, we find that GPT repeats specific sub-sequences more often than chance (Bernoulli with $\mu = \overline{y}$), or what is predicted by our Window Average model. While the Window Average model (green) generates fewer unique sub-sequences than a Bernoulli process (red), this does not account for the bias in GPT-3.5 (`text-davinci-003`, in blue) to repeat many of the same sub-sequences. This disparity increases with longer sub-sequences

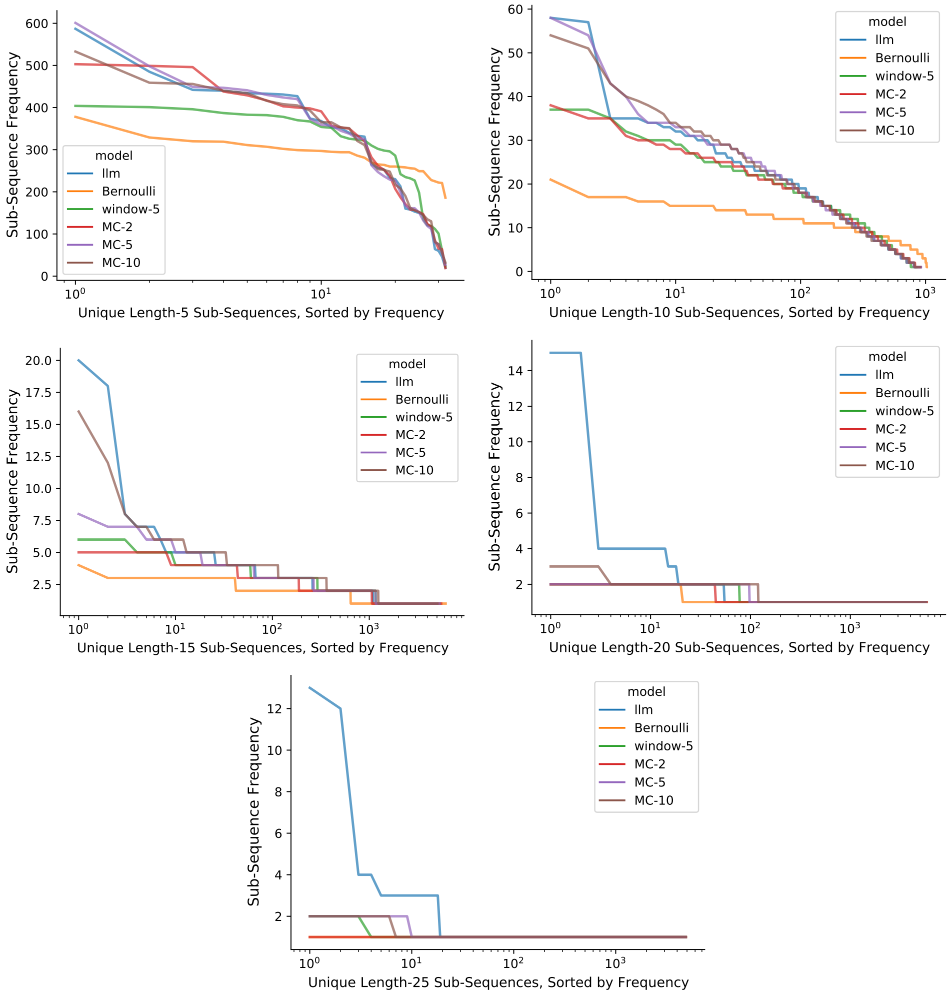

Figure 18: **Distribution of unique sub-sequences for `text-davinci-003`, with additional models, varying sub-sequence lengths** MC-2, MC-5, and MC-10 are Markov Chain models fit to GPT-3.5 flips, with orders $k = \{2, 5, 10\}$

.

In Fig. 18, we show that Markov chains of high order k can account for the sub-sequence distribution, but this only applies when $k <= w$ where w is the sub-sequence length, and the Markov chains can effectively memorizing the sub-sequence distribution of y.

Across both unnormalized and normalized distributions of unique sub-sequences (Fig. 19), we find that GPT-4 repeats the same length-10 sub-sequences significantly more than the other models, and both ChatGPT-based models (`gpt-3.5-turbo-0613`, `gpt-3.5-turbo-instruct`) follow different patterns for $p(\texttt{Tails}) < 50\%$ and $p(\texttt{Tails}) > 50\%$, even when controlling for sequence bias (Right). The only model that generates more unique sub-sequences than chance (above dotted line) is ChatGPT (`gpt-3.5-turbo-0613`).

