OUT OF SUPPORT, NOT OUT OF REACH: ICL WITH UNSUPPORTED INSTRUCTIONS AND COMPOSED TASKS

Anonymous authors
Paper under double-blind review

ABSTRACT

In-Context Learning (ICL) enables pretrained LLMs to adapt to downstream tasks from a small set of input-output demonstrations, without parameter updates. Yet existing theory primarily explains ICL as identifying a latent task already represented in pretraining, leaving unclear whether ICL can succeed when either the instruction or the target task falls outside the pretraining support. We develop a theoretical framework under mild assumptions that separates these two forms of distribution shift. For target tasks represented in pretraining, we show that demonstrations can identify the task and reduce sensitivity to pretrain-supported prompt templates as their number increases. For an unsupported instruction, we show that under a fixed model-induced reference extension, demonstration evidence can dominate an N-independent instruction shift while preserving the exponential identification rate. We then use the same ICL framework as an interpretive lens for long-horizon target tasks that are not represented in pretraining but admit a plan that decomposes them into pretraining-supported subtasks. Applying the supported-task analysis step by step yields a standard accumulated-error bound and clarifies how invoking familiar subtasks along an intermediate-state trajectory can avoid directly selecting the unsupported complete task. Together, our results characterize when demonstrations, instructions, and task decomposition can support ICL beyond the corresponding pretraining supports.

1 INTRODUCTION

In-Context Learning (ICL) (Brown et al., 2020) is widely used to adapt Large Language Models (LLMs) through an instruction and a small set of input–output demonstrations, without updating model parameters. A prominent line of theory explains ICL as implicit Bayesian inference over a latent task (Xie et al., 2022; Wang et al., 2023; Panwar et al., 2024): the demonstrations provide evidence about which task generated the prompt, and prediction improves as the posterior concentrates on that task. The standard account relies on two support assumptions. First, the posterior ranges over latent tasks or concepts already represented by the pretraining mixture, so the target has positive prior probability under that mixture (Xie et al., 2022; Wies et al., 2023; Wang et al., 2023). Second, the instruction is typically omitted or absorbed into a fixed canonical prompt format, rather than modeled as a task-conditional variable that may differ across demonstrations (Akyürek et al., 2023; von Oswald et al., 2023; Ahn et al., 2023). In deployment, however, both assumptions can fail.

Consider first the instruction. Real prompts can depart from the pretraining distribution in systematic ways. A user of Indian English may write *please revert with the numbers*, using *revert* to mean "reply," while the same word is associated predominantly with "undo" in the model's coding data. The term *bias* may denote a regression intercept to one user and a fairness property to another. A company's internal service may retain the name of an open-source project even after its behavior has diverged. Other deviations are inconsistent rather than repeated — typos that differ from slot to slot, or demonstrations assembled by retrieval or by multiple annotators, each carrying its own rendering. What matters is the interaction of two axes: whether the instruction was seen paired with the intended task during pretraining, and whether one instruction governs the prompt or each demonstration carries its own. Table 1 formalizes these seven structures. They lead to our first research question: *When can additional demonstrations overcome an instruction outside the task-conditional pretraining support, and when do additional instructions introduce enough new shift to offset the evidence supplied by those demonstrations?*

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

The other support assumption fails at the level of the target task itself. Language-model agents are increasingly asked to execute long-horizon objectives that may never have appeared as complete tasks during pretraining. At the same time, many such objectives admit plans composed of pretraining-supported skills: extracting a field, transforming a record, invoking a known operation, checking an intermediate state, and aggregating the results. A latent-task posterior cannot directly select a target task with zero pretraining probability, but an agent need not execute that task as one indivisible mapping. As an application of the ICL perspective, we ask: *When can an out-of-support target task be realized by a plan that decomposes it into pretraining-supported subtasks, and how do step-wise errors accumulate over a long execution horizon?* We study execution under a given task decomposition, rather than the separate problem of how an agent discovers that decomposition.

We address both questions in a common latent-task framework. For unsupported instructions, task evidence from demonstrations competes with the shift induced by the instruction structure. For long-horizon tasks, the same perspective is applied along an intermediate-state trajectory of supported subtasks. A pathwise Lipschitz analysis quantifies sensitivity along both types of paths.

Our Contributions. We distinguish instruction-level support from task-level support and characterize how generalization at these two levels is shaped by the number and provenance of demonstrations, instruction placement, and task decomposition. Our main contributions are summarized as follows.

- **Characterizing ICL with Supported and Unsupported Instructions (Section 3).** Holding the target task within the pretraining task support, we distinguish instructions that lie inside or outside the corresponding task-conditional instruction support and analyze the seven prompt structures in Table 1. We show that informative demonstrations identify the target task and, under a fixed model-induced reference extension, preserve their exponential rate despite a fixed unsupported instruction whose continuation cost does not grow with the demonstration count. This is a conditional statement about the specified extension, rather than an identification of its off-support values from the pretraining distribution. For mixed or varying instructions, the instruction-shift cost can instead accumulate; convergence depends on whether task evidence grows faster than that cost.
- **Applying the ICL View to Long-Horizon Tasks through Supported Subtasks (Section 4).** We use the preceding ICL framework to interpret a target task with zero pretraining probability. Although such a task cannot be selected directly by the pretrained latent-task posterior, it may be realized by a plan that composes supported subtasks along an intermediate-state trajectory. We bound the final error by the accumulated errors of the execution steps, revealing why a single poorly supported or unstable subtask can become the bottleneck.
- **Validating the Theoretical Statements through Experiments (Section 5).** We test the predicted interaction between instruction structure and demonstration count, the pathwise characterization of prompt sensitivity, and the effect of decomposition quality on multi-step task execution.

