Genomic Next-Token Predictors are In-Context Learners

Nathan Breslow, Aayush Mishra, Mahler Revsine
Michael C. Schatz, Anqi Liu, Daniel Khashabi
Department of Computer Science, Johns Hopkins University

Abstract

In-context learning (ICL) – the capacity of a model to infer and apply abstract patterns from examples provided within its input – has been extensively studied in large language models trained for next-token prediction on human text. In fact, prior work often attributes this emergent behavior to distinctive statistical properties in *human* language. This raises a fundamental question: can ICL arise *organically* in other sequence domains purely through large-scale predictive training?

To explore this, we turn to genomic sequences, an alternative symbolic domain rich in statistical structure. Specifically, we study the Evo2 genomic model, trained predominantly on next-nucleotide (A/T/C/G) prediction, at a scale comparable to mid-sized LLMs. We develop a controlled experimental framework comprising symbolic reasoning tasks instantiated in both linguistic and genomic forms, enabling direct comparison of ICL across genomic and linguistic models. Our results show that genomic models, like their linguistic counterparts, exhibit log-linear gains in pattern induction as the number of in-context demonstrations increases. To the best of our knowledge, this is the first evidence of organically emergent ICL in genomic sequences, supporting the hypothesis that ICL arises as a consequence of large-scale predictive modeling over rich data. These findings extend emergent meta-learning beyond language, pointing toward a unified, modality-agnostic view of in-context learning.

1 Introduction

Scaling Large Language Models (LLMs) has revealed an unexpected and powerful capacity: in-context learning (ICL) (Radford et al., 2018; Brown et al., 2020a), the ability to infer and apply abstract patterns purely from examples contained within their input. Unlike conventional adaptation, which relies on gradient updates, LLMs internalize mechanisms for flexible pattern induction (Olsson et al., 2022b), allowing them to perform few-shot generalization and analogical reasoning (Srivastava et al., 2023), *without explicit parameter updates*. This behavior, which arises organically from large-scale "next-token" prediction on human language, has been interpreted as *emergent* forms of meta-learning (learning to learn) within a *fixed* parametric system.

Almost all prominent evidence of emergent ICL so far (Srivastava et al., 2023, inter alia) comes from training on human language (e.g., English). This raises a fundamental question: *is there something inherently special about human language that enables ICL to emerge?* One can take two different stances on this:

- ***H_1:*** Human language possesses distinctive distributional properties such as parallelism or compositionality (Chen et al., 2024; Hahn & Goyal, 2023) that uniquely nurture ICL.
- ***H_2:*** Alternatively, perhaps natural language is merely a convenient substrate – having ample data and being naturally interpretable – and ICL could just as well arise in any

LLMs were used during the conception, implementation, and refinement of this paper. We assume full responsibility for all content.

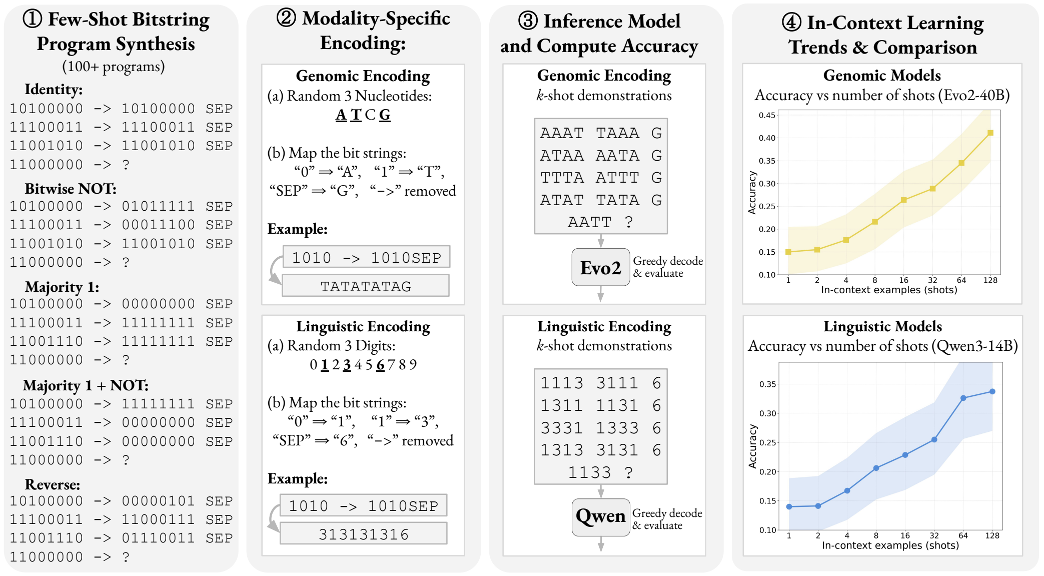


Figure 1: We design parallel symbolic reasoning tasks that allow direct comparison of ICL behavior across modalities (linguistic and genomic). ① Few-shot bitstring program-synthesis tasks (e.g., identity, NOT, majority, reverse) require models to infer mappings from examples. ② Each task is rendered in two modality-specific encodings: genomic (bitstrings mapped to random nucleotides A/T/C/G) and linguistic (bitstrings mapped to random digits), preserving abstract structure but differing in surface form. ③ Both genomic (Evo2) and linguistic (Qwen3) models receive k-shot demonstrations and are greedily decoded to compute exact-match accuracy. ④ **Both models show log-linear accuracy gains with more demonstrations**.

sufficiently structured data domain, provided the model is large enough and trained to compress predictive patterns.

While many works have examined $\boldsymbol{H}_1$, the veracity of $\boldsymbol{H}_2$ remains far less understood. To this end, we turn to a radically different substrate: **the genome**. Genomic sequences can be viewed as a form of *natural* language that has evolved through nature. Like human language, they comprise complex symbol sequences that exhibit rich statistical regularities – motifs, repeats, dependencies (Benegas et al., 2025) that could, in principle, support pattern induction within context.

Testing for the existence of ICL requires large-scale models. Historically, genomic models were small and specialized, optimized for narrow biological tasks such as motif discovery or structure prediction. The advent of deep biological models (Ji et al., 2021) has expanded their scale, paralleling those of early LLMs. Most recently, the **Evo2** series (Brixi et al., 2025) marks a major step forward: a large-scale model trained solely on 'next-nucleotide' prediction. With training scale comparable to mid-sized LLMs (e.g., Qwen3-14B (Yang et al., 2025)), these genomic models finally make it possible to systematically compare genomic and linguistic representations under large-scale autoregressive training. If ICL arises primarily from scale and predictive compression ($\boldsymbol{H}_2$), then large-scale genomic models such as Evo2 must already exhibit it *under our noses*. Like LLMs, they are trained for next-token prediction without any explicit meta-learning signal. Yet, despite extensive analysis of Evo2's biological capabilities (e.g., protein fitness prediction, variant effect estimation), its potential for emergent few-shot or analogical reasoning within context remains unexplored.

The second challenge in evaluating $\boldsymbol{H}_2$ lies in developing a framework to detect and quantify ICL in the genomics domain where "tasks" lack natural human interpretability. Moreover, the task design must permit systematic comparison between ICL behaviors in human language and genomic modalities. To investigate this, in §3 we introduce a controlled experimental framework (Fig. 1) for evaluating ICL across linguistic and genomic modalities. As illustrated in ①–②, we design a suite of symbolic bitstring program-synthesis tasks and render them in two parallel encodings: one using *genomic* alphabets (nucleotides: A/T/C/G) and one using *linguistic* (characters). We then evaluate both model families – Evo2 for genomic sequences and Qwen3 for linguistic text – under matched few-shot prompting conditions (③). This parallel encoding enables direct, apples-to-apples comparison of ICL scaling trends across genomic and linguistic models.

Our experiment's results (§4) reveal striking parallels: **both linguistic and genomic models exhibit log-linear improvements** in pattern induction and extrapolation as the number of demonstrations grows. To our knowledge, this is the first evidence of organically emergent ICL in genomic sequences. These findings suggest $\boldsymbol{H}_2$, that is, *ICL is not an artifact that is specific to human language, but a broader consequence of compression and predictive modeling in pattern-rich sequence spaces.* By demonstrating ICL in a non-linguistic biological domain, we extend the scope of emergent meta-learning beyond natural language, pointing toward a unified view of context-adaptive computation that spans different modalities of data.

We further perform analyses to compare the types of tasks at which genomic and linguistic models excel. Our results reveal distinct patterns: genomic models learn faster with more demonstrations, while linguistic models gain more from scale (§4.2); genomic models perform best on direct copying or local transformations, whereas linguistic models excel on tasks requiring single-bit dependencies or global summary statistics (§4.3, §4.4). We note the caveat that these findings are specific to our experimental conditions (e.g., our choice of reasoning tasks) and may not be a holistic statement about ICL in genomic and linguistic models. We further validate these findings by demonstrating that Evo2 also exhibits robust few-shot learning on a real-world genomic promoter classification task (Appendix E).

Our Contributions. (a) We present a controlled experimental setup for detecting and measuring ICL across linguistic and genomic modalities. Our framework defines a suite of symbolic bitstring reasoning tasks that can be rendered in both representations, enabling direct, apples-to-apples comparison of ICL behavior. (b) Using the Evo2 family of large-scale genomic models, we demonstrate for the first time that models trained solely on nucleotide sequences exhibit clear in-context learning, mirroring the scaling trends of Qwen3 language models. (c) Our findings challenge the idea that ICL depends on linguistic structure. Instead, they point to ICL as a modality-agnostic outcome of large-scale next-token prediction over pattern-rich data, reflecting general principles of compression and contextual inference. Together, these results extend the scope of emergent meta-learning beyond language, toward a unified understanding of context-adaptive computation in artificial and biological systems alike.

2 Related Work and Broader Context

Why does ICL remain interesting? ICL remains a striking and conceptually rich phenomenon: it emerges *without* task-specific training yet enables rapid, in-situ adaptation. It comes closest to long-standing ambitions in classical AI—such as case-based (Aamodt & Plaza, 1994; Lancaster & Kolodner, 1987; Ross, 1984; Ellman, 1989) and analogical reasoning (Hofstadter, 2001; Gentner & Hoyos, 2017; Holyoak et al., 2001) where agents solve new problems by reusing solutions from prior (analogous) examples.

Beyond its theoretical appeal, ICL has become a practical workhorse in recent advances, e.g., synthetic data generation using pretrained models (Wang et al., 2023c; Shao et al., 2023; Tunstall et al., 2023) or reasoning (Wei et al., 2022; Nye et al., 2021; Yao et al., 2023; Yasunaga et al., 2023). In short, ICL is both a window into emergent learning dynamics and a cornerstone for building adaptive, interpretable systems.

Emergent-ICL $\neq$ Meta-ICL: A parallel line of research studies models trained with an explicit *ICL objective* – learning to predict over collections of input–output pairs $(x, f(x))$. For example, regression models that are trained *explicitly* to infer an underlying rule (e.g., linear, polynomial, or sinusoidal) from few-shot examples (Garg et al., 2022; Li et al., 2023c; Raventós et al., 2023; Nejjar et al., 2024). This setup mirrors *meta-learning*, where a model is explicitly "trained to learn" (Finn et al., 2017; Bertinetto et al., 2019; Zintgraf et al., 2019). We refer to such systems as *Meta-ICL* (Panwar et al., 2023; Min et al., 2022; Wu et al., 2022; Kirsch et al., 2022; Zhang et al., 2023), in contrast to *emergent ICL* in large pretrained models, which arises organically from next-token prediction over natural data.

Meta-ICL results demonstrate that transformers *can be optimized* to perform meta-learning but fail to explain why *emergent* ICL appears in models never trained for it. Critically, Meta-ICL generalizes poorly outside its training domain (Naim et al., 2024). For example, a Meta-ICL model trained on linear and cosine functions fails on their composition (Yadlowsky et al., 2023)—and does not capture higher-order abilities like in-context reasoning (Wei et al., 2022; Kojima et al., 2022) or adaptive self-refinement (Madaan et al., 2023) – unless explicitly trained for them. In contrast, emergent ICL in LMs generalizes broadly and flexibly across diverse problem types (Srivastava et al., 2023). These behavioral and mechanistic differences support our working assumption that *Meta-ICL* and *emergent ICL* are fundamentally distinct phenomena (Shen et al., 2024; Mishra et al., 2025). Throughout, our focus in this work is on the latter – the organic emergence of in-context learning in large pretrained models.

ICL $\neq$ Zero-shot learning: Existing literature in the domain of timeseries studies the zero-shot inference capabilities of next-token predictors, i.e., given a context of a given problem their models can predict the forthcoming values (tokens) (Ansari et al., 2024; Das et al., 2024). This differs from the type of ICL we evaluate, which explicitly relies on *few-shot* learning of a transformation, not zero-shot adaptation. In fact, in Fig. 2, we report that Evo2's performance begins far below a naive mode baseline (simply guessing the most common output) at one shot, indicating very poor zero-shot capability. Evo2's ability only becomes meaningful *only when many shots are provided.* This makes the type of ICL we demonstrate qualitatively different from zero-shot ability, as it explicitly only appears in few-shot contexts.

Are there any prior results on emergent ICL in genomic models? Most genomic models to date are *encoders* (Ji et al., 2021; Dalla-torre et al., 2024), which preclude autoregressive generation and the study of ICL in the standard sense. Among the few autoregressive genomic models, most remain small, e.g., GENA-LM (300M) (Fishman et al., 2024) and scGPT (53M) (Cui et al., 2024). The recently-released Evo2 series (Brixi et al., 2025) includes models up to 40B parameters, comparable in scale to modern LLMs. This model family provides the first viable testbed for probing *emergent* ICL, which we study here. A related effort, HyenaDNA (Nguyen et al., 2023), trains on million-token genomic sequences and reports ICL-like behaviors; however, these involve *meta-ICL* mechanisms (e.g., soft prompting and instruction fine-tuning; see their Sec. 4.3), rather than organically emergent ICL. Similarly, Bio-xLSTM (Schmidinger et al., 2024) demonstrates ICL, but only after explicitly training on deliberately constructed few-shot demonstrations instead of natural biological data - another example of Meta-ICL. An additional avenue of research demonstrates that genomic models can perform a form of ICL by copying prior tokens in-context for prediction (Kantroo et al., 2025) – akin to how induction heads function (Olsson et al., 2022a). This, however, doesn't demonstrate any ability to infer latent functions or apply a novel rule in-context.

Are there instances of emergent ICL in other non-linguistic modalities? While transformers have been extended to many modalities beyond language, genuine cases of *emergent* ICL arising purely from large-scale pretraining remain rare. In neuro-signal modeling, for instance, EEG-GPT (Kim et al., 2024) is trained with explicit in-context demonstrations, making it an instance of *Meta-ICL* rather than emergent adaptation. Similarly, multimodal models such as Flamingo (Alayrac et al., 2022), BLIP/BLIP-2 (Li et al., 2022; 2023a), and Emu (Sun et al., 2023; 2024) exhibit in-context behavior only because they are *explicitly trained* to process demonstration patterns – again, meta-learned rather than emergent. Another example of non-linguistic ICL is Chronos 2 – a timeseries model trained to perform in-context learning through cross-batch learning, where the model's performance can be improved by learning from multiple time series provided in-context. However, Chronos 2 is trained using exclusively synthetic data designed to elicit cross batch learning and ICL, which firmly places it in the Meta-ICL regime (Ansari et al., 2025). To our knowledge, the only claimed instance of emergent ICL outside language is in vision: Bai et al. (2024) show that transformers pretrained on natural visual sequences can infer analogical and compositional tasks without explicit supervision. The weakness of this result is that their ICL prompts consist of long sequences of contiguous frames (Bai et al., 2024, Fig 10), which effectively reduces the task to next-frame prediction, rather than true ICL that requires inferring compositions of multiple demonstrations.

3 Experimental Framework for Cross-domain In-context Learning

This section presents our framework for evaluating ICL across linguistic and genomic models. We outline the experimental desiderata (§3.1), setup (§3.2), model selection (§3.3), and evaluation protocol (§3.4). We additionally release code for replicating these experiments on Github.