As a coarse approximation of sequence complexity, we use Gzip file size of appended sequences $gzip(y : y' : y'' : \ldots)$ and mean Levenshtein distance between sequences $d(y, y')$. Gzip [69], a common algorithm for file compression that is highly optimized for compressing strings with redundancy into small file sizes, and Gzip file size has been found to be an effective feature extractor for NLP [70]. Levenshtein distance [71] is a measure of edit distance between two strings.

Since sequence compression is highly correlated with probability, e.g. all sequences with $\overline{y} = 0.99$ will be highly compressible, we normalize the distribution of both plots in Fig. 20 by dividing by the same metric (appended Gzip size, or mean Levenshtein distance) for a Bernoulli distribution centered

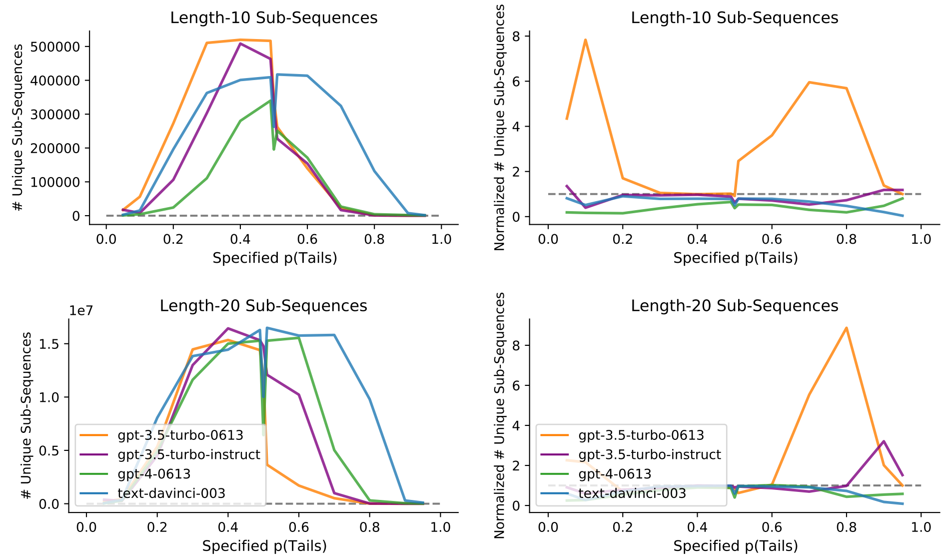

Figure 19: **GPT-4 repeats the same sub-sequences more often than other GPT models** (Left) Number of unique length-10 and length-20 sub-sequences as a function of specified probability $p(\texttt{Tails})$, across all sequences y (note: $|y| = 50$) generated by each GPT model. (Right) The same distributions, with the y-axis normalized by dividing by the same metric (appended Gzip size, or mean Levenshtein distance) for a Bernoulli distribution centered at $\overline{y}$, to control for sequence compression being correlated with probability, e.g. with $\overline{y} = 0.99$, the same sub-sequences of only *'Tails'* flips will appear many times.

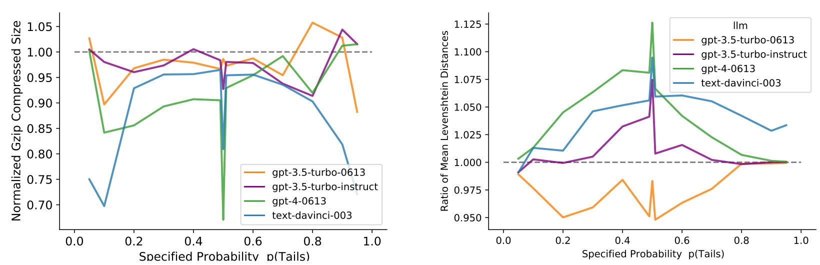

Figure 20: **GPT-generated sequences have lower complexity than Bernoulli sequences**

at $\overline{y}$. For all GPT models except ChatGPT, generated sequences have smaller Levenshtein distance than a Bernoulli process. This is evidence that these LLMs are using memorized sub-sequences ('parroting'), since sequences have repeated structure. On the other hand, ChatGPT produces more dissimilar sequences than chance, suggesting *higher* complexity. In Gzip file size, however, we see a lower-complexity bias in all LLMs (except for a few higher values of $p(\texttt{Tails})$), to varying degrees, produce data Y that is more compressible than data from an equal probability Bernoulli process.