Taken together, our instruction results offer a unified view of ICL beyond pretraining support. For a specified off-support reference extension, demonstrations can supply evidence for the intended latent task that dominates a fixed instruction-shift cost; realizing this behavior in the actual model further requires controlled model–reference error. Task decomposition, meanwhile, brings a novel long-horizon objective within the model's repertoire of pretrained skills. In both cases, success depends on whether the useful evidence dominates the sensitivity accumulated along the path from the test prompt to what the model has already learned.

2 Problem Setup

This section formalizes the latent-task view of ICL and defines the seven prompt formats used throughout our analysis by distinguishing task-conditional instruction support and instruction placement.

Latent Task Model. Let $\mathcal{V}$ be a finite vocabulary and let $\mathcal{S}$ be a finite space of individual inputs, outputs, and intermediate states. Following Xie et al. (2022), we assume that each sample $(x, y) \in \mathcal{S} \times \mathcal{S}$ is generated from a latent task t in a finite task set $\mathcal{T}$. Conditioned on t, $X \sim P_{X|t}$, $Y = f_t(X)$, where $f_t : \mathcal{S} \to \mathcal{S}$ is the task-specific mapping. We write $P(\cdot)$ for distributions and $p(\cdot)$ for their mass or density functions, and assume $\pi(t) = \mathbb{P}(T = t) > 0$ for every $t \in \mathcal{T}$. Section 4 will additionally consider downstream tasks $t^\dagger \notin \mathcal{T}$, for which we extend the pretraining prior by $\pi(t^\dagger) = 0$. An ICL prompt with N demonstrations is written as $\boldsymbol{z}_N = \big(\boldsymbol{i}, (x_1, y_1), \ldots, (x_N, y_N), x_{\text{query}}\big)$; we omit

the subscript and write $\boldsymbol{z}$ when N is clear from context. Here $\boldsymbol{i}$ collects the instruction variables under the chosen prompt format and x_{query} is the query input. Let D_0 denote the support of the pretraining prompt distribution across all values of N. All almost-sure statements as $N \to \infty$ refer to the length-N prefixes of one infinite sequence of demonstrations generated by the stated mechanism.

Instructions. A single task may admit multiple instructions and prompt formats. For a pretraining-supported task $t \in \mathcal{T}$, we model its instructions as draws from the task-conditional distribution $P_{i|t}$. Support is therefore task-conditional: for a target task $t^* \in \mathcal{T}$, an instruction i is *supported* if $p(i \mid t^*) > 0$ and *unsupported* if $p(i \mid t^*) = 0$. This definition asks whether pretraining ever paired the instruction with the intended task, not whether the words appeared in pretraining or whether a user regards them as semantically correct. For example, *revert* may be common in coding data but unsupported for the task of drafting a reply. The distinction matters because $p(i \mid t^*)$ enters the latent-task posterior directly: ordinary Bayesian conditioning applies to supported instructions, whereas an unsupported instruction requires the off-support analysis developed below.

Fix a maximum instruction length L_i, let $e : \mathcal{V} \cup \{\langle \text{pad} \rangle\} \to \mathbb{R}^d$ be the token embedding map with $e(\langle \text{pad} \rangle) = 0$, and right-pad every instruction to length L_i. Write $\mathcal{V}^{\leq L_i} := \bigcup_{m=0}^{L_i} \mathcal{V}^m$. For $i = (v_1, \ldots, v_m) \in \mathcal{V}^{\leq L_i}$, define its padded embedding matrix as $E(i) := [e(v_1), \ldots, e(v_m), 0, \ldots, 0]^\top \in \mathbb{R}^{L_i \times d}$. Thus instructions with different token lengths lie in one common ambient space; Appendix F provides the corresponding padding-stability justification. For each competing task $t \neq t^*$ and supported instruction i, define the task-relative instruction score $r_t(i) := \log(p(i \mid t)/p(i \mid t^*))$. The score compares how likely the instruction is under the competing task t and the target task t^*. For the continuous-path analysis, we use a fixed model-induced extension of this score along the instruction-embedding path defined below. We do not require the off-support extension to arise from separate normalized densities on the continuous instruction-embedding space. Its values away from the symbolic embeddings are additional structure made explicit in the continuation condition below, rather than something identified by the finite symbolic support. Appendix C shows that an analytic Transformer scoring map supplies one concrete such extension and implies the required finite continuation chain.

Prompt Formats. The formats below model common ways in which prompts are assembled. A user or system message supplies one global instruction; a structured dataset may copy one field or operator into every record; and retrieval or multiple annotators may attach a different rendering to each demonstration. These placement patterns matter because a shared instruction contributes one piece of task evidence, whereas independently varying instructions can contribute a new likelihood factor with every demonstration. Crossing the three placement patterns—single, repeated, and varying—with supported versus unsupported instructions gives formats 1–6. Format 7 captures partial corruption, where a supported template remains in most slots but typos or schema errors affect a subset (Table 1). This distinction follows from the assumed generative factorization rather than from the Transformer architecture alone. Appendix G.4 gives a controlled experiment showing that a Transformer can learn the corresponding effective instruction sample sizes from cross-position correlation structure.