3.1 Experimental Desiderata

Desideratum 1: Cross-domain comparability. Our experimental framework requires that each task be performable by *both* language and genomic models. Thus, every task must be representable in both linguistic and nucleotide alphabets. This constraint excludes existing benchmarks that rely on domain-specific semantics – such as language reasoning datasets (e.g., BIG-Bench, StrategyQA, GSM8K (Srivastava et al., 2023; Geva et al., 2021; Cobbe et al., 2021)) or biological tasks (e.g. variant effect prediction, exon identification (Brixi et al., 2025)). While one could, in principle, translate domain-specific tasks into an alternate alphabet (e.g., mapping language tokens to base sequences via quaternary encoding), doing so inherently biases the evaluation: the source domain retains advantages, while the target domain must operate on representations that are unnatural to it. As a result, such cross-domain encodings confound the comparison, reflecting representational translation artifacts rather than genuine differences in ICL behavior.

Desideratum 2: Limited vocabulary. Since genomic models operate on only four nucleotides (A, T, C, G), tasks must be expressible within an equally compact alphabet. This rules out the existing symbolic reasoning and analogy benchmarks (Hodel & West, 2023; Lewis & Mitchell, 2025; Webb et al., 2025), which rely on richer vocabularies or background knowledge (e.g., shapes, colors, or linguistic tokens with explicit semantic/lexical roles). While some of these tasks are more symbolic in nature, they still rely on pretrained knowledge or linguistic instructions about numerical or lexical ordering (i.e. 'e' coming after 'a b c d') – information that couldn't effectively be conveyed in-context to a genomic model (Lewis & Mitchell, 2025; Stevenson et al., 2025). Such tasks cannot be faithfully represented in a four-token regime without introducing artifacts or structural loss. Accordingly, our evaluation focuses on tasks that retain abstract reasoning structure while remaining compatible with the low-vocabulary symbolic space shared across linguistic and genomic models.

3.2 Experimental Setup

Notation: We formally define ICL as follows. Let S be the input domain and O the output domain, and let a task be a latent deterministic function $f : S \to O$. A pretrained autoregressive model M performs ICL by conditioning on an ordered n-shot demonstration set $E = \big((x_1, f(x_1)), \ldots, (x_n, f(x_n))\big)$, where $x_i \in S$, and then receiving a held-out query $x \in S \setminus \{x_1, \ldots, x_n\}$. Conditioned on (E, x), the model produces a prediction $\hat{y} = M(E, x)$, which we compare to the ground-truth $f(x)$. Then the single-trial success is $\mathbb{1}\big[\hat{y} = f(x)\big]$.

Task Formulation: in-context program induction over abstract symbols. Given our desiderata (§3.1), we design symbolic induction tasks that require a small lexicon. Each task requires a model to infer a hidden transformation rule from a few input–output examples and apply it to a new query. We refer to this process as *program synthesis in-context*. This setup isolates in-context learning ability by removing the influence of pretrained world knowledge. If the function family f and symbol space S are chosen carefully, the evaluation is far from the model's pretraining distribution. Similar formulations have been used in prior work on compositional reasoning (Brown et al., 2020b), analogical reasoning via Raven's Progressive Matrices (Raven et al., 1962; Webb et al., 2023), and more recently in ARC-AGI (Chollet, 2019).

Bitstring domain and function space. To maintain compatibility across genomic and linguistic models, we represent all symbols as bitstrings of length k, i.e., $S = \{0,1\}^k$. Genomic models operate over four nucleotides (A, C, G, T), allowing two tokens (e.g., A/C) to encode 0/1 and reserving the others (G/T) as delimiters or separators. Bitstrings are also naturally supported by linguistic models since most tokenizers represent digits as single tokens, ensuring parity in symbol granularity. This design minimizes tokenization confounds while providing a uniform symbolic substrate for comparing both domains. See Fig. 1 for examples. Despite their simplicity, bitstrings support a wide range of composable operations – from basic (identity, constant) to logical (AND, OR), positional (shift, rotation, reversal), and aggregate (majority, parity). This expressivity makes them a convenient and extensible testbed for cross-domain in-context learning evaluation.

We set the symbol space to $S = \{0,1\}^8$, corresponding to all 8-bit binary strings. This choice balances expressivity (256 unique bitstrings) with manageable prompt length. To define the task space, we construct a set of 100 transformation functions $F \subseteq \{f \mid f : S \to S\}$, where each function maps one bitstring to another according to a deterministic rule. Each $f \in F$ is either a single primitive operation or a composition of two primitives, allowing for a controlled yet diverse range of symbolic transformations.

Primitive operations. We build F from a library of 30 fundamental primitives (listed in Appendix A.1) spanning six functional categories: (1) *Bitwise transformations* (e.g., flipping individual bits), (2) *Structural rearrangements* (rotations, shifts, reversals, swaps), (3) *Positional masking and selection* (preserving or zeroing specific bit positions), (4) *Pattern detection* (detecting alternating or palindromic patterns), (5) *Aggregation functions* (majority/minority, parity), and (6) *Trivial mappings* (identity, constant outputs). This taxonomy ensures coverage of both low-level logical operations and higher-order compositional structure.

Automatic generation and validation. To construct the final function set, we use GPT-5-Codex to automatically generate 100 unique transformations by composing primitives into valid Python programs. Each candidate function is verified programmatically for *logical distinctness*: no two programs $f, g \in F$ produce identical outputs for all inputs ($\nexists f, g$ such that $f(x) = g(x)$ for all $x \in S$). All functions undergo manual inspection to confirm correctness, diversity, and adherence to the intended categories. The full list of transformations and corresponding source code is provided in Appendix A.2.

Prompt encoding. Prompts are encoded using a unified symbolic scheme to enable evaluation across linguistic and genomic models. For linguistic models, bits $(0, 1)$ are mapped to two random digits (0–9); for genomic models, to random nucleotides $(\mathrm{A}, \mathrm{T}, \mathrm{C}, \mathrm{G})$. The separator token $(\to)$ is omitted, and in-context examples are separated by a randomly chosen unused token distinct from those representing 0 and 1. (See Fig. 1 for examples.) All mappings are randomized per trial to avoid memorization or positional bias.

3.3 Model Families and Selection Rationale

For fair cross-domain comparison, we use two representative model families: Qwen3 for human language and Evo2 for genomics. The rationale for this selection is as follows:

(a) *Parameter scaling:* Both families span multiple orders of magnitude in parameter count, enabling systematic scaling analysis of ICL ability. The Qwen3 series ranges from 0.6B to 14B parameters, while Evo2 includes 1B, 7B, and 40B models (Yang et al., 2025; Brixi et al., 2025). This parallel scaling structure facilitates consistent measurement of how ICL performance evolves with model size across linguistic and genomic modalities.

(b) *Compute matching:* The largest models in each family are trained with comparable total compute, offering an opportunity for an approximately compute-matched cross-domain comparison. Using the standard $6ND$ estimate (Kaplan et al., 2020), Qwen3-14B-Base is trained with about 3.2×10^{24} FLOPs, while Evo2-40B is trained with 2.25×10^{24} FLOPs (Yang et al., 2025; Brixi et al., 2025), making Evo2 uniquely well-suited for comparison with Qwen3 at scale.

(c) *Availability of base models:* The Qwen3 family releases base (*pre*-instruction-tuned) models at all scales up to 14B parameters, enabling direct evaluation of the intrinsic inductive reasoning ability of pure next-token predictors, without instruction-tuning artifacts. The Evo2 base models have no additional instruction tuning applied—the only nuance is that Evo2 1B is trained at a context length of 8192 nucleotides, whereas the 7B/40B are extended to a context length of one million.

(d) *Tokenizer:* Qwen3 uses a standard BPE tokenizer with a vocabulary size of 151,669. Evo2 uses a byte-level tokenizer, so individual nucleotides are mapped to individual tokens. Notably, Qwen's tokenizer maps single digits to single tokens, allowing for parity in our experiments (Yang et al., 2025; Brixi et al., 2025).

(e) *Context length:* Qwen3's 0.6B and 1.7B models have a context length of 32K. The remaining dense models have a context length of 128K. Evo2's 1B model has a context length of 8K, and the remaining models have a context length of 1M. As none of our experiments approach these limits, we can use all models in both families without fear of context length as a confound (Yang et al., 2025; Brixi et al., 2025).

(f) *Training corpora:* All Qwen3 models are trained on 36 trillion text tokens covering a wide variety of topics and languages. Evo1 1B is trained on 1 trillion tokens, Evo2 7B is trained on 2.4 trillion tokens, and Evo2 40B is trained on 9.3 trillion tokens. These extensive training corpora ensure that any potential ICL dynamics have thoroughly emerged (Yang et al., 2025; Brixi et al., 2025).

(g) *Architecture:* Qwen3 is based on a conventional Llama-like transformer architecture, while Evo2 uses the StripedHyena2 architecture which intersperses convolutional layers with attention-based ones. While ideally both model families would use the same architecture, there were no vanilla transformers at Evo2's compute scale (Yang et al., 2025; Brixi et al., 2025).

(h) *Licensing:* All models are released under Apache 2.0 (Yang et al., 2025; Arc Institute, 2025a;c;b), ensuring reproducibility.

3.4 Evaluation Protocol

Evaluation metric. Building on the setup in §3.2, we now describe how in-context performance is evaluated. For each transformation $f \in F$, we draw an n-shot demonstration set $E \subset S$ and a held-out query $x \in S \setminus E$. The model M receives a few-shot prompt of input–output pairs followed by the query and produces a prediction $\hat{y} = M(E, x)$. A trial is marked correct if $\hat{y} = f(x)$, giving the single-trial indicator $\mathcal{A}(f, E, x, M) = \mathbb{1}\left[\hat{y} = f(x)\right]$.

Since exact enumeration of *all* the programs is intractable, we use a Monte Carlo estimate (a full formulation of what we approximate can be found in the appendix B). Let E_n denote the space of all sets of input bitstrings with cardinality n: $E_n = \{E \subset S : |E| = n\}$. Sample m i.i.d. context sets $E^{(t)} \sim \text{Unif}(E_n)$ for $t = 1, \ldots, m$; for each t, sample $x^{(t)} \sim \text{Unif}(S \setminus E^{(t)})$. Then our empirical estimate of accuracy, for a given transformation f is $\hat{\mathcal{A}}_f(M, n) = \frac{1}{m} \sum_{t=1}^{m} \mathcal{A}\big(f,\ E^{(t)},\ x^{(t)},\ M\big)$ and the overall estimated accuracy for model M is:

$$\hat{P}(M, n) = \frac{1}{|F|} \sum_{f \in F} \hat{\mathcal{A}}_f(M, n). \tag{1}$$

Model suites and sampling configuration. We fix the number of Monte Carlo trials to $m = 8$. The set of evaluated *genomic models* is $\mathcal{M}_G = \{\texttt{evo2\_1b\_base}, \texttt{evo2\_7b}, \texttt{evo2\_40b}\}$ and the set of *linguistic models* is $\mathcal{M}_L = \{\texttt{Qwen3-0.6B-Base}, \texttt{Qwen3-1.7B-Base}, \texttt{Qwen3-4B-Base}, \texttt{Qwen3-8B-Base}, \texttt{Qwen3-14B-Base}\}$. We evaluate across shot counts $\mathcal{N} = \{1, 2, 4, 8, 16, 32, 64, 128\}$. For each model M and shot count n, we compute:

$$P_L = \{\hat{P}(M, n) : M \in \mathcal{M}_L, n \in \mathcal{N}\}, \quad P_G = \{\hat{P}(M, n) : M \in \mathcal{M}_G, n \in \mathcal{N}\}.$$

We estimate standard errors via a two-stage nonparametric cluster bootstrap, resampling functions (clusters) and within each selected function, resampling its evaluation samples with 5000 replicates for each (M, n).

Mode baseline. We define a mode baseline that always predicts the most frequent output observed in the context. For a given function f, context set $E \subset S$, and query input $x \in S \setminus E$, the mode prediction is $\hat{y}_{\text{mode}}(f, E, x) = \arg\max_{y \in S} \left|\{ e \in E : f(e) = y \}\right|$, where ties in the arg max are broken randomly. This simply corresponds to guessing the most common output in the few-shot examples. The overall mode baseline with n shots and across the set of all functions F is:

$$\hat{P}_{\text{mode}}(n) = \frac{1}{m|F|} \sum_{f \in F} \sum_{t=1}^{m} \mathbb{1}\left[\hat{y}_{\text{mode}}(f, E^{(t)}, x^{(t)}) = f(x^{(t)})\right], \tag{2}$$

where $E^{(t)}$ and $x^{(t)}$ are sampled identically to the model evaluation in Eq.1. This baseline corresponds to making an educated guess based only on the overall distribution of function outputs that simply learns the majority statistics of the prompt without attempting to infer the underlying transformation.

4 Empirical Results

This section reports empirical findings on few-shot bitstring generalization. §4.1 presents the main accuracy trends with respect to model size and shot count. §4.3 examines sensitivity to task complexity using a BitLoad measure, and §4.4 contrasts the models' qualitative competencies across individual transformations.

4.1 Main Results

Across both model families, accuracy generally rises linearly with respect to log(shots). A linear regression linking performance to log(shots) yields highly significant slopes for all models (all $p \leq 10^{-3}$ via one-sided t-test on slope). As Fig. 2 shows, Evo2 models show cleanly monotonic gains, with a pronounced step from 1B to 7B, and near-indistinguishable curves for 7B and 40B; by 128 shots, both 7B and 40B surpass 40% accuracy (Fig. 2a). Qwen3 also trends upward overall but with non-monotonic patches—especially for smaller models in the 4–16 shot band—before resuming clear improvements from 32 to 128 shots; at 128 shots, the 14B model approaches 35% (Fig. 2b). A full table of all accuracies is in Appendix C.

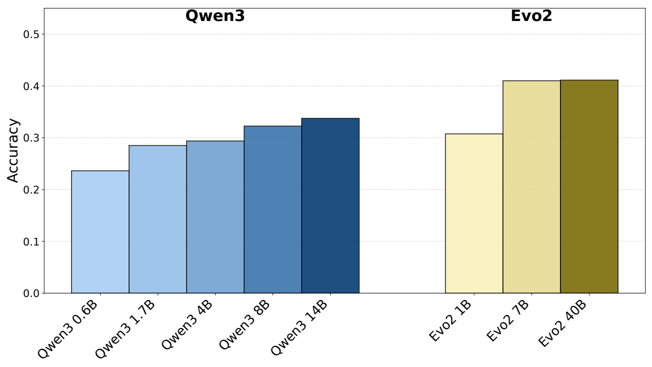


Figure 3: Performance at $n = 128$ shots. All model accuracies increase monotonically with respect to parameter count.

Evo2 outperforms Qwen3 at comparable sizes. Evo2 1B beats Qwen3 1.7B and Evo2 7B beats Qwen3 8B at higher shot counts (Fig. 2c). Averaging within families reinforces the same picture: comparable performance in the 1–4 shot range, Evo2 pulling ahead around 8–16 shots, and the gap persisting at higher shot counts (Fig. 2d).