I Background on Algorithmic and Subjective Randomness

We focus on cognitive interpretability of LLMs in the domain of random sequences of binary values. Random binary sequences are a minimal domain that have been studied extensively in statistics, formal language theory, and algorithmic information theory [24, 72, 73]. We can use this domain to systematically test few-shot learning as a function of context length $|x|$ by testing different input sequences x. We can also test zero-shot learning by having models generate sequences with no context ($|x| = 0$), without relying on alternate prompt formats such as chain-of-thought reasoning [21, 22]. Moreover, language generation trajectories over binary sequences can also be analyzed and visualized much more easily than typical user-chatbot interaction trajectories [55, 74], since the token-by-token branching factor is only two. Random binary sequences have also been a target domain in cognitive science (specifically, *subjective randomness*), where researchers have studied the mechanisms and concepts that underlie how people generate random binary sequences or evaluate the randomness of given sequences [61, 64, 75].

Randomness of a sequence x, defined in terms of Bayesian model comparison between the class of non-random models with the class of random models, can be translated to be the difference between the sequence length $|x|$ and the algorithmic complexity, or *Kolmogorov complexity* of the sequence $K(x)$.

$$\begin{aligned}\text{randomness}(x) &= \log P(x|\text{random}) - \log P(x|\text{non-random}) \\ &= \log\ 2^{-|x|} - \log\ 2^{-K(x)} \\ &= K(x) - |x|\end{aligned}$$

The likelihood given a truly random Bernoulli process $p(x|\text{random}) = 2^{-|x|}$ since sequences of equal length have equal probability and there are $2^{|x|}$ binary sequences of length $|x|$. This can be thought of as a uniform prior over programs, where every program is an exact copy of the output string.

The likelihood of x given the space of non-random processes marginalizes over the posterior of all non-random programs (hypotheses) $\mathcal{H}$:

$$p(x|\text{non-random}) = \sum_{h \in \mathcal{H}} p(h)\ p(x|h)$$

.

A natural prior for programs $p(h)$ is the description length of that program, where common metrics used in software engineering such as *lines of code* or *number of functions* can be seen as practical estimations of program description length.

If we assume $p(x|h)$ is a binary likelihood, that is:

$$p(x|h) = \begin{cases} 1 & \text{if } h \text{ generates } x \\ 0 & \text{otherwise} \end{cases}$$

and we simplify the problem to finding the maximum a-priori hypothesis h, and set a prior over hypotheses (programs) proportional to their length $p(h) = 2^{-|x|}$, this equates to finding the program with lowest Kolmogorov complexity $K(x)$:

$$P(x|\text{non-random}) \approx \max_h p(h)\ p(x|h) = 2^{-K(x)}$$

where Kolmogorov complexity $K(x)$ is defined as the description length of the shortest program that generates x as output:

$$K(x) = \underset{\{p \in \Sigma^* | Evaluate(p) = x\}}{\text{argmin}} |p|$$

The notation $p \in \Sigma^*$ is analogous to $h \in \mathcal{H}$, but refers to a formal alphabet Σ that programs are comprised of. In the general case, Kolmogorov complexity $K(x)$ is uncomputable due to the halting problem, since the expression $Evaluate(p) = x$ might run forever if p has an infinite loop.

References

[1] Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*, 2022.

[2] Neel Nanda, Lawrence Chan, Tom Liberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability. *arXiv preprint arXiv:2301.05217*, 2023.

[3] Michael Hanna, Ollie Liu, and Alexandre Variengien. How does gpt-2 compute greater-than?: Interpreting mathematical abilities in a pre-trained language model. *arXiv preprint arXiv:2305.00586*, 2023.

[4] Ziqian Zhong, Ziming Liu, Max Tegmark, and Jacob Andreas. The clock and the pizza: Two stories in mechanistic explanation of neural networks. *arXiv preprint arXiv:2306.17844*, 2023.

[5] Stephanie CY Chan, Adam Santoro, Andrew K Lampinen, Jane X Wang, Aaditya Singh, Pierre H Richemond, Jay McClelland, and Felix Hill. Data distributional properties drive emergent few-shot learning in transformers. *arXiv preprint arXiv:2205.05055*, 2022.