The support labels in the table follow directly from the sampling mechanism. In formats 1, 3, and 5, every instruction is drawn from $P_{i|t^*}$ and is therefore supported. Formats 2, 4, and 6 instead select instructions outside its support, so $p(\tilde{i} \mid t^*) = 0$. Format 7 contains both the shared supported template and distinct unsupported corruptions. For notational convenience, we use $\boldsymbol{i}$ to denote the collection of all instructions appearing in a prompt, regardless of the specific prompt format. Unless otherwise specified, input-output pairs (X, Y) are drawn i.i.d. conditional on the latent task.

3 ICL WITH SUPPORTED AND UNSUPPORTED INSTRUCTIONS

Throughout this section, the target task is pretraining-supported: $t^* \in \mathcal{T}$ and $\pi(t^*) > 0$. The distinction studied below concerns only whether the instruction is supported conditional on that task.

Let $M(y \mid \boldsymbol{z})$ denote the output probability of the actual pretrained model. The Bayesian predictor $p(y \mid \boldsymbol{z})$ on supported prompts, and its fixed continuation $\bar{p}(y \mid \boldsymbol{z})$ on unsupported prompts, serve as theoretical reference predictors, which need not coincide with M exactly (see Appendix D.1).

Table 1: Seven prompt formats classified by support and instruction placement. Supported means that the instruction was paired with the intended task with positive probability during pretraining; unsupported means that this probability is zero. Appendix D.2 gives complete examples.

#	Support	Appearance	Prompt format	Example
1	Supported	Single	$\boldsymbol{z} = (i, (x_n, y_n)_{n=1}^N, x_{\text{query}})$	A standard instruction stated once: "Classify ticket priority."
2	Unsupported	Single	$\boldsymbol{z} = (\tilde{i}, (x_n, y_n)_{n=1}^N, x_{\text{query}})$	Indian-English *revert* used once to mean "reply."
3	Supported	Repeated	$\boldsymbol{z} = \big((i, x_n, y_n)_{n=1}^N, (i, x_{\text{query}})\big)$	A standard schema key, e.g. `task: sentiment`, repeated in every record.
4	Unsupported	Repeated	$\boldsymbol{z} = \big((\tilde{i}, x_n, y_n)_{n=1}^N, (\tilde{i}, x_{\text{query}})\big)$	One private API label, e.g. `operation: Atlas`, reused in every record.
5	Supported	Varying	$\boldsymbol{z} = \big((i_n, x_n, y_n)_{n=1}^N, (i_{\text{query}}, x_{\text{query}})\big)$	Retrieved examples using familiar paraphrases such as "sentiment" and "tone."
6	Unsupported	Varying	$\boldsymbol{z} = \big((\tilde{i}_n, x_n, y_n)_{n=1}^N, (\tilde{i_{\text{query}}}, x_{\text{query}})\big)$	Multi-tenant examples carrying distinct private labels such as `triage_acme` and `route_7G`.
7	Mixed	Corrupted	$\boldsymbol{z} = \big((i_n, x_n, y_n)_{n=1}^N, (i_{\text{query}}, x_{\text{query}})\big), i_n \in \{i, \tilde{i}_n\}$	A standard template with sparse typos such as `sentimnet` and `sentimemt`.

3.1 ICL WITH PRETRAINING-SUPPORTED INSTRUCTIONS

We first consider prompt formats whose instructions lie in the support of the pretraining distribution.

Assumption 3.1 (Identifiability of task evidence)**.** For every pair of distinct tasks $t_1, t_2 \in \mathcal{T}$, the demonstration distributions satisfy $\mathrm{KL}\big(P_{X,Y|t_1} \,\|\, P_{X,Y|t_2}\big) > 0$, and the corresponding demonstration log-likelihood ratio is integrable under $P_{X,Y|t_1}$. For format 5, the log-likelihood ratio of the joint instruction–demonstration distribution is also integrable. Under the format-specific generation mechanism described in Section 2,

$$\mathrm{KL}\big(P_{\boldsymbol{i}|t_1} \otimes P_{X,Y|t_1} \,\|\, P_{\boldsymbol{i}|t_2} \otimes P_{X,Y|t_2}\big) = \mathrm{KL}\big(P_{\boldsymbol{i}|t_1} \,\|\, P_{\boldsymbol{i}|t_2}\big) + \mathrm{KL}\big(P_{X,Y|t_1} \,\|\, P_{X,Y|t_2}\big) > 0. \quad (1)$$

Thus the task evidence is identifiable in formats 1, 3, and 5, while the demonstration KL condition above also supplies the gap used in the unsupported-instruction results below.

Under this assumption, the demonstrations provide exponentially increasing evidence for the target task. In format 5, the independently sampled paraphrases form part of this task evidence rather than a single shared instruction factor.

Theorem 3.2 (Informal: model bound with supported instructions)**.** *Consider formats 1, 3, and 5 under Assumption 3.1. Let t^* be the target task. Then there exist constants $\alpha, \beta > 0$ and a finite model-approximation error $\varepsilon_M \geq 0$ such that, almost surely, for all sufficiently large N,*

$$|M(y_{query} \mid \boldsymbol{z}) - p(y_{query} \mid x_{query}, t^*)| \leq \beta e^{-\alpha N} + \varepsilon_M. \quad (2)$$

The bound shows several key insights. The exponent α is controlled by the least distinguishable competing task: when every demonstration provides positive evidence for t^*, posterior mass on the competitors decays exponentially with N. The prefactor β collects effects that do not grow with the demonstration count, including the task prior, the query distribution, and a shared instruction factor. Thus, in formats 1 and 3, copying the same supported instruction into more positions does not create additional independent evidence; the improvement with N comes from the new input–output pairs. In format 5, by contrast, each supported paraphrase is part of an instruction–demonstration triple, so it can contribute additional evidence for identifying the task. See Appendix D.4 for complete proof.