The mode baseline's performance doesn't improve with more shots, and lags Qwen3 and Evo2 at high shot counts (Fig. 2d). This indicates that the in-context learning performed by Qwen3 and Evo2 is qualitatively different than simply sampling from the distribution of possible outputs in-context. Qwen3 and Evo2's performance requires that they learn, in-context, how to condition on the input. Furthermore, for Qwen3, advantages

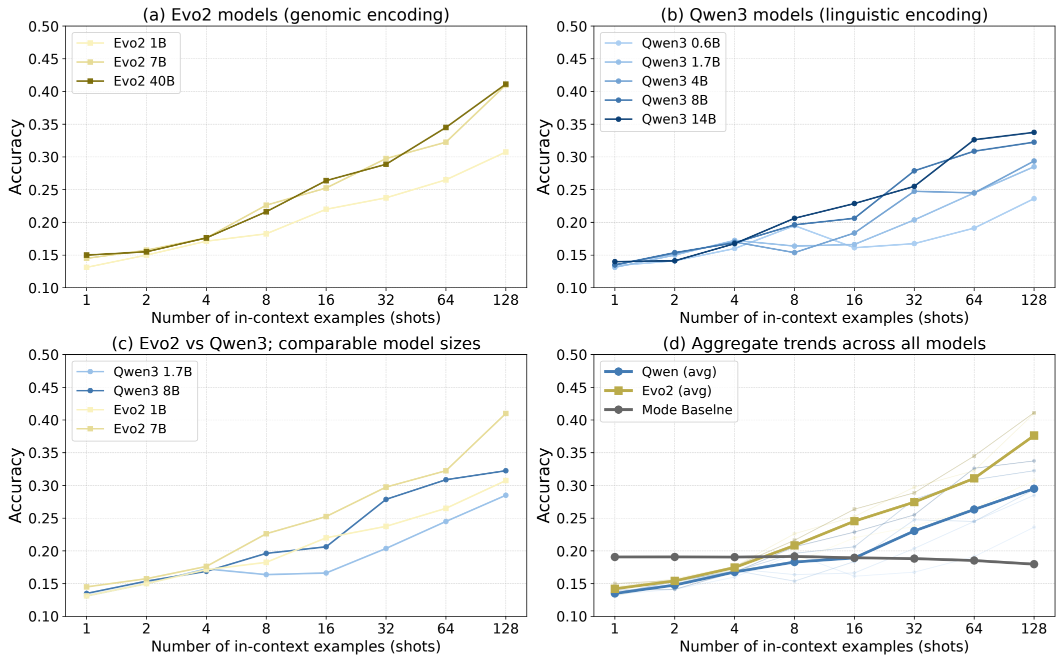


Figure 2: Few-shot performance of Qwen3 and Evo2 models. (a) Evo2 model performance with respect to log(shots). All models monotonically improve – the 7B and 40B have roughly equivalent performance, and the 1B trails behind them. (b) Qwen3 model performance with respect to log(shots). All models improve, but not always monotonically. Smaller models struggle in 4-16 shot range. (c) At comparable sizes, Evo2 outperforms Qwen3. (d) Averaged performance across both model families shows consistent improvement with respect to log(shots). All models exceed the mode baseline shown in gray color. The exact accuracies and error bars (bootstrap-based standard errors) are included in Appendix C.

over the mode baseline become consistently significant ($p < 0.05$ via one-sided z-test on bootstrapped standard errors) at n=128 for all sizes greater than 0.6B. For Evo2, statistically significant advantages emerge slightly earlier: for 1B at n=64, the 7B at $n = 32$, and the 40B at $n = 16$. Regardless, all models surpass the naive baseline.

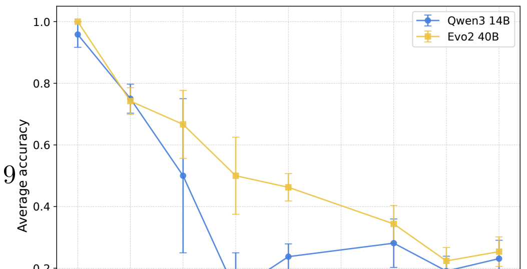
4.2 Analysis of ICL Capability Across Model Families and Sizes

As shown in Fig. 3, **both Qwen3 and Evo2 exhibit clear performance gains with increasing scale**. Qwen3 displays a strong positive correlation between accuracy and parameter count ($p < 0.05$ for all shot counts $\geq$16, one-sided t-test on slope), while Evo2 shows a distinct jump from 1B to 7B but little improvement beyond that. This suggests that in-context program induction becomes more robust with scale, though saturation may occur once models reach sufficient capacity (A detailed statistical analysis in Appendix D).

4.3 ICL Sensitivity to Task Complexity: BitLoad Analysis

To understand which transformations Qwen3 and Evo2 infer most effectively, we analyze their performance across varying task complexities. We focus on the largest models in each family (Qwen3-14B and Evo2-40B) under the ($n = 128$) shot regime, providing both models ample opportunity to display their ICL abilities.

Defining BitLoad. We introduce *BitLoad*, a measure of a function's intrinsic complexity. Informally, BitLoad quantifies how many input bits influence the output.



Formally, it is defined as:

$$\text{BitLoad}(f) = \sum_{i=1}^{k} \mathbb{1}\left[\exists x, j : f_j(x) \neq f_j(x^{\oplus i})\right], \tag{3}$$

where $x^{\oplus i}$ denotes x with bit i flipped, and $f_j(x)$ returns the j-th output bit. Intuitively, it counts the number of bit positions whose perturbation changes the output. So, the larger the BitLoad of a function, the harder it is since it requires the model to attend to more bits. This metric is similar to the existing statistical measure of "relevant features", just defined specifically for our bitstring manipulation tasks. (Nevo & El-Yaniv, 2002).

Fig. 4 shows the mean accuracy of Qwen vs Evo with respect to the BitLoad of all of our tasks. A full table of the BitLoad of every tested task is attached in the appendix A.3. We see that both models achieve near-perfect accuracy on constant (0 BitLoad) tasks and remain similar at BitLoad 1. However, by BitLoad 2, Evo begins to outperform Qwen, and beyond that point Qwen's accuracy drops sharply – falling below 20% by BitLoad 4 – while Evo's performance declines more gradually, remaining above 40% before converging near 20% at BitLoad 8. This asymmetry suggests that Evo maintains partial generalization as dependency depth increases (BitLoad value), whereas Qwen's ICL collapses more abruptly.

While BitLoad strongly correlates with task accuracy, Fig. 4 also shows notable deviations within the error bars. Thus, while task complexity is a strong predictor of ICL accuracy, other factors (e.g., transformation depth, pretraining exposure) likely play meaningful roles, which motivate our qualitative study in §4.4.

4.4 Qualitative Analysis of Functional and Behavioral Differences

To complement the quantitative BitLoad analysis (§4.3), we conduct a per-task comparison to identify qualitative differences between Qwen3 and Evo2. Specifically, we analyze which tasks each model performs best on and how their inductive profiles diverge. We focus on the $n = 128$ shot regime for the largest models – Qwen3-14B and Evo2-40B – where both have maximal opportunity to exhibit ICL. We rank all tasks by model accuracy and examine the top ten for each model. Tasks that appear in one model's top ten but not the other are considered "exclusive."

Exclusive competencies. Qwen's exclusive tasks involve right-shift operations: `"spread_last_bit"` → `"shift_right_zero"` and `"edge_mask"` → `"shift_right_zero"`. For instance, `\shift_right_zero"` pads the bitstring on the left with a zero and truncates the last bit (e.g., 01010000 → 00101000). Qwen achieves 100% accuracy on `"spread_last_bit"` → `"shift_right_zero"`, whereas Evo2 achieves only 50%. A similar but smaller gap appears for `"edge_mask"` → `"shift_right_zero"` (87.5% vs. 62.5%).

In contrast, Evo2's exclusive strengths involve multi-bit transformations. A clear example is `"flip_bits"` → `"right_half"`, which applies bitwise NOT followed by masking the first half of the input (e.g., 01011100 → 00000011). Evo2 achieves 87.5% accuracy, while Qwen only 25%. This task has a BitLoad of four, consistent with Evo's superior performance on medium-complexity (2–4 bit) transformations observed in Fig. 4.

Shared strengths. Despite these differences, 7 of the top 10 tasks are shared between Qwen and Evo, yielding an intersection-over-union of 0.54. Both models excel at simple transformations such as constant-outputs or single-bit dependencies, e.g., `"spread_last_bit"` which copies the final bit to all positions.

Differential skill profiles. To sharpen contrasts, we identify the ten tasks most favoring each model. Qwen's advantages are concentrated in simple shifts and aggregation tasks. It outperforms Evo by 37.5% on the `"minority"` operation (output all 1s if zeros > ones), and by a similar margin on two parity-based tasks requiring counting the number of 1s. These trends suggest that Qwen may be better at reasoning over global properties of bitstrings. Its superiority on simple shifts may also be explained by the extreme rarity of frame shift mutations in DNA due to how catastrophic they are – a single nucleotide offset can decimate an entire protein. This is empirically supported by the fact that single-nucleotide deletions/shifts increase perplexity far more than other common mutations when presented to Evo2 (Brixi et al., 2025).

Evo2, by contrast, dominates tasks requiring full-bitstring manipulation. It achieves 62.5% on bitwise NOT (vs. 0% for Qwen), 62.5% on identity (vs. 12.5%), and large margins on compositions such as `"flip_bits"` $\rightarrow$ `"right_half"` (87.5% vs. 25%) and `"rotl1"` $\rightarrow$ `"flip_bits"` (37.5% vs. 0%). This exposes perhaps the most important difference between Qwen and Evo's ICL in this specific context: Evo can learn simple full-bitstring operations in-context, whereas Qwen cannot. Notably, Qwen3's base models are trivially capable of learning the identity in a more familiar few-shot context – when examples are presented with arrows and newlines separating them, instead of our intentionally unfamiliar encoding. Thus these results should be taken as an existence proof of ICL in Evo2, not a definitive statement of Evo2 having more ICL ability than Qwen. We leave a comparison of these models across broader tasks to future work.

5 Discussion

What are the implications of our findings on prior efforts to explain the emergence of ICL? First, let us organize the existing frameworks for pinpointing the conditions under which ICL emerges:

(E_1) **ICL's emergence is due to data distributional properties:** The distributional properties of data, such as "parallel structures" in human language pretraining data (Chen et al., 2024), its compositional structure (Hahn & Goyal, 2023), "burstiness" (Chan et al., 2022) and other such properties (Wibisono & Wang, 2024; Reddy, 2023) may be of importance (and perhaps necessary) for the emergence of ICL.

(E_2) **ICL's emergence is due to a compression mechanism:** The large-scale compression mechanism during massive pretraining might drive ICL (Elmoznino et al., 2024a;b; Hahn & Goyal, 2023).

(E_3) **ICL's emergence may require specific architectural properties:** While Transformers might be better suited for ICL than LSTMs (Xie et al., 2021), evidence is mixed (Lee et al., 2023), and non-Transformer models have also demonstrated ICL capabilities (Grazzi et al., 2024; Park et al., 2024).

Our findings refine existing hypotheses about ICL's origins. The emergence of ICL in genomic models challenges accounts that rely solely on language-specific distributional structures (E_1). The presence of ICL across both genomic and linguistic models supports the compression-based explanation (E_2), suggesting that large-scale sequence compression and its induced inductive biases drive ICL across modalities. With respect to architecture (E_3), Evo2 – an autoregressive hybrid combining convolutional and attention layers rather than a pure Transformer – exhibits similar scaling behavior, indicating that ICL does not depend on the pure Transformer form. Instead, architecture provides an expressive substrate that enables pattern induction once exposed to sufficiently large and structured data. Overall, these results position ICL as a modality-agnostic outcome of large-scale next-token prediction, rather than a phenomenon tied to linguistic statistics or a specific architecture.

What are the implications of our findings on the frameworks to explain how ICL operates? We next consider the major perspectives that seek to explain how ICL operates. One view holds that ICL functions as a mix of *task learning* and *task retrieval*, with demonstrations serving either to recall pretrained capabilities or to enable learning on the fly (Pan et al., 2023; Lin & Lee, 2024; Wang et al., 2024; Fang et al., 2025). Our symbolic reasoning tasks, instantiated in both linguistic and genomic domains, provide direct evidence for this *task learning* mode, aligning with this hypothesis and prior work (Pan et al., 2023; Fang et al., 2025). Because these tasks do not depend on pretrained semantic priors, they do *not* invoke task *retrieval*, offering limited insight into the Bayesian view that interprets ICL as implicit inference over latent concepts (Xie et al., 2021; Panwar et al., 2023; Wang et al., 2023b; Jiang et al., 2024). Meanwhile, our results remain agnostic toward the optimization-based hypothesis, which posits that ICL implements an implicit gradient-descent-like process (Akyürek et al., 2022; Ahn et al., 2023; Mahankali et al., 2023; Li et al., 2023b), as well as the induction-based account, which attributes ICL to specialized "circuits" for performing inductive generalization (Elhage et al., 2021; Olsson et al., 2022b; Wang et al., 2023a; Bansal et al., 2023; Ren et al., 2024). Together, our results most strongly support the presence of genuine task *learning* within ICL.

Does Evo2 have an innate advantage on these tasks? Possibly, for multiple reasons. First, though Evo2's is trained on less tokens total, all of Evo2's training tokens are long sequences of repeated nucleotides, and very little of Qwen3's training tokens are long sequences of repeated digits. Second, Evo2's StripedHyena2 architecture was found to to significantly outperform a vanilla transformer on long DNA sequences (Brixi et al., 2025). This could give Evo2 an innate advantage on long contexts containing the same few symbols vs Qwen3. Our result results do not imply Evo2's ICL ability is superior to Qwen3's, and we concede that our few-shot prompting setup may be biased toward Evo2. Attempts to make the task more legible to Qwen3, however, ran into the confound that Qwen3 has pretraining exposure to bitstring manipulation. Any prompting setup that included 0s and 1s immediately resulted in an extreme increase in Qwen3's performance. Future work is needed to establish a method that controls for Evo2's structural advantages without granting Qwen3 an unfair edge via its pretraining knowledge.

Why not test the models on semantic tasks? Semantic tasks – such as identifying the capitals associated with countries, classifying malformed proteins, etc. – require a fair amount of pretraining exposure to the concepts involved in the task as well as measuring ICL. While it's undoubtedly ICL when a model infers a semantic transformation (for instance, country$\rightarrow$capital, word$\rightarrow$opposite), the pretraining knowledge necessary to manifest this ICL precludes its use in extreme cross-modality comparisons. Focusing on far simpler bitstring transformations that can be learned entirely in-context allows for an apples-to-apples comparison between Evo2 and Qwen3. To ensure that the ICL observed is not an artifact of this simplified domain, we additionally demonstrate in Appendix E that Evo2 exhibits robust, scaling ICL on a native genomic task (promoter classification), though we exclude this from the main comparison to maintain parity with the linguistic models.

Beyond the $\boldsymbol{H}_1$-$\boldsymbol{H}_2$ dichotomy. While we presented $\boldsymbol{H}_1$ and $\boldsymbol{H}_2$ (in §1) as contrasting hypotheses, they are not necessarily mutually exclusive. It is entirely plausible that both hold simultaneously – that certain distributional properties inherent in natural data, shared across human language and other natural domains such as genomics, contribute to the emergence of ICL. In this view, linguistic compositionality may represent one instance of a broader statistical substrate that fosters ICL. We framed the two hypotheses as a dichotomy primarily to highlight the contrast between language-specific and modality-general explanations, rather than to suggest that only one can be true.

6 Conclusion

We introduce a suite of bitstring reasoning tasks that can be encoded in both natural language and genomic sequences, showing that genomic models – like their linguistic counterparts – exhibit clear in-context learning. Across all Evo2 model sizes, we observe robust log-linear gains in accuracy with increasing demonstrations, paralleling the scaling trends of Qwen3 language models. These findings challenge the notion that ICL is unique to human language, suggesting it emerges whenever an expressive model is trained autoregressively on structured, pattern-rich data.

Potential future work: This proof-of-existence for ICL lays the groundwork for many future research directions. One could broaden the test suite to incorporate tasks beyond bitstrings – up to four symbols are natively available in nucleotides, and far more can be used if one encodes at the codon level.

Another promising direction is mechanistic interpretability: analyzing Evo2's internal activations to identify which circuits enable ICL and how the model aggregates context to predict the next nucleotide. Comparing these mechanisms to known induction circuits (Elhage et al., 2021) in LLMs could reveal whether analogous structures exist across architectures – a question that connects biological and linguistic ICL dynamics.