[6] Alberto Bietti, Vivien Cabannes, Diane Bouchacourt, Herve Jegou, and Leon Bottou. Birth of a transformer: A memory viewpoint. *arXiv preprint arXiv:2306.00802*, 2023.

[7] Howard Gardner. *The mind's new science: A history of the cognitive revolution*. Basic books, 1987.

[8] Joshua Tenenbaum. Bayesian modeling of human concept learning. *Advances in neural information processing systems*, 11, 1998.

[9] Noah D Goodman, Joshua B Tenenbaum, Jacob Feldman, and Thomas L Griffiths. A rational analysis of rule-based concept learning. *Cognitive science*, 32(1):108–154, 2008.

[10] Tomer D Ullman and Joshua B Tenenbaum. Bayesian models of conceptual development: Learning as building models of the world. *Annual Review of Developmental Psychology*, 2: 533–558, 2020.

[11] Tomer D Ullman, Noah D Goodman, and Joshua B Tenenbaum. Theory learning as stochastic search in the language of thought. *Cognitive Development*, 27(4):455–480, 2012.

[12] Yu Bai, Fan Chen, Huan Wang, Caiming Xiong, and Song Mei. Transformers as statisticians: Provable in-context learning with in-context algorithm selection. *arXiv preprint arXiv:2306.04637*, 2023.

[13] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[14] Mengjie Zhao, Yi Zhu, Ehsan Shareghi, Ivan Vulić, Roi Reichart, Anna Korhonen, and Hinrich Schütze. A closer look at few-shot crosslingual transfer: The choice of shots matters. *arXiv preprint arXiv:2012.15682*, 2020.

[15] Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning*, pages 12697–12706. PMLR, 2021.

[16] Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Rethinking the Role of Demonstrations: What Makes In-Context Learning Work?, October 2022. URL http://arxiv.org/abs/2202.12837. arXiv:2202.12837 [cs].

[17] Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. *arXiv preprint arXiv:2104.08786*, 2021.

[18] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*, 2022.

[19] Miles Turpin, Julian Michael, Ethan Perez, and Samuel R Bowman. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. *arXiv preprint arXiv:2305.04388*, 2023.

[20] Steven T Piantadosi, Joshua B Tenenbaum, and Noah D Goodman. Bootstrapping in a language of thought: A formal model of numerical concept learning. *Cognition*, 123(2):199–217, 2012.

[21] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022.

[22] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213, 2022.

[23] Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*, 2022.

[24] Michael Sipser. Introduction to the theory of computation. *ACM Sigact News*, 27(1):27–29, 1996.

[25] Noam Chomsky. Three models for the description of language. *IRE Transactions on information theory*, 2(3):113–124, 1956.

[26] Grégoire Delétang, Anian Ruoss, Jordi Grau-Moya, Tim Genewein, Li Kevin Wenliang, Elliot Catt, Chris Cundy, Marcus Hutter, Shane Legg, Joel Veness, et al. Neural networks and the chomsky hierarchy. *arXiv preprint arXiv:2207.02098*, 2022.

[27] Gail Weiss, Yoav Goldberg, and Eran Yahav. Thinking like transformers. In *International Conference on Machine Learning*, pages 11080–11090. PMLR, 2021.

[28] Hui Shi, Sicun Gao, Yuandong Tian, Xinyun Chen, and Jishen Zhao. Learning bounded context-free-grammar via lstm and the transformer: Difference and the explanations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8267–8276, 2022.

[29] Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: Part 1, context-free grammar. *arXiv preprint arXiv:2305.13673*, 2023.

[30] Satwik Bhattamishra, Kabir Ahuja, and Navin Goyal. On the ability and limitations of transformers to recognize formal languages. *arXiv preprint arXiv:2009.11264*, 2020.

[31] Kaiyue Wen, Yuchen Li, Bingbin Liu, and Andrej Risteski. (un) interpretability of transformers: a case study with dyck grammars. 2023.

[32] Bingbin Liu, Jordan T Ash, Surbhi Goel, Akshay Krishnamurthy, and Cyril Zhang. Exposing attention glitches with flip-flop language modeling. *arXiv preprint arXiv:2306.00946*, 2023.