3.2 ICL WITH PRETRAINING-UNSUPPORTED INSTRUCTIONS

3.2.1 SINGLE OR REPEATED UNSUPPORTED INSTRUCTIONS.

We first consider formats 2 and 4 when the same unsupported instruction is used throughout the prompt. Format 2 contains one displayed occurrence; format 4 contains $N + 1$ copies of one shared realization. The repeated copies in format 4 are not treated as $N + 1$ independent task observations: the instruction template is sampled once and copied deterministically into all displayed slots. Thus,

for every task t,

$$p(\boldsymbol{z} \mid t) = p(\boldsymbol{i} \mid t)p(x_{\text{query}} \mid t) \prod_{n=1}^{N} p(x_n, y_n \mid t). \tag{3}$$

Format 2 is the corresponding single-occurrence special case.

Let $\boldsymbol{i}_0$ be a supported anchor and $\widetilde{\boldsymbol{i}}$ the unsupported endpoint. In the padded instruction-embedding space of Section 2, define

$$E_s := (1-s)E(\boldsymbol{i}_0) + sE(\widetilde{\boldsymbol{i}}), \qquad s \in [0,1]. \tag{4}$$

For each competing task, we analytically continue its supported instruction log-likelihood ratio along this path. We denote the corresponding continued predictive probability by $\bar{p}(y_{\text{query}} \mid \boldsymbol{z})$. This continuation is a model-induced reference extension; because the endpoint has zero probability, its value is not identified by the pretraining distribution alone.

Under this condition, we can give an exponential bound on the continued reference predictor and a corresponding bound for the actual model.

Theorem 3.3 (Informal: unsupported shared instruction)**.** *Suppose Assumption 3.1 holds and, for every $t \neq t^*$, the supported log-likelihood ratio admits a fixed finite analytic continuation from the supported anchor to the unsupported endpoint. Then there exist constants $\alpha, \beta > 0$, independent of N, such that, almost surely for all sufficiently large N, the continued reference predictor satisfies*

$$|\bar{p}(y_{query} \mid \boldsymbol{z}) - p(y_{query} \mid x_{query}, t^*)| \leq \beta e^{-\alpha N}. \tag{5}$$

For the actual pretrained model,

$$\left| M(y_{query} \mid \boldsymbol{z}_N(\widetilde{\boldsymbol{i}})) - p(y_{query} \mid x_{query}, t^*) \right| \leq \beta e^{-\alpha N} + \varepsilon_{M,N}(\widetilde{\boldsymbol{i}}), \tag{6}$$

where $\varepsilon_{M,N}(\widetilde{\boldsymbol{i}})$ is the model–reference transport error along the tied shared-template path, defined formally in Appendix D.1. Assumption D.21 shows when the transport error is uniformly bounded, and when the actual model converges to the target-task predictor.

Theorem 3.3 handles a prompt outside the pretraining support in the repeated-instruction setting. In particular, the same unsupported instruction $\widetilde{\boldsymbol{i}}$ is repeated across all demonstrations and satisfies $p(\widetilde{\boldsymbol{i}} \mid t^*) = 0$, so the ordinary Bayes conditional at $\boldsymbol{z}_N(\widetilde{\boldsymbol{i}})$ is not determined directly by the pretraining distribution. Because all displayed occurrences are copies of one shared instruction realization, its fixed finite continuation is performed only once and introduces only an N-independent factor. At the same time, evidence from the demonstrations that identifies the target task accumulates with N. Therefore, even though $p(\widetilde{\boldsymbol{i}} \mid t^*) = 0$, the theoretical continuation error still decreases exponentially for this specified reference extension. Appendix D.1.1 gives sufficient quantitative conditions under which the transport error is small, controlled study in Appendix C.6 complements this result.

Remark 3.4. This setting reflects a common mismatch between the notation represented during LLM pretraining and the notation chosen by a user. A user may consistently express a supported task using a symbol or style of language that never appeared in pretraining. The theorem suggests that, provided the underlying latent task was learned during pretraining and the new notation is used consistently across the demonstrations, sufficiently many examples can still identify the target task within the reference construction. When the model-transport error is also small, the actual model can exhibit the corresponding behavior from the input pattern rather than requiring that notation itself have been observed during pretraining. The continuation perspective is also relevant when $p(\boldsymbol{i} \mid t^*) > 0$ but extremely small, since it may occur too rarely in a finite pretraining corpus for the model to learn its association with t^* reliably, making it effectively unsupported for the learned model.

3.2.2 Mixed and varying supported and unsupported instructions (format 6-7).

We next consider format 7, in which most demonstrations share a supported instruction while a subset contain distinct unsupported corruptions. The supported occurrences act as a common anchor, and each corrupted occurrence is connected to that anchor by the same construction used in Section 3.2.1, which quantifies how strongly the corruption can favor a competing task (see Appendix D.6).

Intuitively, every informative demonstration supplies evidence that separates the target task from its competitors, while each distinct unsupported instruction can shift some of that evidence back toward

a competing task. Generalization therefore depends on whether the evidence accumulated from the demonstrations grows faster than the total adverse effect of the corruptions.