Finally, this work motivates searching for ICL in other non-linguistic modalities – time series (Das et al., 2024), system logs (Akhauri et al., 2025), physics simulations (Holzschuh et al., 2025), chess games (Ruoss et al., 2024), and climate projections (Duncan et al., 2025). Each offers a structured, patterned substrate that could support its own form of contextual reasoning. These diverse modalities, each with their unique structure and constraints, suggest a rich world of non-linguistic ICL capability waiting to be explored, and this work represents a maiden voyage into these extremely interesting waters.

Broader Impact Statement

Understanding ICL is pivotal for both the scientific study and practical control of LLMs. ICL ties to major efforts to interpret model behavior, characterize how reasoning and abstraction evolve with scale, and design mechanisms for controllability and steering. It also powers pragmatic applications such as synthetic data generation. By demonstrating that ICL arises even in non-linguistic settings, this work broadens the empirical foundation for studying ICL as a general computational phenomenon rather than a quirk of human-language training. A clearer mechanistic understanding of how ICL emerges and operates across modalities can inform how we build, guide, and evaluate large models – improving their reliability, controllability, and usefulness while reducing risks from misalignment or overgeneralization.

Acknowledgment

This work is supported by ONR grant (N0001424-1-2089). We acknowledge the use of computational resources on the DSAI cluster. We sincerely thank the JHU CLSP and DSAI communities for their helpful comments and feedback.

References

Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI communications*, 7(1):39–59, 1994. URL `https://www.iiia.csic.es/~enric/papers/AICom.pdf`.

Kwangjun Ahn, Xiang Cheng, Hadi Daneshmand, and Suvrit Sra. Transformers learn to implement preconditioned gradient descent for in-context learning. In *Advances in Neural Information Processing Systems* (NeurIPS), 2023. URL `http://papers.nips.cc/paper_files/paper/2023/hash/8ed3d610ea4b68e7afb30ea7d01422c6-Abstract-Conference.html`.

Yash Akhauri, Bryan Lewandowski, Cheng-Hsi Lin, Adrian N. Reyes, Grant C. Forbes, Arissa Wongpanich, Bangding Yang, Mohamed S. Abdelfattah, Sagi Perel, and Xingyou Song. Performance prediction for large systems via text-to-text regression, 2025. URL `https://arxiv.org/abs/2506.21718`.

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. In *International Conference on Learning Representations* (ICLR), 2022. URL `https://arxiv.org/abs/2211.15661`.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob L. Menick, Sebastian Borgeaud, Andy Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karén Simonyan. Flamingo: a visual language model for few-shot learning. In *Advances in Neural Information Processing Systems* (NeurIPS), 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/960a172bc7fbf0177ccccbb411a7d800-Abstract-Conference.html.

Abdul Fatir Ansari, Lorenzo Stella, Ali Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Hao Wang, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, and Yuyang Wang. Chronos: Learning the language of time series. *Transactions on Machine Learning Research*, November 2024. ISSN 2835-8856. doi: 10.48550/arXiv.2403.07815. URL https://openreview.net/forum?id=gerNCVqqtR. Published 11 Nov 2024.

Abdul Fatir Ansari, Oleksandr Shchur, Jaris Küken, Andreas Auer, Boran Han, Pedro Mercado, Syama Sundar Rangapuram, Huibin Shen, Lorenzo Stella, Xiyuan Zhang, Mononito Goswami, Shubham Kapoor, Danielle C. Maddix, Pablo Guerron, Tony Hu, Junming Yin, Nick Erickson, Prateek Mutalik Desai, Hao Wang, Huzefa Rangwala, George Karypis, Yuyang Wang, and Michael Bohlke-Schneider. Chronos-2: From univariate to universal forecasting. *arXiv preprint arXiv:2510.15821*, October 2025. doi: 10.48550/arXiv.2510.15821. URL https://arxiv.org/abs/2510.15821.

Arc Institute. Evo 2 1b base. https://huggingface.co/arcinstitute/evo2_1b_base, 2025a. Model card on Hugging Face.

Arc Institute. Evo 2 40b. https://huggingface.co/arcinstitute/evo2_40b, 2025b. Model card on Hugging Face.

Arc Institute. Evo 2 7b. https://huggingface.co/arcinstitute/evo2_7b, 2025c. Model card on Hugging Face.

Yutong Bai, Xinyang Geng, Karttikeya Mangalam, Amir Bar, Alan L Yuille, Trevor Darrell, Jitendra Malik, and Alexei A Efros. Sequential modeling enables scalable learning for large vision models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22861–22872, 2024. URL https://arxiv.org/pdf/2312.00785.

Hritik Bansal, Karthik Gopalakrishnan, Saket Dingliwal, Sravan Bodapati, Katrin Kirchhoff, and Dan Roth. Rethinking the role of scale for in-context learning: An interpretability-based case study at 66 billion scale. In *Annual Meeting of the Association for Computational Linguistics* (ACL), 2023. URL https://aclanthology.org/2023.acl-long.660/.

Gonzalo Benegas, Chengzhong Ye, Carlos Albors, Jianan Canal Li, and Yun S Song. Genomic language models: opportunities and challenges. *Trends in Genetics*, 2025.

Luca Bertinetto, João F. Henriques, Philip H. S. Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers. In *International Conference on Learning Representations* (ICLR), 2019. URL https://openreview.net/forum?id=HyxnZh0ct7.

Garyk Brixi, Matthew G. Durrant, Jerome Ku, Michael Poli, Greg Brockman, Daniel Chang, Gabriel A. Gonzalez, Samuel H. King, David B. Li, Aditi T. Merchant, Mohsen Naghipourfar, Eric Nguyen, Chiara Ricci-Tam, David W. Romero, Gwanggyu Sun, Ali Taghibakshi, Anton Vorontsov, Brandon Yang, Myra Deng, Liv Gorton, Nam Nguyen, Nicholas K. Wang, Etowah Adams, Stephen A. Baccus, Steven Dillmann, Stefano Ermon, Daniel Guo, Rajesh Ilango, Ken Janik, Amy X. Lu, Reshma Mehta, Mohammad R.K. Mofrad, Madelena Y. Ng, Jaspreet Pannu, Christopher Ré, Jonathan C. Schmok, John St. John, Jeremy Sullivan, Kevin Zhu, Greg Zynda, Daniel Balsam, Patrick Collison, Anthony B. Costa, Tina Hernandez-Boussard, Eric Ho, Ming-Yu Liu, Thomas McGrath, Kimberly Powell, Dave P. Burke, Hani Goodarzi, Patrick D. Hsu, and Brian L. Hie. Genome modeling and design across all domains of life with evo 2. *bioRxiv*, 2025. doi: 10.1101/2025.02.18.638918. URL https://www.biorxiv.org/content/early/2025/02/21/2025.02.18.638918.

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* (NeurIPS), 2020a. URL `https://arxiv.org/abs/2005.14165`.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901, 2020b. URL `https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html`.

Stephanie Chan, Adam Santoro, Andrew Lampinen, Jane Wang, Aaditya Singh, Pierre Richemond, James McClelland, and Felix Hill. Data distributional properties drive emergent in-context learning in transformers. *Advances in Neural Information Processing Systems* (NeurIPS), 2022. URL `https://arxiv.org/abs/2205.05055`.

Yanda Chen, Chen Zhao, Zhou Yu, Kathleen McKeown, and He He. Parallel structures in pre-training data yield in-context learning. *ArXiv preprint*, abs/2402.12530, 2024. URL `https://arxiv.org/abs/2402.12530`.

François Chollet. On the measure of intelligence. *arXiv preprint arXiv:1911.01547*, 2019.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. URL `https://arxiv.org/pdf/2110.14168`.

Haotian Cui, Chloe Wang, Hassaan Maan, Kuan Pang, Fengning Luo, Nan Duan, and Bo Wang. scGPT: toward building a foundation model for single-cell multi-omics using generative AI. *Nature methods*, 21(8):1470–1480, 2024.

Hugo Dalla-torre, Liam Gonzalez, Javier Mendoza-Revilla, Nicolas Lopez Carranza, Adam Henryk Grzywaczewski, Francesco Oteri, Christian Dallago, Evan Trop, Bernardo P. de Almeida, Hassan Sirelkhatim, Guillaume Richard, Marcin J. Skwark, Karim Beguir, Marie Lopez, and Thomas Pierrot. Nucleotide transformer: building and evaluating robust foundation models for human genomics. *Nature Methods*, 22:287 – 297, 2024. URL `https://api.semanticscholar.org/CorpusID:274369582`.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 10148–10167. PMLR, 21–27 Jul 2024. URL `https://proceedings.mlr.press/v235/das24c.html`.

James P. C. Duncan, Elynn Wu, Surya Dheeshjith, Adam Subel, Troy Arcomano, Spencer K. Clark, Brian Henn, Anna Kwa, Jeremy McGibbon, W. Andre Perkins, William Gregory, Carlos Fernandez-Granda, Julius Busecke, Oliver Watt-Meyer, William J. Hurlin, Alistair Adcroft, Laure Zanna, and Christopher Bretherton. Samudrace: Fast and accurate coupled climate modeling with 3d ocean and atmosphere emulators, 2025. URL `https://arxiv.org/abs/2509.12490`.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. A mathematical framework for transformer circuits.

Transformer Circuits Thread, 2021. URL https://transformer-circuits.pub/2021/framework/index.html.

Thomas Ellman. Explanation-based learning: A survey of programs and perspectives. *ACM Computing Surveys (CSUR)*, 21(2):163–221, 1989.

Eric Elmoznino, Thomas Jiralerspong, Yoshua Bengio, and Guillaume Lajoie. A complexity-based theory of compositionality. *ArXiv preprint*, abs/2410.14817, 2024a. URL https://arxiv.org/abs/2410.14817.

Eric Elmoznino, Tom Marty, Tejas Kasetty, Leo Gagnon, Sarthak Mittal, Mahan Fathi, Dhanya Sridhar, and Guillaume Lajoie. In-context learning and occam's razor. *ArXiv preprint*, abs/2410.14086, 2024b. URL https://arxiv.org/abs/2410.14086.

Zhouxiang Fang, Aayush Mishra, Muhan Gao, Anqi Liu, and Daniel Khashabi. ICL Ciphers: Quantifying "Learning" in In-Context Learning via Substitution Ciphers. In *Conference on Empirical Methods in Natural Language Processing* (EMNLP), 2025. URL https://arxiv.org/abs/2504.19395.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning* (ICML), 2017. URL http://proceedings.mlr.press/v70/finn17a.html.

Veniamin S. Fishman, Yuri Kuratov, Maxim Petrov, Aleksei Shmelev, Denis Shepelin, N. Chekanov, Olga L. Kardymon, and Mikhail S. Burtsev. Gena-lm: a family of open-source foundational dna language models for long sequences. *Nucleic Acids Research*, 53, 2024. URL https://api.semanticscholar.org/CorpusID:259166623.

Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes. *Advances in Neural Information Processing Systems* (NeurIPS), 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/c529dba08a146ea8d6cf715ae8930cbe-Abstract-Conference.html.

Dedre Gentner and Christian Hoyos. Analogy and abstraction. *Topics in cognitive science*, 9(3):672–693, 2017. URL https://doi.org/10.1111/tops.12278.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics* (TACL), 2021. URL https://arxiv.org/abs/2101.02235.

Riccardo Grazzi, Julien Siems, Simon Schrodi, Thomas Brox, and Frank Hutter. Is mamba capable of in-context learning? *ArXiv preprint*, abs/2402.03170, 2024. URL https://arxiv.org/abs/2402.03170.

Katarína Grešová, Vlastimil Martinek, David Čechák, Petr Šimeček, and Panagiotis Alexiou. Genomic benchmarks: a collection of datasets for genomic sequence classification. *BMC Genomic Data*, 24:25, May 2023. doi: 10.1186/s12863-023-01123-8.

Michael Hahn and Navin Goyal. A theory of emergent in-context learning as implicit structure induction. *arXiv preprint arXiv:2303.07971*, 2023. URL https://arxiv.org/abs/2303.07971.

Damian Hodel and Jevin West. Response: Emergent analogical reasoning in large language models. *arXiv preprint arXiv:2308.16118*, 2023.

Douglas R Hofstadter. Analogy as the core of cognition. *The analogical mind: Perspectives from cognitive science*, 2001. URL https://doi.org/10.7551/mitpress/1251.001.0001.

K Holyoak, Dedre Gentner, and B Kokinov. The place of analogy in cognition. *The analogical mind: Perspectives from cognitive science*, 119, 2001. URL https://doi.org/10.7551/mitpress/1251.001.0001.

Benjamin Holzschuh, Qiang Liu, Georg Kohl, and Nils Thuerey. PDE-transformer: Efficient and versatile transformers for physics simulations. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 23562–23602. PMLR, 13–19 Jul 2025. URL `https://proceedings.mlr.press/v267/holzschuh25a.html`.

Yanrong Ji, Zhihan Zhou, Han Liu, and Ramana V. Davuluri. Dnabert: pre-trained bidirectional encoder representations from transformers model for dna-language in genome. *Bioinformatics*, 37(15):2112–2120, 2021. doi: 10.1093/bioinformatics/btab083. URL `https://doi.org/10.1093/bioinformatics/btab083`.

Zhongtao Jiang, Yuanzhe Zhang, Kun Luo, Xiaowei Yuan, Jun Zhao, and Kang Liu. On the in-context generation of language models. In *Conference on Empirical Methods in Natural Language Processing* (EMNLP), 2024. doi: 10.18653/v1/2024.emnlp-main.568. URL `https://aclanthology.org/2024.emnlp-main.568/`.

Pranav Kantroo, Günter P. Wagner, and Benjamin B. Machta. In-context learning can distort the relationship between sequence likelihoods and biological fitness, 2025. URL `https://arxiv.org/abs/2504.17068`.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020. URL `https://arxiv.org/abs/2001.08361`.

Jonathan W Kim, Ahmed Alaa, and Danilo Bernardo. Eeg-gpt: exploring capabilities of large language models for eeg classification and interpretation. *ArXiv preprint*, abs/2401.18006, 2024. URL `https://arxiv.org/abs/2401.18006`.

Louis Kirsch, James Harrison, Jascha Sohl-Dickstein, and Luke Metz. General-purpose in-context learning by meta-learning transformers. *ArXiv preprint*, abs/2212.04458, 2022. URL `https://arxiv.org/abs/2212.04458`.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems* (NeurIPS), 2022. URL `https://arxiv.org/abs/2205.11916`.

Juliana S Lancaster and Janet L Kolodner. Problem solving in a natural task as a function of experience. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 9, 1987.

Ivan Lee, Nan Jiang, and Taylor Berg-Kirkpatrick. Exploring the relationship between model architecture and in-context learning ability. *ArXiv preprint*, abs/2310.08049, 2023. URL `https://arxiv.org/abs/2310.08049`.

Martha Lewis and Melanie Mitchell. Evaluating the robustness of analogical reasoning in GPT models. *Transactions on Machine Learning Research*, February 2025. URL `https://openreview.net/forum?id=t5cy5v9wph`.

Junnan Li, Dongxu Li, Caiming Xiong, and Steven C. H. Hoi. BLIP: bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning* (ICML), volume 162 of *Proceedings of Machine Learning Research*, 2022. URL `https://proceedings.mlr.press/v162/li22n.html`.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven C. H. Hoi. BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models. In *International Conference on Machine Learning* (ICML), volume 202 of *Proceedings of Machine Learning Research*, 2023a. URL `https://proceedings.mlr.press/v202/li23q.html`.

Shuai Li, Zhao Song, Yu Xia, Tong Yu, and Tianyi Zhou. The closeness of in-context learning and weight shifting for softmax regression. *arXiv preprint arXiv:2304.13276*, 2023b. URL https://arxiv.org/abs/2304.13276.

Yingcong Li, Muhammed Emrullah Ildiz, Dimitris Papailiopoulos, and Samet Oymak. Transformers as algorithms: Generalization and stability in in-context learning. In *International Conference on Machine Learning* (ICML), 2023c. URL https://proceedings.mlr.press/v202/li23l.html.