[33] Bingbin Liu, Jordan T Ash, Surbhi Goel, Akshay Krishnamurthy, and Cyril Zhang. Transformers learn shortcuts to automata. *arXiv preprint arXiv:2210.10749*, 2022.

[34] William Merrill and Ashish Sabharwal. The parallelism tradeoff: Limitations of log-precision transformers. *Transactions of the Association for Computational Linguistics*, 11:531–545, 2023.

[35] William Merrill, Nikolaos Tsilivis, and Aman Shukla. A tale of two circuits: Grokking as competition of sparse and dense subnetworks. *arXiv preprint arXiv:2303.11873*, 2023.

[36] William Merrill and Ashish Sabharwal. Transformers implement first-order logic with majority quantifiers. *arXiv preprint arXiv:2210.02671*, 2022.

[37] Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An Explanation of In-context Learning as Implicit Bayesian Inference, July 2022. URL http://arxiv.org/abs/2111.02080. arXiv:2111.02080 [cs].

[38] Yingcong Li, M. Emrullah Ildiz, Dimitris Papailiopoulos, and Samet Oymak. Transformers as Algorithms: Generalization and Stability in In-context Learning, February 2023. URL http://arxiv.org/abs/2301.07067. arXiv:2301.07067 [cs, stat].

[39] Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. *arXiv preprint arXiv:2211.15661*, 2022.

[40] Michael Hahn and Navin Goyal. A Theory of Emergent In-Context Learning as Implicit Structure Induction, March 2023. URL http://arxiv.org/abs/2303.07971. arXiv:2303.07971 [cs].

[41] Gabriel Goh, Nick Cammarata, Chelsea Voss, Shan Carter, Michael Petrov, Ludwig Schubert, Alec Radford, and Chris Olah. Multimodal neurons in artificial neural networks. *Distill*, 6(3): e30, 2021.

[42] Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. *arXiv preprint arXiv:2012.14913*, 2020.

[43] Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 2022.

[44] Kenneth Li, Aspen K Hopkins, David Bau, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Emergent world representations: Exploring a sequence model trained on a synthetic task. *arXiv preprint arXiv:2210.13382*, 2022.

[45] Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. *arXiv preprint arXiv:2211.00593*, 2022.

[46] Bilal Chughtai, Lawrence Chan, and Neel Nanda. A toy model of universality: Reverse engineering how networks learn group operations. *arXiv preprint arXiv:2302.03025*, 2023.

[47] Wes Gurnee, Neel Nanda, Matthew Pauly, Katherine Harvey, Dmitrii Troitskii, and Dimitris Bertsimas. Finding neurons in a haystack: Case studies with sparse probing. *arXiv preprint arXiv:2305.01610*, 2023.

[48] Alex Foote, Neel Nanda, Esben Kran, Ioannis Konstas, Shay Cohen, and Fazl Barez. Neuron to graph: Interpreting language model neurons at scale. *arXiv preprint arXiv:2305.19911*, 2023.

[49] Ekdeep Singh Lubana, Eric J Bigelow, Robert P Dick, David Krueger, and Hidenori Tanaka. Mechanistic mode connectivity. In *International Conference on Machine Learning*, pages 22965–23004. PMLR, 2023.

[50] Tom Lieberum, Matthew Rahtz, János Kramár, Geoffrey Irving, Rohin Shah, and Vladimir Mikulik. Does circuit analysis interpretability scale? evidence from multiple choice capabilities in chinchilla. *arXiv preprint arXiv:2307.09458*, 2023.

[51] Boaz Barak, Benjamin Edelman, Surbhi Goel, Sham Kakade, Eran Malach, and Cyril Zhang. Hidden progress in deep learning: Sgd learns parities near the computational limit. *Advances in Neural Information Processing Systems*, 35:21750–21764, 2022.

[52] Ziming Liu, Ouail Kitouni, Niklas S Nolte, Eric Michaud, Max Tegmark, and Mike Williams. Towards understanding grokking: An effective theory of representation learning. *Advances in Neural Information Processing Systems*, 35:34651–34663, 2022.

[53] Ulrike Hahn and Paul A Warren. Perceptions of randomness: why three heads are better than four. *Psychological review*, 116(2):454, 2009.

[54] Ben Prystawski and Noah D Goodman. Why think step-by-step? reasoning emerges from the locality of experience. *arXiv preprint arXiv:2304.03843*, 2023.