Format 6 is the limiting case in which every demonstration has a distinct unsupported instruction. The demonstrations provide accumulating evidence for the target task, but the continuation costs of the unsupported instructions also accumulate with N. Convergence remains possible when the demonstration evidence grows faster than the total continuation cost. Thus the analysis gives no unconditional convergence guarantee for format 6. This explains why increasing the number of examples need not remove instruction sensitivity in this setting, consistent with Figure 1b. Complete details are given in Appendix D.7.

Remark 3.5. These setting models typos in user-provided demonstrations. Some demonstrations may use a supported instruction from the pretraining support, whereas a typo may produce an instruction that the model never observed during pretraining. Moreover, different typos need not produce the same unsupported instruction, which distinguishes it from the repeated setting of Section 3.2.1. Their demonstration-dependent cost can still vanish when it grows more slowly than the evidence supplied by the demonstrations, while the actual model retains the displayed transport error.

4 ICL FOR TASK BEYOND THE PRETRAINING TASK SUPPORT

Language-model agents are often asked to carry out long-horizon tasks that may not be represented as single latent tasks during pretraining. Such a target has $\pi(t^\dagger) = 0$ and therefore receives zero posterior mass for every prompt and every N. Consequently, the Bayesian latent-task reference for direct ICL can only form a mixture of supported-task predictions, and increasing N cannot make this reference predictor select the unsupported target task. In this section, we apply the preceding ICL analysis to an agent that follows a plan decomposing such a task into supported subtasks, and our aim here is to use the ICL framework to explain how support and model–reference errors behave along agent execution, rather than to introduce a separate theory of agent learnability.

4.1 EXECUTING LONG-HORIZON TASKS THROUGH SUPPORTED SUBTASKS

Agent execution circumvents the reference-level obstruction by never asking the posterior to represent $t^\dagger$ as a single latent task. Consider a K-step plan with supported subtasks $t_0, \dots, t_{K-1} \in \mathcal{T}$. Let $x_n^{(k)} \in \mathcal{S}$ be the state of demonstration n after step k, where the state contains any execution history needed by the next step. We assume the recursion $x_n^{(k+1)} = f_{t_k}(x_n^{(k)})$, and define $f_{t^\dagger}(x)$ as the terminal state $x^{(K)}$ obtained by this recursion from $x^{(0)} = x$. Thus $f_{t^\dagger} = f_{t_{K-1}} \circ \cdots \circ f_{t_0}$. We reuse $\boldsymbol{z}_N$ for the full agent-execution prompt, whose N demonstrations are the trajectories $(x_n^{(0)}, \dots, x_n^{(K)})$. For step k and a current state x, let $\boldsymbol{z}_N^{(k)}(x)$ denote the prompt formed by the supported instruction for t_k, the N transition pairs $(x_n^{(k)}, x_n^{(k+1)})$, and x. Starting from $x^{(0)} = x_{\text{query}}$, the agent produces $x^{(1)}, \dots, x^{(K)}$ sequentially, with $y_{\text{query}} = x^{(K)}$. The latent task of the step-k prompt is therefore the supported subtask $t_k \in \mathcal{T}$. This is the decisive structural point: although $\pi(t^\dagger) = 0$, every step prompt represents a supported task with $\pi(t_k) > 0$.

Assumption 4.1 (Step-wise supported identifiability)**.** There are constants $\alpha_k, \beta_k > 0$ independent of N such that, almost surely for all sufficiently large N and every state x reachable at step k,

$$1 - p\big(f_{t_k}(x) \mid \boldsymbol{z}_N^{(k)}(x)\big) \le \beta_k e^{-\alpha_k N}. \tag{7}$$

Note that Assumption 4.1 is not an extra hypothesis about $t^\dagger$; it is the component of Theorem 3.2 applied K times, together with the requirement that the bound be uniform over the correct intermediate states rather than at a single query.

Theorem 4.2 (Informal: step-wise ICL bound for a decomposed long-horizon task)**.** *Let $f_{t^\dagger} = f_{t_{K-1}} \circ \cdots \circ f_{t_0}$ with $t_0, \dots, t_{K-1} \in \mathcal{T}$, and let Assumption 4.1 hold. For each step k, let $\varepsilon_M^{(k)}(N) \ge 0$ be its finite model-transport error, defined formally in Appendix E. Then, almost surely for all sufficiently large N,*

$$\Big| M\big(y_{\textit{query}} \mid \boldsymbol{z}_N\big) - p(y_{\textit{query}} \mid x_{\textit{query}}, t^\dagger) \Big| \le \sum_{k=0}^{K-1} \Big[\beta_k e^{-\alpha_k N} + \varepsilon_M^{(k)}(N) \Big]. \tag{8}$$

This bound follows by applying the supported-task ICL guarantee at each step and accumulating the resulting errors across the fixed execution horizon under the explicit step-call protocol above. It connects a standard composition argument to the support distinctions developed above. For each step, α_k measures how quickly the demonstrations distinguish the intended subtask from competing supported tasks, while β_k collects the N-independent difficulty of that step, including its prior, instruction and query factors and the uniformity over reachable intermediate states. The bound is strongest when K is moderate, every α_k is bounded away from zero, and the β_k remain controlled. Its decreasing component is determined by the hardest step through $\min_k \alpha_k$, whereas every step also contributes its model-transport error. Thus one poorly identified or poorly approximated subtask can dominate the entire chain.

The main-text bound uses an explicit step-call protocol: at step k, the executor calls the model on exactly the extracted prompt $\boldsymbol{z}_N^{(k)}(x^{(k)})$ and retains no additional model-visible history or cache. Full-context autoregressive rollout requires additional rollout and, when applicable, padding corrections, treated in Appendix E and F.