Ziqian Lin and Kangwook Lee. Dual operating modes of in-context learning. In *International Conference on Machine Learning* (ICML), 2024. URL https://arxiv.org/pdf/2402.18819.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems* (NeurIPS), 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/91edff07232fb1b55a505a9e9f6c0ff3-Abstract-Conference.html.

Arvind Mahankali, Tatsunori B Hashimoto, and Tengyu Ma. One step of gradient descent is provably the optimal in-context learner with one layer of linear self-attention. *ArXiv preprint*, abs/2307.03576, 2023. URL https://arxiv.org/abs/2307.03576.

Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. MetaICL: Learning to learn in context. In *Conference of the North American Chapter of the Association for Computational Linguistics* (NAACL), 2022. URL https://aclanthology.org/2022.naacl-main.201/.

Aayush Mishra, Daniel Khashabi, and Anqi Liu. Ia2: Alignment with icl activations improves supervised fine-tuning. *arXiv preprint arXiv:2509.22621*, 2025. URL https://arxiv.org/abs/2509.22621.

Omar Naim, Guilhem Fouilhé, and Nicholas Asher. Re-examining learning linear functions in context. *ArXiv preprint*, abs/2411.11465, 2024. URL https://arxiv.org/abs/2411.11465.

Ismail Nejjar, Faez Ahmed, and Olga Fink. Im-context: In-context learning for imbalanced regression tasks. *ArXiv preprint*, abs/2405.18202, 2024. URL https://arxiv.org/abs/2405.18202.

Ziv Nevo and Ran El-Yaniv. On online learning of decision lists. *J. Mach. Learn. Res.*, 3:271–301, 2002. URL https://jmlr.org/papers/v3/nevo02a.html.

Eric Nguyen, Michael Poli, Marjan Faizi, Armin Thomas, Michael Wornow, Callum Birch-Sykes, Stefano Massaroli, Aman Patel, Clayton Rabideau, Yoshua Bengio, et al. HyenaDNA: long-range genomic sequence modeling at single nucleotide resolution. *Advances in neural information processing systems*, 36:43177–43201, 2023.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, Charles Sutton, and Augustus Odena. Show your work: Scratchpads for intermediate computation with language models. 2021. URL https://arxiv.org/abs/2112.00114.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. In-context learning and induction heads, 2022a. URL https://arxiv.org/abs/2209.11895.

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*, 2022b. URL `https://arxiv.org/abs/2209.11895`.

Jane Pan, Tianyu Gao, Howard Chen, and Danqi Chen. What in-context learning "learns" in-context: Disentangling task recognition and task learning. In *Findings of the Association for Computational Linguistics: ACL 2023*, July 2023. URL `https://aclanthology.org/2023.findings-acl.527`.

Madhur Panwar, Kabir Ahuja, and Navin Goyal. In-context learning through the bayesian prism. *ArXiv preprint*, abs/2306.04891, 2023. URL `https://arxiv.org/abs/2306.04891`.

Jongho Park, Jaeseung Park, Zheyang Xiong, Nayoung Lee, Jaewoong Cho, Samet Oymak, Kangwook Lee, and Dimitris Papailiopoulos. Can mamba learn how to learn? a comparative study on in-context learning tasks. *ArXiv preprint*, abs/2402.04248, 2024. URL `https://arxiv.org/abs/2402.04248`.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018. URL `https://openai.com/index/language-unsupervised/`.

John Carlyle Raven, John H. Court, and John Carlyle Raven. Manual for raven's progressive matrices and vocabulary scales. 1962. URL `https://api.semanticscholar.org/CorpusID:143337389`.

Allan Raventós, Mansheej Paul, Feng Chen, and Surya Ganguli. Pretraining task diversity and the emergence of non-bayesian in-context learning for regression. In *Advances in Neural Information Processing Systems* (NeurIPS), 2023. URL `http://papers.nips.cc/paper_files/paper/2023/hash/2e10b2c2e1aa4f8083c37dfe269873f8-Abstract-Conference.html`.

Gautam Reddy. The mechanistic basis of data dependence and abrupt learning in an in-context classification task. In *International Conference on Learning Representations* (ICLR), 2023. URL `https://arxiv.org/pdf/2312.03002`.

Jie Ren, Qipeng Guo, Hang Yan, Dongrui Liu, Quanshi Zhang, Xipeng Qiu, and Dahua Lin. Identifying semantic induction heads to understand in-context learning. *arXiv preprint arXiv:2402.13055*, 2024.

Brian H Ross. Remindings and their effects in learning a cognitive skill. *Cognitive psychology*, 16(3):371–416, 1984.

Anian Ruoss, Grégoire Delétang, Sourabh Medapati, Jordi Grau-Moya, Li Kevin Wenliang, Elliot Catt, John Reid, Cannada A. Lewis, Joel Veness, and Tim Genewein. Amortized planning with large-scale transformers: A case study on chess. In *Advances in Neural Information Processing Systems*, volume 37, pp. 65765–65790, 2024. doi: 10.52202/079017-2102. URL `https://proceedings.neurips.cc/paper_files/paper/2024/hash/78f0db30c39c850de728c769f42fc903-Abstract-Conference.html`.

Niklas Schmidinger, Lisa Schneckenreiter, Philipp Seidl, Johannes Schimunek, Pieter-Jan Hoedt, Johannes Brandstetter, Andreas Mayr, Sohvi Luukkonen, Sepp Hochreiter, and Günter Klambauer. Bio-xlstm: Generative modeling, representation and in-context learning of biological and chemical sequences, 2024. URL `https://arxiv.org/abs/2411.04165`.

Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. Synthetic prompting: Generating chain-of-thought demonstrations for large language models. In *International Conference on Machine Learning*, pp. 30706–30775. PMLR, 2023.

Lingfeng Shen, Aayush Mishra, and Daniel Khashabi. Do pretrained transformers learn in-context by gradient descent? In *International Conference on Machine Learning* (ICML), 2024. URL `https://arxiv.org/abs/2310.08540`.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, Agnieszka Kluska, Aitor Lewkowycz, Akshat Agarwal, Alethea Power, Alex Ray, Alex Warstadt, Alexander W. Kocurek, Ali Safaya, Ali Tazarv, Alice Xiang, Alicia Parrish, Allen Nie, Aman Hussain, Amanda Askell, Amanda Dsouza, Ambrose Slone, Ameet Rahane, Anantharaman S. Iyer, Anders Andreassen, Andrea Madotto, Andrea Santilli, Andreas Stuhlmüller, Andrew Dai, Andrew La, Andrew Lampinen, Andy Zou, Angela Jiang, Angelica Chen, Anh Vuong, Animesh Gupta, Anna Gottardi, Antonio Norelli, Anu Venkatesh, Arash Gholamidavoodi, Arfa Tabassum, Arul Menezes, Arun Kirubarajan, Asher Mullokandov, Ashish Sabharwal, Austin Herrick, Avia Efrat, Aykut Erdem, Ayla Karakaş, B. Ryan Roberts, Bao Sheng Loe, Barret Zoph, Bartłomiej Bojanowski, Batuhan Özyurt, Behnam Hedayatnia, Behnam Neyshabur, Benjamin Inden, Benno Stein, Berk Ekmekci, Bill Yuchen Lin, Blake Howald, Bryan Orinion, Cameron Diao, Cameron Dour, Catherine Stinson, Cedrick Argueta, César Ferri Ramírez, Chandan Singh, Charles Rathkopf, Chenlin Meng, Chitta Baral, Chiyu Wu, Chris Callison-Burch, Chris Waites, Christian Voigt, Christopher D. Manning, Christopher Potts, Cindy Ramirez, Clara E. Rivera, Clemencia Siro, Colin Raffel, Courtney Ashcraft, Cristina Garbacea, Damien Sileo, Dan Garrette, Dan Hendrycks, Dan Kilman, Dan Roth, Daniel Freeman, Daniel Khashabi, Daniel Levy, Daniel Moseguí González, Danielle Perszyk, Danny Hernandez, Danqi Chen, Daphne Ippolito, Dar Gilboa, David Dohan, David Drakard, David Jurgens, Debajyoti Datta, Deep Ganguli, Denis Emelin, Denis Kleyko, Deniz Yuret, Derek Chen, Derek Tam, Dieuwke Hupkes, Diganta Misra, Dilyar Buzan, Dimitri Coelho Mollo, Diyi Yang, Dong-Ho Lee, Dylan Schrader, Ekaterina Shutova, Ekin Dogus Cubuk, Elad Segal, Eleanor Hagerman, Elizabeth Barnes, Elizabeth Donoway, Ellie Pavlick, Emanuele Rodola, Emma Lam, Eric Chu, Eric Tang, Erkut Erdem, Ernie Chang, Ethan A. Chi, Ethan Dyer, Ethan Jerzak, Ethan Kim, Eunice Engefu Manyasi, Evgenii Zheltonozhskii, Fanyue Xia, Fatemeh Siar, Fernando Martínez-Plumed, Francesca Happé, Francois Chollet, Frieda Rong, Gaurav Mishra, Genta Indra Winata, Gerard de Melo, Germán Kruszewski, Giambattista Parascandolo, Giorgio Mariani, Gloria Wang, Gonzalo Jaimovitch-López, Gregor Betz, Guy Gur-Ari, Hana Galijasevic, Hannah Kim, Hannah Rashkin, Hannaneh Hajishirzi, Harsh Mehta, Hayden Bogar, Henry Shevlin, Hinrich Schütze, Hiromu Yakura, Hongming Zhang, Hugh Mee Wong, Ian Ng, Isaac Noble, Jaap Jumelet, Jack Geissinger, Jackson Kernion, Jacob Hilton, Jaehoon Lee, Jaime Fernández Fisac, James B. Simon, James Koppel, James Zheng, James Zou, Jan Kocoń, Jana Thompson, Janelle Wingfield, Jared Kaplan, Jarema Radom, Jascha Sohl-Dickstein, Jason Phang, Jason Wei, Jason Yosinski, Jekaterina Novikova, Jelle Bosscher, Jennifer Marsh, Jeremy Kim, Jeroen Taal, Jesse Engel, Jesujoba Alabi, Jiacheng Xu, Jiaming Song, Jillian Tang, Joan Waweru, John Burden, John Miller, John U. Balis, Jonathan Batchelder, Jonathan Berant, Jörg Frohberg, Jos Rozen, Jose Hernandez-Orallo, Joseph Boudeman, Joseph Guerr, Joseph Jones, Joshua B. Tenenbaum, Joshua S. Rule, Joyce Chua, Kamil Kanclerz, Karen Livescu, Karl Krauth, Karthik Gopalakrishnan, Katerina Ignatyeva, Katja Markert, Kaustubh D. Dhole, Kevin Gimpel, Kevin Omondi, Kory Mathewson, Kristen Chiafullo, Ksenia Shkaruta, Kumar Shridhar, Kyle McDonell, Kyle Richardson, Laria Reynolds, Leo Gao, Li Zhang, Liam Dugan, Lianhui Qin, Lidia Contreras-Ochando, Louis-Philippe Morency, Luca Moschella, Lucas Lam, Lucy Noble, Ludwig Schmidt, Luheng He, Luis Oliveros Colón, Luke Metz, Lütfi Kerem Şenel, Maarten Bosma, Maarten Sap, Maartje ter Hoeve, Maheen Farooqi, Manaal Faruqui, Mantas Mazeika, Marco Baturan, Marco Marelli, Marco Maru, Maria Jose Ramírez Quintana, Marie Tolkiehn, Mario Giulianelli, Martha Lewis, Martin Potthast, Matthew L. Leavitt, Matthias Hagen, Mátyás Schubert, Medina Orduna Baitemirova, Melody Arnaud, Melvin McElrath, Michael A. Yee, Michael Cohen, Michael Gu, Michael Ivanitskiy, Michael Starritt, Michael Strube, Michał Swedrowski, Michele Bevilacqua, Michihiro Yasunaga, Mihir Kale, Mike Cain, Mimee Xu, Mirac Suzgun, Mitch Walker, Mo Tiwari, Mohit Bansal, Moin Aminnaseri, Mor Geva, Mozhdeh Gheini, Mukund Varma T, Nanyun Peng, Nathan A. Chi, Nayeon Lee, Neta Gur-Ari Krakover, Nicholas Cameron, Nicholas Roberts, Nick Doiron, Nicole Martinez, Nikita Nangia, Niklas Deckers, Niklas Muennighoff, Nitish Shirish Keskar, Niveditha S. Iyer, Noah Constant, Noah Fiedel, Nuan Wen, Oliver Zhang, Omar Agha, Omar Elbaghdadi, Omer Levy, Owain Evans, Pablo Antonio Moreno Casares, Parth Doshi, Pascale Fung, Paul Pu Liang, Paul Vicol, Pegah

Alipoormolabashi, Peiyuan Liao, Percy Liang, Peter Chang, Peter Eckersley, Phu Mon Htut, Pinyu Hwang, Piotr Miłkowski, Piyush Patil, Pouya Pezeshkpour, Priti Oli, Qiaozhu Mei, Qing Lyu, Qinlang Chen, Rabin Banjade, Rachel Etta Rudolph, Raefer Gabriel, Rahel Habacker, Ramon Risco, Raphaël Millière, Rhythm Garg, Richard Barnes, Rif A. Saurous, Riku Arakawa, Robbe Raymaekers, Robert Frank, Rohan Sikand, Roman Novak, Roman Sitelew, Ronan LeBras, Rosanne Liu, Rowan Jacobs, Rui Zhang, Ruslan Salakhutdinov, Ryan Chi, Ryan Lee, Ryan Stovall, Ryan Teehan, Rylan Yang, Sahib Singh, Saif M. Mohammad, Sajant Anand, Sam Dillavou, Sam Shleifer, Sam Wiseman, Samuel Gruetter, Samuel R. Bowman, Samuel S. Schoenholz, Sanghyun Han, Sanjeev Kwatra, Sarah A. Rous, Sarik Ghazarian, Sayan Ghosh, Sean Casey, Sebastian Bischoff, Sebastian Gehrmann, Sebastian Schuster, Sepideh Sadeghi, Shadi Hamdan, Sharon Zhou, Shashank Srivastava, Sherry Shi, Shikhar Singh, Shima Asaadi, Shixiang Shane Gu, Shubh Pachchigar, Shubham Toshniwal, Shyam Upadhyay, Shyamolima, Debnath, Siamak Shakeri, Simon Thormeyer, Simone Melzi, Siva Reddy, Sneha Priscilla Makini, Soo-Hwan Lee, Spencer Torene, Sriharsha Hatwar, Stanislas Dehaene, Stefan Divic, Stefano Ermon, Stella Biderman, Stephanie Lin, Stephen Prasad, Steven T. Piantadosi, Stuart M. Shieber, Summer Misherghi, Svetlana Kiritchenko, Swaroop Mishra, Tal Linzen, Tal Schuster, Tao Li, Tao Yu, Tariq Ali, Tatsu Hashimoto, Te-Lin Wu, Théo Desbordes, Theodore Rothschild, Thomas Phan, Tianle Wang, Tiberius Nkinyili, Timo Schick, Timofei Kornev, Titus Tunduny, Tobias Gerstenberg, Trenton Chang, Trishala Neeraj, Tushar Khot, Tyler Shultz, Uri Shaham, Vedant Misra, Vera Demberg, Victoria Nyamai, Vikas Raunak, Vinay Ramasesh, Vinay Uday Prabhu, Vishakh Padmakumar, Vivek Srikumar, William Fedus, William Saunders, William Zhang, Wout Vossen, Xiang Ren, Xiaoyu Tong, Xinran Zhao, Xinyi Wu, Xudong Shen, Yadollah Yaghoobzadeh, Yair Lakretz, Yangqiu Song, Yasaman Bahri, Yejin Choi, Yichi Yang, Yiding Hao, Yifu Chen, Yonatan Belinkov, Yu Hou, Yufang Hou, Yuntao Bai, Zachary Seid, Zhuoye Zhao, Zijian Wang, Zijie J. Wang, Zirui Wang, and Ziyi Wu. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research* (TMLR), 2023. URL `https://arxiv.org/abs/2206.04615`.