[55] Muru Zhang, Ofir Press, William Merrill, Alisa Liu, and Noah A Smith. How language model hallucinations can snowball. *arXiv preprint arXiv:2305.13534*, 2023.

[56] Alex Renda, Aspen Hopkins, and Michael Carbin. Can llms generate random numbers? evaluating llm sampling in controlled domains llm sampling underperforms expectations, 2023. http://people.csail.mit.edu/renda/llm-sampling-paper.

[57] janus. Mysteries of mode collapse. LessWrong, 2022. URL https://www.lesswrong.com/posts/t9svvNPNmFf5Qa3TA/mysteries-of-mode-collapse.

[58] Andrej Karpathy. A baby GPT with two tokens 0/1 and context length of 3, view[ed] as a finite state markov chain., 2023. URL https://twitter.com/karpathy/status/1645115622517542913.

[59] Pedro A Ortega, Jane X Wang, Mark Rowland, Tim Genewein, Zeb Kurth-Nelson, Razvan Pascanu, Nicolas Heess, Joel Veness, Alex Pritzel, Pablo Sprechmann, et al. Meta-learning of sequential strategies. *arXiv preprint arXiv:1905.03030*, 2019.

[60] Joshua B Tenenbaum, Charles Kemp, Thomas L Griffiths, and Noah D Goodman. How to grow a mind: Statistics, structure, and abstraction. *science*, 331(6022):1279–1285, 2011.

[61] Ruma Falk and Clifford Konold. Making sense of randomness: Implicit encoding as a basis for judgment. 1997.

[62] Thomas L Griffiths and Joshua B Tenenbaum. From Algorithmic to Subjective Randomness. *Neural Information Processing Systems*, 2003.

[63] Thomas L Griffiths and Joshua B Tenenbaum. Probability, algorithmic complexity, and subjective randomness. *Proceedings of the Cognitive Science Society*, 2004.

[64] Thomas L. Griffiths, Dylan Daniels, Joseph L. Austerweil, and Joshua B. Tenenbaum. Subjective randomness as statistical inference. *Cognitive Psychology*, 103:85–109, June 2018. ISSN 00100285. doi: 10.1016/j.cogpsych.2018.02.003. URL https://linkinghub.elsevier.com/retrieve/pii/S0010028517302281.

[65] Yuan Yang and Steven T Piantadosi. One model for the learning of language. *Proceedings of the National Academy of Sciences*, 119(5):e2021865119, 2022.

[66] Eric Bigelow and Steven T Piantadosi. Inferring priors in compositional cognitive models. In *CogSci*, 2016.

[67] Micah Goldblum, Marc Finzi, Keefer Rowan, and Andrew Gordon Wilson. The no free lunch theorem, kolmogorov complexity, and the role of inductive biases in machine learning. *arXiv preprint arXiv:2304.05366*, 2023.

[68] Grégoire Delétang, Anian Ruoss, Paul-Ambroise Duquenne, Elliot Catt, Tim Genewein, Christopher Mattern, Jordi Grau-Moya, Li Kevin Wenliang, Matthew Aitchison, Laurent Orseau, et al. Language modeling is compression. *arXiv preprint arXiv:2309.10668*, 2023.

[69] Peter Deutsch. Gzip file format specification version 4.3. Technical report, 1996.

[70] Zhiying Jiang, Matthew YR Yang, Mikhail Tsirlin, Raphael Tang, and Jimmy Lin. Less is more: Parameter-free text classification with gzip. *arXiv preprint arXiv:2212.09410*, 2022.

[71] Vladimir I Levenshtein et al. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710. Soviet Union, 1966.

[72] Gregory J Chaitin. On the length of programs for computing finite binary sequences. *Journal of the ACM (JACM)*, 13(4):547–569, 1966.

[73] Ming Li, Paul Vitányi, et al. *An introduction to Kolmogorov complexity and its applications*. Springer, 1997.

[74] Lukas Berglund, Asa Cooper Stickland, Mikita Balesni, Max Kaufmann, Meg Tong, Tomasz Korbak, Daniel Kokotajlo, and Owain Evans. Taken out of context: On measuring situational awareness in llms. *arXiv preprint arXiv:2309.00667*, 2023.

[75] Amos Tversky and Thomas Gilovich. The "hot hand": Statistical reality or cognitive illusion? *Chance*, 2(4):31–34, 1989.