Remark 4.3. In practice, this interpretation favors agent plans built from supported, well-specified operations with clear intermediate states. Decomposing a difficult long-horizon task into simpler supported steps can make each step easier to identify and therefore improve its exponential rate. However, an unnecessarily long plan creates more opportunities for error, and an ambiguous instruction, an unsupported intermediate representation, or a weakly supported subtask can make one α_k small or one β_k large. A concise, well-chosen plan can therefore outperform direct prediction, whereas a long or poorly chosen execution plan can be less reliable.

4.2 Factors Governing Generalization Beyond Pretraining

The pathwise Lipschitz sensitivity. Recall the instruction path E_s from (4), which connects the supported anchor $\boldsymbol{i}_0$ to the unsupported instruction $\widetilde{\boldsymbol{i}}$. Define $\mathcal{L}_{\mathrm{path}}$ as the smallest constant such that, for every competing task t and every $s, s' \in [0,1]$,

$$|r_t(E_s) - r_t(E_{s'})| \le \mathcal{L}_{\mathrm{path}} \|E_s - E_{s'}\|_F, \tag{9}$$

where the reference log-odds along unsupported parts of the path are understood through the continuation in Section 3.2.1. The resulting bound can be written schematically as

$$\text{error} \;\lesssim\; \beta \exp\Big(-\alpha N + \mathcal{L}_{\mathrm{path}} \|E(\widetilde{\boldsymbol{i}}) - E(\boldsymbol{i}_0)\|_F\Big). \tag{10}$$

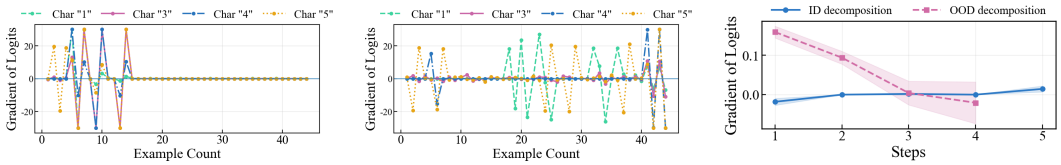
Thus a good anchor is not merely a point with a small local derivative: it should make the combined anchor and whole-path cost small. The value of $\mathcal{L}_{\mathrm{path}}$ is determined by the pretrained model, the chosen anchor and path, and the competing tasks. It tends to be smaller when pretraining has learned nearby instructions, representations, or computations that change smoothly toward the test instruction, and can be large when the path crosses a poorly learned or task-ambiguous region. In experiments (Section 5), we use the discrete gradient of the correct-output log probability as the prompt changes with the demonstration count as an observable proxy for effective prompt sensitivity. In the agent experiment, changes in output log probability across execution steps provide a qualitative diagnostic of step-wise stability.

The demonstrations sensitivity. The demonstrations etermine the evidence rate α in (10): α is large when each additional demonstration supplies information that distinguishes the target task from its plausible competitors. Such demonstrations are mutually consistent with one latent rule and include inputs on which competing rules predict different outputs. By contrast, repetitive examples on which several tasks agree leave the task identity ambiguous and produce a small α. Good demonstrations therefore make the favorable term αN grow rapidly enough to dominate the fixed anchor and path penalties, and they reduce the sensitivity of the final prediction.

For example, consider the prompt

$$\texttt{Japan} \to \texttt{Tokyo}, \quad \texttt{France} \to \texttt{Paris}, \quad \texttt{USA} \to ?$$

The first two pairs do not distinguish "country to capital" from "country to largest city," because the two rules agree on both demonstrations. For the query, the two interpretations instead predict `Washington DC` and `New York`. The demonstrations thus provide little evidence against either



(a) **Consistent mismatched instructions.** The discrete gradients of the correct-digit log probabilities diminish as the repeated demonstrations identify the intended task.

(b) **Varying mismatched instructions.** The discrete gradients of the correct-digit log probabilities remain irregular when different instruction mismatches accumulate across demonstrations.

(c) **Step-wise sensitivity under agent task decomposition.** The in-distribution execution plan has a smoother gradient profile, whereas the out-of-distribution plan exhibits larger fluctuations.

Figure 1: Empirical analogues of (a) consistent and (b) varying instruction mismatch, together with (c) the role of supported, stable subtasks in Theorem 4.2 and Section 4.2. In (a)–(b), the fixed correct answer is `1345`; *Char "1"*, *Char "3"*, *Char "4"*, and *Char "5"* denote the score-gradient traces for these correct digits at the first through fourth output positions, respectively.

Table 2: Test accuracy (↑) of ICL under ambiguous and identifying demonstrations ($k = 8$).

Demonstration	Qwen3-30B-A3B	Qwen3-235B-A22B
Ambiguous / **Identifying**	0.164 ± 0.039/**0.559 ± 0.074**	0.202 ± 0.037/**0.602 ± 0.043**

competing task, corresponding to a small α and a prediction that can change sharply under a small modification of the prompt. In contrast, consider

$$\texttt{Australia} \rightarrow \texttt{Sydney}, \qquad \texttt{Turkey} \rightarrow \texttt{Istanbul}, \qquad \texttt{USA} \rightarrow ?$$

The first two pairs rule out the capital interpretation and strongly favor the largest-city interpretation. The resulting larger α makes the prediction for $\texttt{USA} \rightarrow ?$ more stable and improves the bound.