Claire E. Stevenson, Alexandra Pafford, Han L. J. van der Maas, and Melanie Mitchell. Can large language models generalize analogy solving like children can?, 2025. URL `https://arxiv.org/abs/2411.02348`.

Quan Sun, Qiying Yu, Yufeng Cui, Fan Zhang, Xiaosong Zhang, Yueze Wang, Hongcheng Gao, Jingjing Liu, Tiejun Huang, and Xinlong Wang. Generative pretraining in multimodality. *ArXiv preprint*, abs/2307.05222, 2023. URL `https://arxiv.org/abs/2307.05222`.

Quan Sun, Yufeng Cui, Xiaosong Zhang, Fan Zhang, Qiying Yu, Yueze Wang, Yongming Rao, Jingjing Liu, Tiejun Huang, and Xinlong Wang. Generative multimodal models are in-context learners. In *IEEE Conference on Computer Vision and Pattern Recognition* (CVPR), 2024. URL `https://openaccess.thecvf.com/content/CVPR2024/html/Sun_Generative_Multimodal_Models_are_In-Context_Learners_CVPR_2024_paper.html`.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero, Alexander M. Rush, and Thomas Wolf. Zephyr: Direct distillation of lm alignment. *arXiv preprint 2310.16944*, 2023. URL `https://arxiv.org/abs/2310.16944`.

Ramzan Kh. Umarov and Victor V. Solovyev. Recognition of prokaryotic and eukaryotic promoters using convolutional deep learning neural networks. *PLOS ONE*, 12(2):e0171410, 2017. doi: 10.1371/journal.pone.0171410. URL `https://doi.org/10.1371/journal.pone.0171410`.

Lean Wang, Lei Li, Damai Dai, Deli Chen, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. Label words are anchors: An information flow perspective for understanding in-context learning. In *Conference on Empirical Methods in Natural Language Processing* (EMNLP), 2023a. doi: 10.18653/v1/2023.emnlp-main.609. URL `https://aclanthology.org/2023.emnlp-main.609`.

Xiaolei Wang, Xinyu Tang, Wayne Xin Zhao, and Ji-Rong Wen. Investigating the pre-training dynamics of in-context learning: Task recognition vs. task learning. *ArXiv preprint*, abs/2406.14022, 2024. URL `https://arxiv.org/abs/2406.14022`.

Xinyi Wang, Wanrong Zhu, Michael Saxon, Mark Steyvers, and William Yang Wang. Large language models are latent variable models: Explaining and finding good demonstrations for in-context learning. In *Advances in Neural Information Processing Systems* (NeurIPS), 2023b. URL `http://papers.nips.cc/paper_files/paper/2023/hash/3255a7554605a88800f4e120b3a929e1-Abstract-Conference.html`.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-Instruct: Aligning Language Model with Self Generated Instructions. In *Annual Meeting of the Association for Computational Linguistics* (ACL), 2023c. URL `https://arxiv.org/abs/2212.10560`.

Taylor Webb, Keith J Holyoak, and Hongjing Lu. Emergent analogical reasoning in large language models. *Nature Human Behaviour*, 7(9):1526–1541, 2023. URL `https://arxiv.org/abs/2212.09196`.

Taylor W Webb, Keith J Holyoak, and Hongjing Lu. Evidence from counterfactual tasks supports emergent analogical reasoning in large language models. *PNAS nexus*, 4(5): pgaf135, 2025.

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* (NeurIPS), 2022. URL `https://arxiv.org/abs/2201.11903`.

Kevin Christian Wibisono and Yixin Wang. In-context learning from training on unstructured data: The role of co-occurrence, positional information, and training data structure. In *ICML 2024 Workshop on Theoretical Foundations of Foundation Models*, 2024. URL `https://openreview.net/forum?id=Zvwwnfwxa4`.

Zhaofeng Wu, Robert L Logan IV, Pete Walsh, Akshita Bhagia, Dirk Groeneveld, Sameer Singh, and Iz Beltagy. Continued pretraining for better zero-and few-shot promptability. In *Conference on Empirical Methods in Natural Language Processing* (EMNLP), pp. 4517–4531, 2022. URL `https://arxiv.org/pdf/2210.10258`.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit bayesian inference. In *International Conference on Learning Representations* (ICLR), 2021. URL `https://arxiv.org/abs/2111.02080`.

Steve Yadlowsky, Lyric Doshi, and Nilesh Tripuraneni. Pretraining data mixtures enable narrow model selection capabilities in transformer models. *arXiv preprint arXiv:2311.00871*, 2023. URL `https://arxiv.org/abs/2311.00871`.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL `https://arxiv.org/abs/2505.09388`.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations* (ICLR), 2023. URL `https://openreview.net/pdf?id=WE_vluYUL-X`.

Michihiro Yasunaga, Xinyun Chen, Yujia Li, Panupong Pasupat, Jure Leskovec, Percy Liang, Ed H. Chi, and Denny Zhou. Large language models as analogical reasoners, 2023. URL `https://arxiv.org/abs/2310.01714`.

Fred Zhang, Jiaxin Ye, and Zhuoran Yang. In-context multi-armed bandits via supervised pretraining. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*, 2023. URL `https://openreview.net/forum?id=5IHOpideQK`.

Luisa M. Zintgraf, Kyriacos Shiarlis, Vitaly Kurin, Katja Hofmann, and Shimon Whiteson. Fast context adaptation via meta-learning. In *International Conference on Machine Learning* (ICML), 2019. URL `http://proceedings.mlr.press/v97/zintgraf19a.html`.

A Further Details on Synthetic Task Definition

A.1 Table of Primitives

The following table describes the thirty primitives used to construct the task space via composition – their use is described in §3.2.

Primitive	**Description**
`alternating_start_one`	Produce a mask marking positions that differ from the alternating 1010... pattern starting with 1 (1 = mismatch, 0 = match).
`alternating_start_zero`	Produce a mask marking positions that differ from the alternating 0101... pattern starting with 0 (1 = mismatch, 0 = match).
`center_mask`	Zero out the first and last bits while leaving the interior bits unchanged; strings of length ≤ 2 become all zeros.
`double_rotl`	Circularly rotate the bitstring two positions to the left.
`double_rotr`	Circularly rotate the bitstring two positions to the right.
`edge_mask`	Preserve the first and last bits and zero out every interior bit (length 0/1 strings pass through).
`flip_bits`	Invert every bit, swapping 0s for 1s and vice versa.
`identity`	Return the bitstring unchanged.
`invert_prefix`	Flip the bits in the left half of the string, keep the right half as is.
`invert_suffix`	Keep the left half as is and flip every bit in the right half of the string.
`keep_even_positions`	Keep bits at even indices (0-based) and zero out bits at odd indices.
`keep_odd_positions`	Keep bits at odd indices (0-based) and zero out bits at even indices.
`left_half`	Preserve the left half of the string and replace the right half with zeros.
`majority`	Fill the string with the majority bit from the input; ties resolve to all 1s.
`meta_constant`	Returns a random, pre-set constant.
`minority`	Fill the string with the minority bit from the input; ties resolve to all 0s.
`mirror_half`	Copy the left half of the string onto the right half in reverse order, keeping the center bit unchanged for odd lengths.
`ones_if_palindrome`	Output all 1s if the input is a palindrome; otherwise output all 0s.
`parity_fill`	Output all 1s when the input contains an odd number of 1s; otherwise output all 0s.
`reverse_bits`	Reverse the order of the bits in the string.
`right_half`	Zero out the left half and keep the right half unchanged.
`rotl1`	Circularly rotate the bitstring one position to the left.
`rotr1`	Circularly rotate the bitstring one position to the right.
`shift_left_zero`	Shift the string left by one, dropping the first bit and appending a 0 on the right.
`shift_right_zero`	Shift the string right by one, inserting a 0 on the left and dropping the last bit.
`spread_first_bit`	Replace every position with the first bit of the input.
`spread_last_bit`	Replace every position with the last bit of the input.
`swap_halves`	Swap the left and right halves of the string.
`swap_pairs`	Swap each adjacent pair of bits (positions 0/1, 2/3, ...).
`xor_with_s0`	Can only be applied after another primitive. Computes the logical XOR between the original input s_0 and the output of the first primitive.

Table 1: The 30 unary primitives used to construct functions in F.

A.2 Table of Functions

The following table describes the one hundred specific compositions of primitives used to construct the evaluation suite in §3.2.

Function	**Function**	**Function**
identity	spread_last_bit	swap_halves → shift_left_zero
rotl1	invert_prefix	swap_halves → shift_right_zero
reverse_bits	invert_suffix	shift_left_zero → swap_halves
flip_bits	meta_constant	shift_right_zero → swap_halves
swap_halves	flip_bits → reverse_bits	keep_even_positions → flip_bits
majority	rotl1 → reverse_bits	keep_odd_positions → flip_bits
minority	reverse_bits → rotl1	flip_bits → keep_even_positions
parity_fill	rotl1 → flip_bits	flip_bits → keep_odd_positions
alternating_start_one	swap_halves → reverse_bits	edge_mask → flip_bits
alternating_start_zero	swap_halves → flip_bits	center_mask → flip_bits
left_half	double_rotl → flip_bits	shift_left_zero → keep_even_positions
right_half	rotr1 → flip_bits	shift_left_zero → keep_odd_positions
double_rotl	spread_first_bit → flip_bits	shift_right_zero → keep_even_positions
rotr1	spread_last_bit → flip_bits	shift_right_zero → keep_odd_positions
double_rotr	left_half → flip_bits	keep_even_positions → reverse_bits
ones_if_palindrome	right_half → flip_bits	keep_odd_positions → reverse_bits
mirror_half	flip_bits → left_half	shift_left_zero → parity_fill
spread_first_bit	flip_bits → right_half	shift_right_zero → parity_fill
spread_last_bit	double_rotl → reverse_bits	parity_fill → shift_left_zero
invert_prefix	rotl1 → swap_halves	parity_fill → shift_right_zero
invert_suffix	xor_with_s0	spread_first_bit → shift_left_zero
meta_constant	flip_bits → xor_with_s0	spread_last_bit → shift_right_zero
shift_left_zero	ones_if_palindrome → flip_bits	spread_first_bit → keep_even_positions
shift_right_zero	flip_bits → mirror_half	spread_last_bit → keep_odd_positions
swap_pairs	invert_prefix → reverse_bits	spread_first_bit → edge_mask
keep_even_positions	left_half → reverse_bits	spread_last_bit → edge_mask
keep_odd_positions	right_half → reverse_bits	spread_first_bit → center_mask
edge_mask	parity_fill → flip_bits	spread_last_bit → center_mask
center_mask	rotl1 → spread_first_bit	rotl1 → shift_left_zero
xor_with_s0	shift_left_zero → flip_bits	rotl1 → shift_right_zero
flip_bits → reverse_bits	shift_right_zero → flip_bits	shift_left_zero → rotl1
rotl1 → reverse_bits	flip_bits → shift_left_zero	shift_right_zero → rotl1
reverse_bits → rotl1	flip_bits → shift_right_zero	reverse_bits → edge_mask
rotl1 → flip_bits	swap_pairs → flip_bits	reverse_bits → center_mask
swap_halves → reverse_bits	shift_left_zero → reverse_bits	edge_mask → shift_left_zero
swap_halves → flip_bits	shift_right_zero → reverse_bits	edge_mask → shift_right_zero
double_rotl → flip_bits	spread_first_bit → flip_bits	shift_left_zero → shift_left_zero
rotr1 → flip_bits	shift_left_zero → edge_mask	shift_left_zero → swap_pairs

Table 2: The complete set of 100 functions in F, consisting of 30 single primitives and 70 composed functions ($(f \to g)(x) = g(f(x))$).

A.3 BitLoad of Functions

Function / Composition	**BitLoad**	**Function / Composition**	**BitLoad**
identity	8	rotl1	8
reverse_bits	8	flip_bits	8
swap_halves	8	majority	8
minority	8	parity_fill	8
alternating_start_one	8	alternating_start_zero	8
left_half	4	right_half	4
double_rotl	8	rotr1	8
double_rotr	8	ones_if_palindrome	8
mirror_half	4	spread_first_bit	1
spread_last_bit	1	invert_prefix	8
invert_suffix	8	meta_constant	0
flip_bits → reverse_bits	8	rotl1 → reverse_bits	8
reverse_bits → rotl1	8	rotl1 → flip_bits	8
swap_halves → reverse_bits	8	swap_halves → flip_bits	8
double_rotl → flip_bits	8	rotr1 → flip_bits	8
spread_first_bit → flip_bits	1	spread_last_bit → flip_bits	1
left_half → flip_bits	4	right_half → flip_bits	4
flip_bits → left_half	4	flip_bits → right_half	4
double_rotl → reverse_bits	8	rotl1 → swap_halves	8
xor_with_s0	0	flip_bits → xor_with_s0	0
ones_if_palindrome → flip_bits	8	flip_bits → mirror_half	4
invert_prefix → reverse_bits	8	left_half → reverse_bits	4
right_half → reverse_bits	4	parity_fill → flip_bits	8
rotl1 → spread_first_bit	1	shift_left_zero	7
shift_right_zero	7	swap_pairs	8
keep_even_positions	4	keep_odd_positions	4
edge_mask	2	center_mask	6
shift_left_zero → flip_bits	7	shift_right_zero → flip_bits	7
flip_bits → shift_left_zero	7	flip_bits → shift_right_zero	7
swap_pairs → flip_bits	8	shift_left_zero → reverse_bits	7
shift_right_zero → reverse_bits	7	swap_halves → shift_left_zero	7
swap_halves → shift_right_zero	7	shift_left_zero → swap_halves	7
shift_right_zero → swap_halves	7	keep_even_positions → flip_bits	4
keep_odd_positions → flip_bits	4	flip_bits → keep_even_positions	4
flip_bits → keep_odd_positions	4	edge_mask → flip_bits	2
center_mask → flip_bits	6	shift_left_zero → keep_even_positions	4
shift_left_zero → keep_odd_positions	3	shift_right_zero → keep_even_positions	3
shift_right_zero → keep_odd_positions	4	keep_even_positions → reverse_bits	4
keep_odd_positions → reverse_bits	4	shift_left_zero → parity_fill	7
shift_right_zero → parity_fill	7	parity_fill → shift_left_zero	8
parity_fill → shift_right_zero	8	spread_first_bit → shift_left_zero	1
spread_last_bit → shift_right_zero	1	spread_first_bit → keep_even_positions	1
spread_last_bit → keep_odd_positions	1	spread_first_bit → edge_mask	1
spread_last_bit → edge_mask	1	spread_first_bit → center_mask	1
spread_last_bit → center_mask	1	rotl1 → shift_left_zero	7
rotl1 → shift_right_zero	7	shift_left_zero → rotl1	7
shift_right_zero → rotl1	7	reverse_bits → edge_mask	2
reverse_bits → center_mask	6	edge_mask → shift_left_zero	1
edge_mask → shift_right_zero	1	shift_left_zero → shift_left_zero	6
shift_left_zero → swap_pairs	7	shift_left_zero → edge_mask	1

Table 3: BitLoad for every primitive and composed function in F. Used in §4.3.

B A Global Metric Over All Programs

For a given function f, model M, and number of in-context examples n, we define the average accuracy over all context sets $E_N = \{E \subset S : |E| = n\}$:

$$A_f(M, n) = \frac{1}{|E_N|(|S| - n)} \sum_{E \in E_N} \sum_{x \in S \setminus E} A(f, E, x, M).$$

The overall benchmark score across all functions F is then

$$P(M, n) = \frac{1}{|F|} \sum_{f \in F} A_f(M, n).$$

Because exact evaluation is intractable (for $n = 8$, $|E_N|(|S| - N) = \binom{|S|}{n}(|S| - n) = \binom{256}{8} \cdot 248 \approx 1.016 \times 10^{17}$) we estimate $A_f(M, n)$ via Monte Carlo, as discussed in §3.4.