Applying the sensitivity view to agent execution. For an unsupported target task, the Bayesian reference for direct ICL cannot select $t^\dagger$ from its pretraining task posterior. A decomposed agent plan avoids this reference-level obstruction by using supported subtasks $t_0, \ldots, t_{K-1}$. Applying Theorem 3.2 yields two contributions: $\beta_k e^{-\alpha_k N}$ measures subtask identification error, and $\varepsilon_M^{(k)}(N)$ measures model–reference error. Additional demonstrations reduce the first term, while the second must be controlled separately. The zero prior of the complete task is therefore no longer decisive: the agent only needs to identify and execute the supported steps. However, the errors add across the K steps, so one weakly identified or poorly approximated subtask can dominate the final result.

5 Experiments

5.1 Demonstration Evidence for Task Identification

Section 4.2 predicts that ICL performance under prompt shift is governed by the demonstration evidence rate α. Here we give a motivational example of the corresponding intuition: demonstrations that better distinguish the target task from plausible competitors should produce more reliable task identification and higher ICL accuracy.

We use Sport Identification as the representative example here. **(Setup)** Given a country, the task is to output its associated sport label. We construct two types of demonstrations: *identifying* demonstrations, where the sport is a distinctive national or traditional sport that strongly pins down a unique latent rule, and *ambiguous* demonstrations, where the sport is related to the country but can be explained by multiple plausible latent rules, thereby providing weaker evidence about which rule the prompt instantiates. We sample $k \in \{2, 4, 8\}$ demonstrations from the pool and evaluate on a fixed test set. **(Results)** As shown in Table 2, the identifying pool consistently yields higher accuracy than the ambiguous pool, which is consistent with the prediction that informative demonstrations provide a larger evidence rate, leading to stronger task identification and better generalization. See Appendix G.2 for details, and see Appendix G.3 for a controlled validation.

5.2 Prompt Sensitivity vs. Demonstration Count

In Section 3, Theorems 3.2 and 3.3, together with Section 3.2.2, predict different scaling for fixed and varying instruction shifts. We design the following experiment to test the corresponding distinction.

(Setup) We use a pretrained `Qwen3-235B-A22B` model on a three-digit addition task with a fixed query and target answer. We vary only the operator pattern in the demonstrations: the *addition-only* regime uses $+$ throughout; the *consistent-mismatch* regime uses the same $\times$ symbol throughout while retaining sum outputs; and the *varying-mismatch* regime independently selects $\times$, $/$, or $-$ for each demonstration. These regimes provide an aligned baseline and empirical analogues of consistent and varying instruction mismatch. Because the exact conditional support of a pretrained LLM is not observable, the experiment tests the effect of consistent versus varying instruction mismatch rather than claiming that these symbols have literal zero pretraining probability. **(Results)** We evaluate the model under the three prompt formats with 1 to 50 demonstrations. We visualize the discrete gradient, as the demonstration count changes, of the model-assigned log probability for each correct output digit in Figures 1a and 1b. In the addition-only regime, the gradient remains near zero throughout. Under a consistent mismatch, sensitivity is initially high but gradually decreases as the demonstrations identify addition. Under varying mismatches, the gradients remain irregular and exhibit large spikes even at high demonstration counts. This persistent sensitivity is qualitatively consistent with the theoretical distinction between fixed and varying instruction shifts. See Appendix G.5 for details.

5.3 Task Decomposition Quality in Controlled Agent Execution

We instantiate the multi-step execution setting of Section 4.1 with an arithmetic agent that follows a fixed plan. The agent maintains an explicit state $x^{(k)}$ and updates it at step k by applying the prescribed operation f_{t_k}. We intervene on whether the operations in the plan are supported, while holding the target task and execution horizon fixed. The experiment therefore illustrates the subtask-identification mechanisms summarized by Theorem 4.2: the ICL view favors execution when the decomposition consists of supported subtasks with controlled step-wise sensitivity. A decomposition containing unsupported subtasks does not receive the same guarantee.

(Setup) We perform two-stage LoRA training for `Llama3.2-3B`. The base model already performs one- to five-digit addition and simple multiplication reliably, and the first stage adds 20K six-digit addition samples. We treat seven-digit addition as the composed target task. For the *In-Distribution Agent Plan*, the agent executes six-digit addition, one-digit addition, multiplication by 10, and aggregation as its component operations, and we add 10K step-level training examples for each required operation. The *Out-of-Distribution Agent Plan* instead requires the agent to perform intermediate seven-digit additions. These operations receive no additional training and cannot be executed reliably in isolation. We evaluate the same model under both plans. **(Evaluation)** At every execution step, we record whether the state transition is correct and the log probability assigned to the correct output. Variation in the latter across steps is a qualitative diagnostic of step-wise stability. Since adjacent steps generally implement different operations, this quantity is not a direct estimate of the within-step path constant in Section 4.2. **(Results)** Figure 1c shows that the in-distribution agent plan has a smoother step-wise log-probability profile. The out-of-distribution plan exhibits larger fluctuations and performs worse than direct prediction without agent execution, illustrating the cost of including a poorly supported operation in the plan. Additional results are provided in Appendix G.6.