C Full Accuracy Results

Here we show the full table of accuracies used in §4.1 to analyze the ICL capabilities of Evo2 and Qwen3 and perform the necessary statistical tests.

Model	1 Shot	2 Shots	4 Shots	8 Shots	16 Shots	32 Shots	64 Shots	128 Shots
Qwen3 0.6B	$13.4_{\pm 2.3}$	$14.1_{\pm 2.3}$	$16.0_{\pm 2.6}$	$19.5_{\pm 2.7}$	$16.1_{\pm 2.4}$	$16.8_{\pm 2.5}$	$19.1_{\pm 2.7}$	$23.6_{\pm 2.9}$
Qwen3 1.7B	$13.1_{\pm 2.2}$	$15.0_{\pm 2.7}$	$\mathbf{17.2}_{\pm 2.8}$	$16.4_{\pm 2.6}$	$16.6_{\pm 2.7}$	$20.4_{\pm 2.9}$	$24.5_{\pm 3.4}$	$28.5_{\pm 3.5}$
Qwen3 4B	$13.5_{\pm 2.4}$	$15.2_{\pm 2.4}$	$17.0_{\pm 2.6}$	$15.4_{\pm 2.5}$	$18.4_{\pm 2.7}$	$24.8_{\pm 3.2}$	$24.5_{\pm 3.4}$	$29.4_{\pm 3.4}$
Qwen3 8B	$13.5_{\pm 2.3}$	$\mathbf{15.4}_{\pm 2.5}$	$16.9_{\pm 2.7}$	$19.6_{\pm 2.7}$	$20.6_{\pm 2.8}$	$\mathbf{27.9}_{\pm 3.3}$	$30.9_{\pm 3.5}$	$32.2_{\pm 3.5}$
Qwen3 14B	$\mathbf{14.0}_{\pm 2.4}$	$14.1_{\pm 2.4}$	$16.8_{\pm 2.7}$	$\mathbf{20.6}_{\pm 2.9}$	$\mathbf{22.9}_{\pm 3.1}$	$25.5_{\pm 3.2}$	$\mathbf{32.6}_{\pm 3.7}$	$\mathbf{33.8}_{\pm 3.5}$
Evo2 1B	$13.1_{\pm 2.2}$	$15.0_{\pm 2.4}$	$17.1_{\pm 2.9}$	$18.2_{\pm 2.7}$	$22.0_{\pm 2.9}$	$23.8_{\pm 2.8}$	$26.5_{\pm 3.0}$	$30.8_{\pm 3.1}$
Evo2 7B	$14.5_{\pm 2.4}$	$\mathbf{15.8}_{\pm 2.7}$	$\mathbf{17.6}_{\pm 2.9}$	$\mathbf{22.6}_{\pm 2.9}$	$25.2_{\pm 3.0}$	$\mathbf{29.8}_{\pm 3.1}$	$32.2_{\pm 3.1}$	$41.0_{\pm 3.4}$
Evo2 40B	$\mathbf{15.0}_{\pm 2.6}$	$15.5_{\pm 2.5}$	$\mathbf{17.6}_{\pm 2.8}$	$21.6_{\pm 3.1}$	$\mathbf{26.4}_{\pm 3.1}$	$28.9_{\pm 3.1}$	$\mathbf{34.5}_{\pm 3.2}$	$\mathbf{41.1}_{\pm 3.3}$

Table 4: In-context learning performance across model families and shot counts. Values show accuracy ± standard error. All numbers are percentages. **Bold** numbers show the best performance within a model family. Models are ordered by parameter count within each family.

D Meta-Regression for ICL Efficacy with Number of Demonstrations

We perform linear regressions to predict accuracy from shot count with each model. For each model M, fit the linear regression: $\hat{P}(M, n) = \alpha_0(M) + \alpha_1(M)\log(n) + \varepsilon$. The raw regressions are shown in Fig. 5a. Predictably, all α_1 are positive as all models are capable of learning in-context.

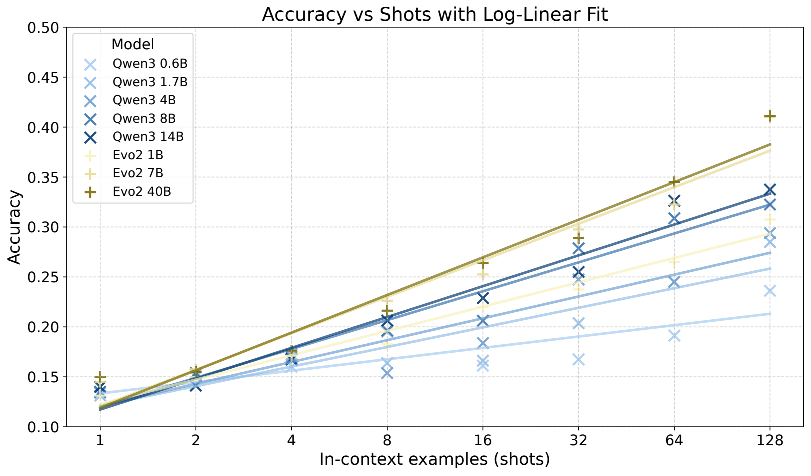
We can interpret α_0 as representing the model's base accuracy at the task, what it would logically achieve with only one shot to identify the task. We can then interpret α_1 as the model's ICL efficacy: the speed at which it adapts to the task being presented and at which its accuracy improves. Analyzing how these values change across parameter values reveals insights into the ICL abilities of both the Qwen3 and Evo2 models.

First, we analyze how α_0 changes in each model family – this analysis can be seen in Fig. 5b. Both Evo2's and Qwen3's initial α_0 remains essentially constant model-to-model, indicating that all models have similar levels of few-shot baseline performance. Notably, Evo2 and Qwen3 have essentially identical intercepts at around 0.12. This implies that despite drastically different training data, the overall amount of prior knowledge the models have coming into this task is roughly similar. This rules out Qwen simply having 'less experience' with this sort of task.

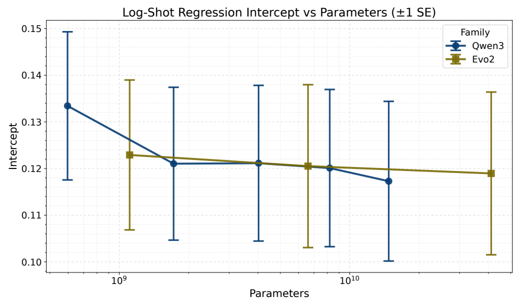
If one looks at α_1 – ICL efficacy – in Fig. 5c, a dramatically different picture is painted. Here, both Qwen3 and Evo2 follow similar patterns with a significant difference. Qwen3's ICL efficacy increases monotonically with respect to parameters, more than doubling from the 0.6B to the 14B. Evo2 follows suit (albeit less dramatically), with ICL efficacy monotonically increasing from the 1B to the 40B.

In absolute terms, however, Evo2, when parameter-matched, adapts in-context faster than Qwen3 does. Evo2 40B outperforms Qwen3 14B significantly, and it takes until Qwen3 8B to exceed the ICL ability of Evo2 1B.

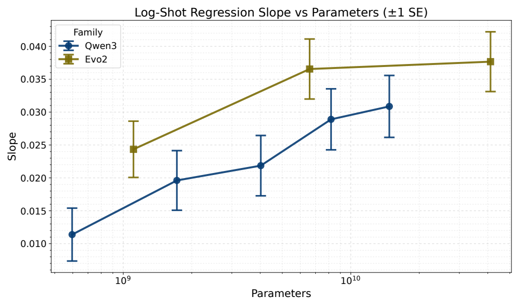
Taken together, this data suggests that Qwen3 and Evo2 have similar amounts of pretraining exposure to be able to solve these tasks, and that Evo2 simply has better overall ICL capability (in this regime) – even though Qwen's ICL ability increases more rapidly with respect to parameters.



(a) Few-shot performance of Qwen3 and Evo2 models. All models show consistent linear improvement with respect to log(shots). In contrast, no such improvement occurs for the naive baseline.



(b) Baseline accuracy decreases slightly with scale as the model gains more parameters for both Evo2 and Qwen3.



(c) ICL rate vs. model size: sharp gains up to 4B for Qwen3; mild boost from Evo2 1B to 7B; both plateau after 4–7B.

Figure 5: Few-shot behavior and scaling trends across Qwen3 and Evo2.

E Demonstration of Evo2's ICL Ability on a Genomic Task

We demonstrate that Evo2 7B can perform few-shot in-context learning (ICL) on human_nontata_promoters, a binary genomic classification task in which the model must predict whether a 251 nucleotide-long human DNA sequence is a promoter. To make the task non-trivial, sequences in the positive class are restricted to non-TATA promoters, removing the "TATA box", a pattern that serves as an obvious, exploitable marker (Umarov & Solovyev, 2017; Grešová et al., 2023).

For each test query, we construct a few-shot prompt by concatenating balanced examples from the training set (equal numbers of promoters and non-promoters), followed by a held-out test sequence. Each training example is immediately followed by an encoded label. The label consists of a 24 nucleotide buffer (a single nucleotide repeated 24 times) and then a 24 nucleotide label run (another nucleotide repeated 24 times). The buffer nucleotide and the two label nucleotides (for class 0 vs. class 1) are re-sampled uniformly at random without replacement from $\{A, C, G, T\}$ for every trial, so the mapping changes across evaluations. This prevents any nucleotide-specific biasing effects.

To predict the test label, we score two candidate prompts - one where the test sequence is followed by the class-0 label and one followed by the class-1 label - and choose the label with lower perplexity on the label run (i.e., computed only over the final 24 label nucleotides, excluding the buffer). This lets us measure whether Evo2 can infer the label mapping from the few-shot context and apply it to the query sequence.

We observe monotonic improvement with respect to shot count, with accuracy rising from below random baseline at 1-shot (48.2%) to 63.6% at 16-shot. These gains further rise to 68.6% after 256 shots. A saturated power law achieves an R^2 of 0.998 in the 2-256 shot regime, which is characteristic of few-shot ICL. Furthermore, we show via ablation that the improvement is due to correctly-labeled examples presented in context—randomly permuting the labels in the 16-shot regime drops performance from 63.6% back to random chance (50.2%).

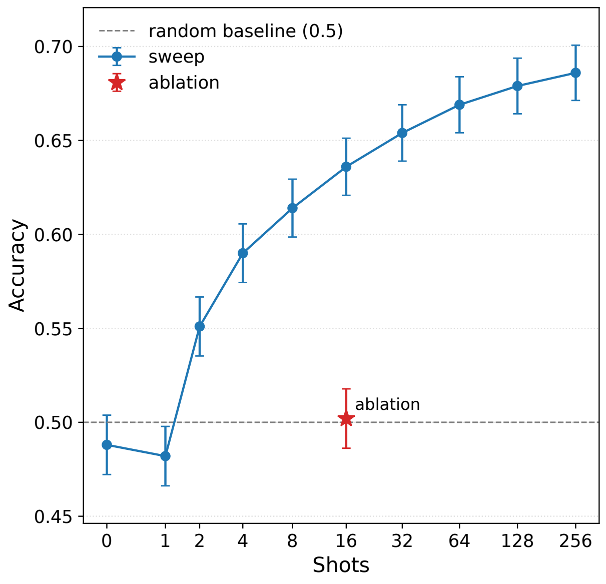


Figure 6: Performance of Evo2 7B on human_nontata_promoters at varying shot counts. Error bars are ± one standard error. Evo2 7B's performance shows clear monotonic improvements with respect to shot count.

F Analysis of ICL with Program Synthesis Tasks: Bit-Diversity

BitLoad captures a measure of theoretical information bottleneck required to make predictions in program synthesis tasks. However, this does not always correlate with what models find easy to do. In Fig. 8 and Fig. 9, we plot the accuracy of Evo and Qwen models with respect to BitLoad. We notice that few high BitLoad tasks have high accuracy compared to all medium BitLoad tasks, illustrating that some high BitLoad tasks can be easy for these models to solve, probably due to the nature of patterns found during pre-training.

To further analyze the nature of ICL exhibited through these tasks, we define BitDiversity as *the number of minority bits in the output string.* In Fig. 10 and Fig. 11, we plot the accuracy with respect to BitDiversity. These plots try to estimate the effect of entropy in the output on model performance, and we see a more expected trend: models tend to perform better on low-entropy outputs, regardless of BitLoad. However, bin-wise performance trends are always increasing with shots, supporting the central hypothesis that ICL increases with increasing number of demos in these models.

Prevalence of 0-BitDiversity outputs. Looking at the expected and predicted outputs of trials from our tasks (Fig. 12), we made a few interesting observations about 0-BitDiversity (BD) outputs.

- Around 25% of true targets are 0-BD. This is a significantly high number as we only have 2 0-BD bit strings in all possible bit strings of any length K. It implies that many of our tasks create low BitDiversity outputs on random inputs, i.e., 0-BD outputs.
- Models tend to produce a lot of 0-BD outputs, much higher than the number of 0-BD true targets in the low-shot regime. But this number quickly drops to the expected number with higher shots.
- A large portion of the baseline (1-shot) performance can be explained by this prevalence of 0-BD outputs, but with more shots we get stronger evidence of ICL with increasing correctly predicted non 0-BD cases.

Understandable Mistakes. A potential confound is what we will call ”understandable mistakes”. These mistakes occur when the model outputs an incorrect answer that would be correct for some other programs given the few-shot context. Formally, a model's output y (see 3.4 for notation): $y = M\Big(e_1 \to f(e_1), e_2 \to f(e_2), \ldots, e_N \to f(e_N), x\Big)$ is an **understandable mistake** if $y \neq f(x)$ but there exists $f_2 \in F$ such that $\forall e_i \quad f(e_i) = f_2(e_i)$ and $y = f_2(x)$.

These understandable mistakes are an alarming confound at low shot counts, but their effect vanishes by $N = 16$. They occur most in tasks with low BitDiversity, which can often be confused with each other. Figure 7 below shows how understandable mistakes in Qwen3-4B's inferences decay exponentially as the number of shots increases.

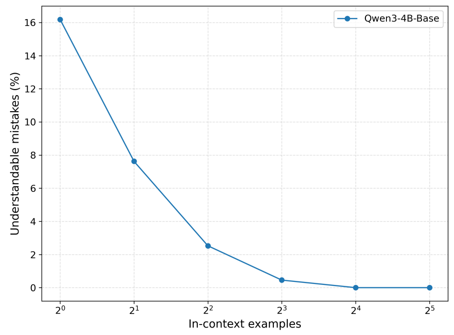


Figure 7: Qwen3-4B's rate of understandable mistakes with respect to the number of shots. Despite starting at 16% in the one-shot regime, they fall to less than 1% by 8 shots and vanish entirely at 32 shots. This underscores how understandable mistakes are only a confound at very low shot counts and can essentially be ignored past 4 shots.

Noise due to Monte Carlo Trials. We use $m = 8$ for our per-function Monte Carlo trials when we compute accuracy. This has little aggregate impact when comparing model performance across the entire suite, but drastically reduces the significance of results when comparing models and trends at the task-level. We only have eight discrete bands of accuracy at which we can estimate a model's per-task performance – which reduces the expressiveness of regression. Worse, this can lead to models getting a lucky prompt or two with a 0-BD output, which raises performance to 12.5% or

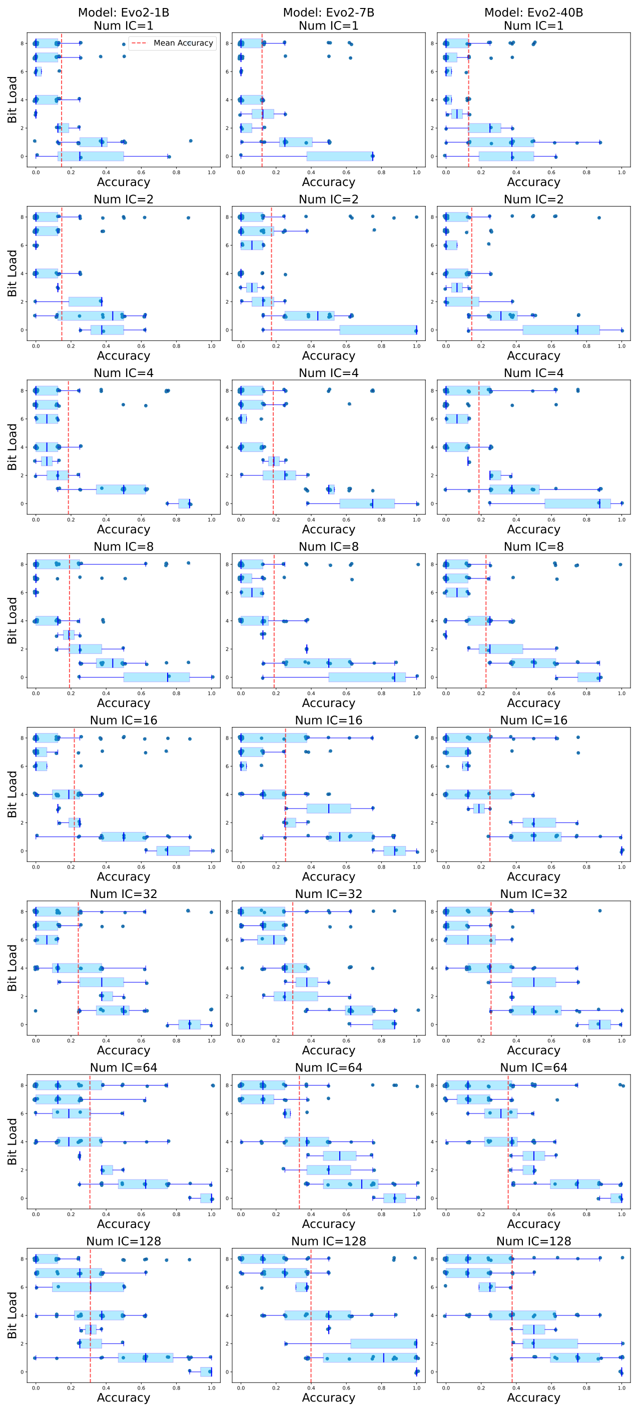


Figure 8: BitLoad vs Accuracy for all Evo models at all ICL shot-numbers. The per-BitLoad distribution of model performance at each shot level shows an uneven affinity of models for solving tasks at different BitLoads. However, all trends support our overall hypotheses.

25% without the model truly understanding the task. Addressing these confounds would simply require increasing m by an order of magnitude or so. Alternatively, tasks of interest could be identified and m selectively increased for those tasks to enable more nuanced analysis.

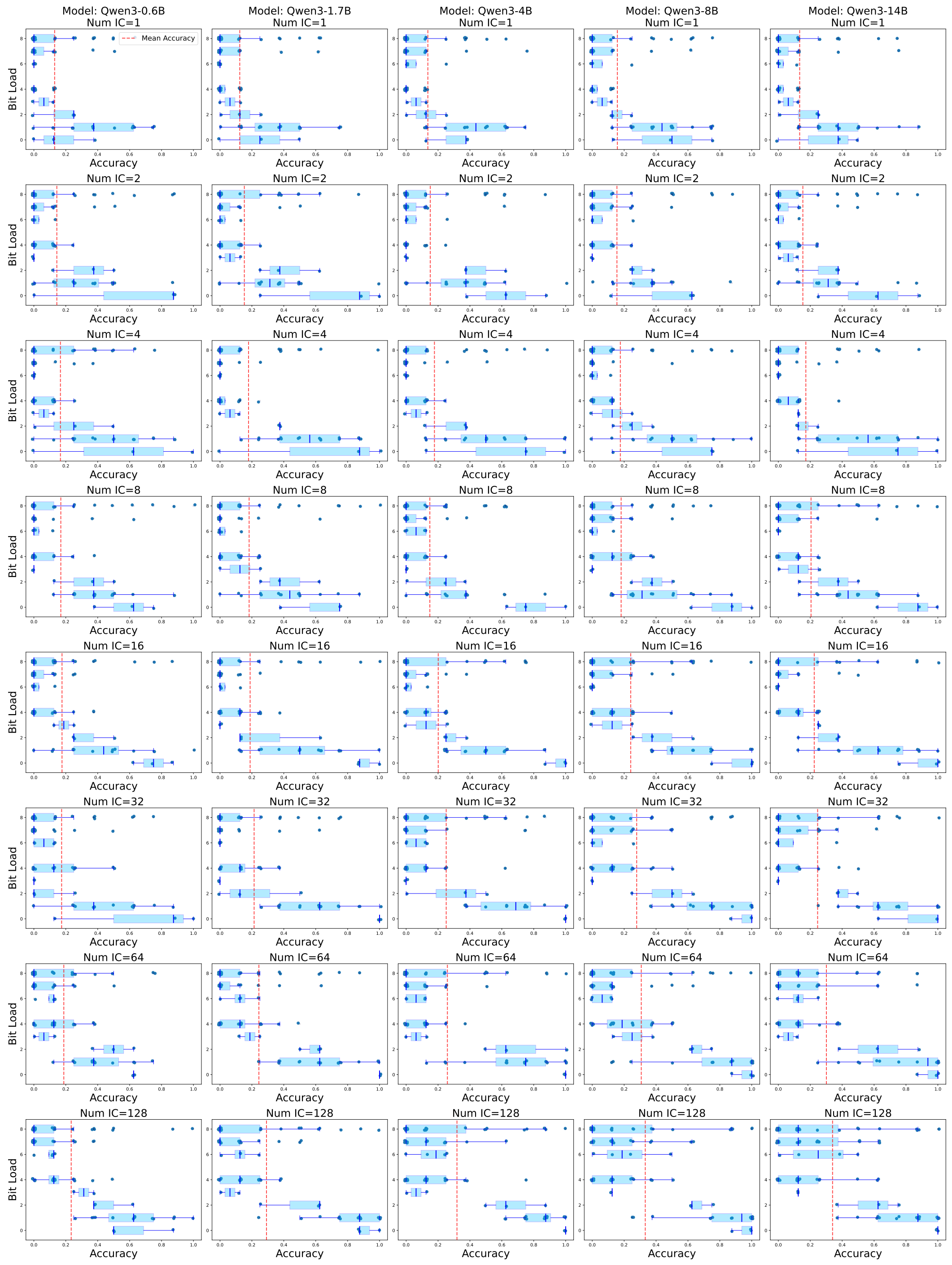


Figure 9: BitLoad vs Accuracy for all Qwen models at all ICL shot-numbers.

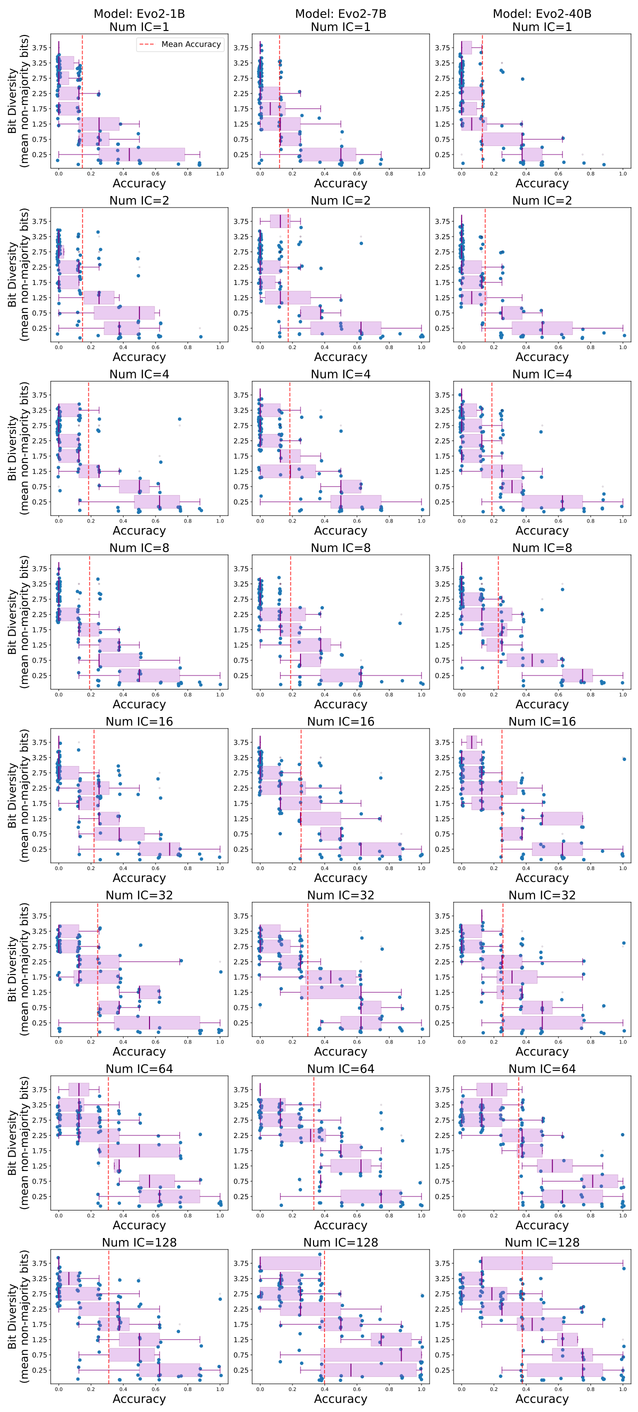


Figure 10: BitDiversity vs Accuracy for all Evo models at all ICL shot-numbers. Model performance follows a more natural pattern of increasing performance with decreasing output entropy. This highlights that it is difficult for models to decipher ICL patterns for high entropy outputs.

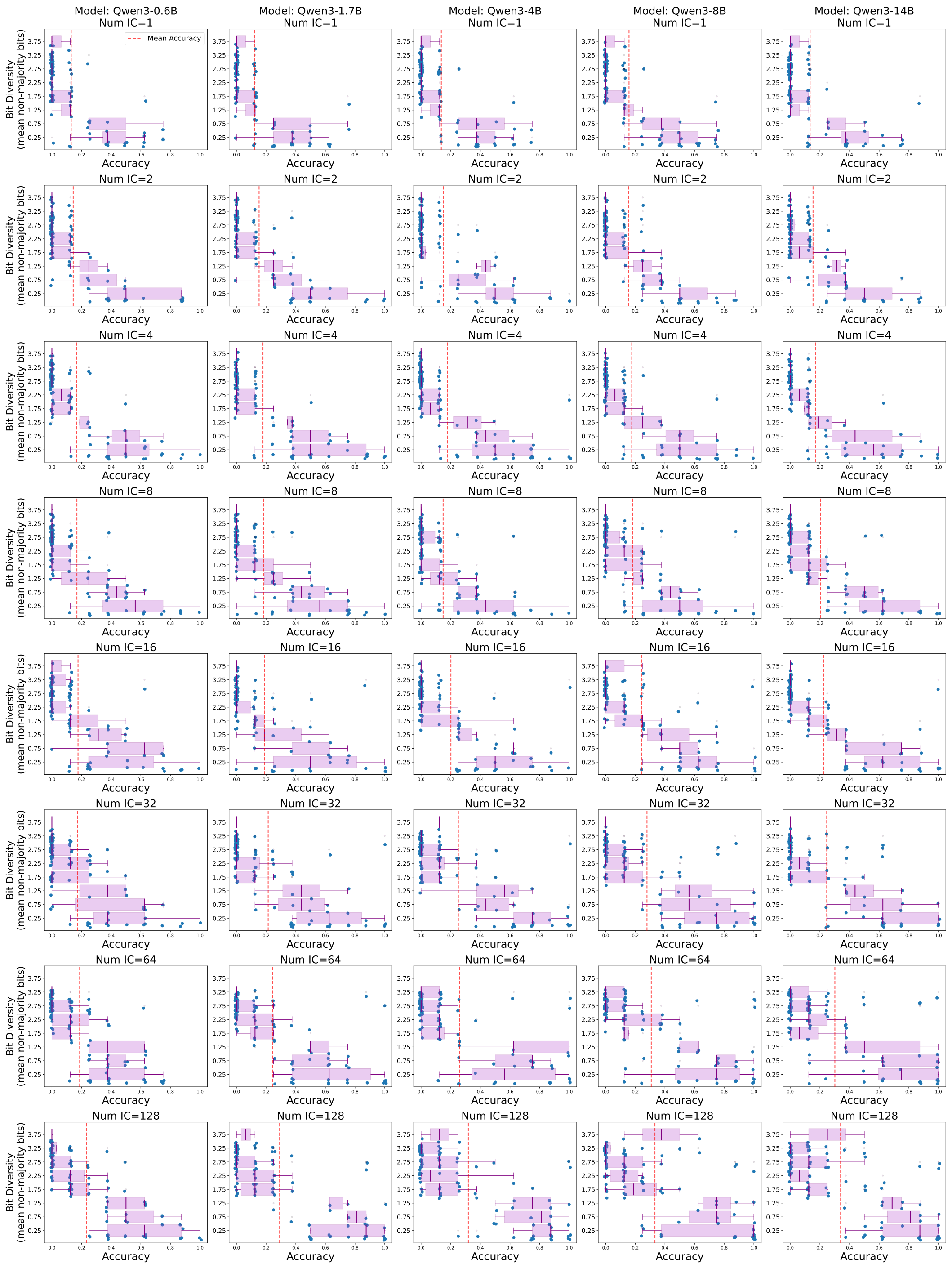


Figure 11: BitDiversity vs Accuracy for all Qwen models at all ICL shot-numbers.

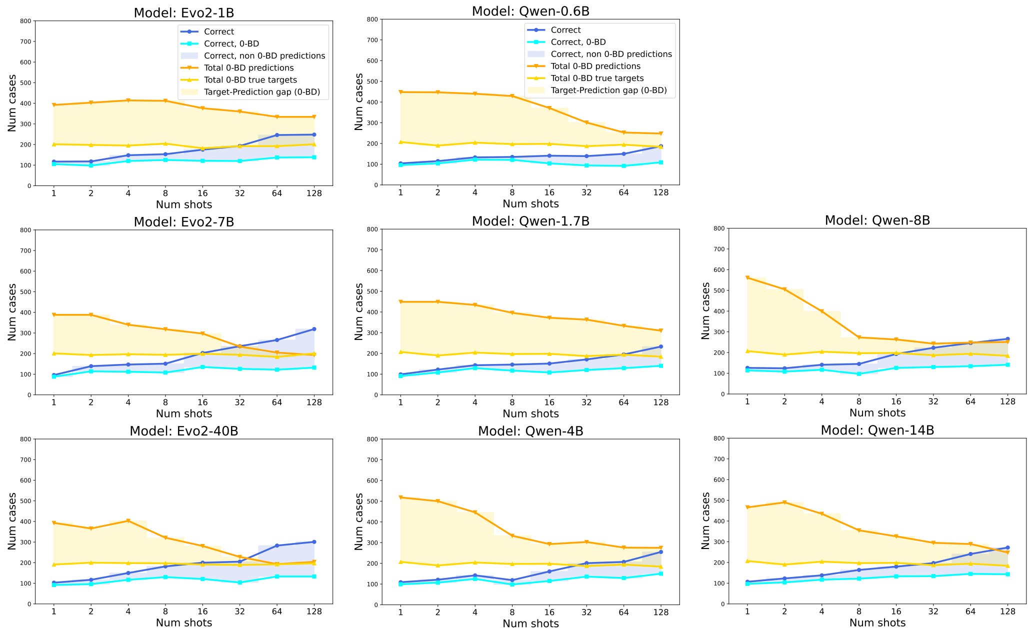


Figure 12: Enumerating cases of 0 BitDiversity. For our defined tasks, around 25% of the true targets have 0 BitDiversity (BD). With low-shots, models tend to produce a large number of 0-BD predictions. But this number decreases significantly with increasing shots and tends to match the actual prior value. Similarly, a majority of the baseline (1-shot) performance of models can be explained through 0-BD outputs. With a higher number of shots, the model starts learning higher entropy patterns and presents stronger evidence of ICL.