6 Conclusion

In this work, we separate two forms of generalization in in-context learning (ICL) beyond pretraining support. At the instruction level, supported instructions yield exponential convergence, and the continued reference predictor preserves this rate for a single or repeated unsupported instruction. Mixed prompts also converge when their total continuation burden grows more slowly than the demonstration evidence. At the task level, the Bayesian latent-task reference for direct ICL assigns zero posterior mass to a target outside its supported task set. Within the same reference model, applying the ICL perspective step by step explains how an agent may execute such a long-horizon target without selecting it as a single latent task, provided that a plan decomposes it into supported, identifiable subtasks. The resulting, standard composition bound accumulates the errors of the execution steps. Together, the instruction results and this agent interpretation clarify when more demonstrations, a different prompt format, or a task decomposition may extend model behavior beyond the tasks directly represented in pretraining.

AI USE STATEMENT

Generative AI tools were used to assist with developing and editing experimental code, constructing synthetic experimental pipelines, analyzing and interpreting results, searching and summarizing relevant literature, and drafting, editing, and formatting the manuscript. The authors reviewed all AI-assisted material, and validated reported results. The authors made all final research decisions and take responsibility for the complete contents of the paper, including all text, code, experiments, and artifacts produced with AI assistance.

REPRODUCIBILITY STATEMENT

We state the probabilistic setting, assumptions, and model-level distinctions used by the theory in Sections 2–4 and Appendix C. Complete proofs of the instruction- and CoT-level results, including the model–reference transport terms, are provided in Appendices D and E. The main experimental protocols are summarized in Section 5; Appendix G reports the task constructions, model variants, training and evaluation procedures, prompt formats, demonstration counts, random-seed aggregation, metrics, and additional results. It also documents the controlled model–reference, CoT error-propagation, continuation-reach, and Qwen3 soft-prompt interpolation experiments, together with their numerical sanity checks.

REFERENCES

Rishabh Agarwal, Avi Singh, Lei Zhang, Bernd Bohnet, Luis Rosias, Stephanie Chan, Biao Zhang, Ankesh Anand, Zaheer Abbas, Azade Nova, John D. Co-Reyes, Eric Chu, Feryal Behbahani, Aleksandra Faust, and Hugo Larochelle. Many-shot in-context learning. *Advances in Neural Information Processing Systems*, 37:76930–76966, 2024.

Kwangjun Ahn, Xiang Cheng, Hadi Daneshmand, and Suvrit Sra. Transformers learn to implement preconditioned gradient descent for in-context learning. *Advances in Neural Information Processing Systems*, 36, 2023.

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. *International Conference on Learning Representations*, 2023.

Yu Bai, Fan Chen, Huan Wang, Caiming Xiong, and Song Mei. Transformers as statisticians: Provable in-context learning with in-context algorithm selection. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

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

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.

Guhao Feng, Bohang Zhang, Yuntian Gu, Haotian Ye, Di He, and Liwei Wang. Towards revealing the mystery behind chain of thought: A theoretical perspective. *Advances in Neural Information Processing Systems*, 2023.

Zixuan Gong, Xiaolin Hu, Huayi Tang, and Yong Liu. Towards auto-regressive next-token prediction: In-context learning emerges from generalization. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=gK1rl98VRp.

Jianliang He, Xintian Pan, Siyu Chen, and Zhuoran Yang. In-context linear regression demystified: Training dynamics and mechanistic interpretability of multi-head softmax attention. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=3TM3fxwTps.

Juno Kim and Taiji Suzuki. Transformers learn nonlinear features in context. In *ICLR 2024 Workshop on Mathematical and Empirical Understanding of Foundation Models*, 2024. URL https://openreview.net/forum?id=WvvAu6XI1M.

Juno Kim, Tai Nakamaki, and Taiji Suzuki. Transformers are minimax optimal nonparametric in-context learners, 2024. URL https://arxiv.org/abs/2408.12186.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners, 2023. URL https://arxiv.org/abs/2205.11916.

Hongbo Li, Lingjie Duan, and Yingbin Liang. Provable in-context learning of nonlinear regression with transformers, 2025. URL https://arxiv.org/abs/2507.20443.

Zhiyuan Li, Hong Liu, Denny Zhou, and Tengyu Ma. Chain of thought empowers transformers to solve inherently serial problems. *International Conference on Learning Representations*, 2024.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, William B. Dolan, Lawrence Carin, and Weizhu Chen. What makes good in-context examples for gpt-3? *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pp. 100–114, 2022.

Fei Lu and Yue Yu. Transformer learns the cross-task prior and regularization for in-context learning, 2025. URL https://arxiv.org/abs/2505.12138.

Arvind V. Mahankali, Tatsunori Hashimoto, and Tengyu Ma. One step of gradient descent is provably the optimal in-context learner with one layer of linear self-attention. In *The Twelfth International Conference on Learning Representations*, 2024.

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

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? *Empirical Methods in Natural Language Processing*, 2022.

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, 2022. URL https://arxiv.org/abs/2209.11895.

Madhur Panwar, Kabir Ahuja, and Navin Goyal. In-context learning through the bayesian prism. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=HX5ujdsSon.

Hongjin Su, Jungo Kasai, Chen Henry Wu, Weijia Shi, Tianlu Wang, Jiayi Xin, Rui Zhang, Mari Ostendorf, Luke Zettlemoyer, Noah A. Smith, and Tao Yu. Selective annotation makes language models better few-shot learners. In *The Eleventh International Conference on Learning Representations*, 2023.

Eric Todd, Millicent L. Li, Arnab Sen Sharma, Aaron Mueller, Byron C. Wallace, and David Bau. Function vectors in large language models, 2024. URL https://arxiv.org/abs/2310.15213.

Johannes von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. *International Conference on Machine Learning*, 2023.

Xinyi Wang, Wanrong Zhu, and William Yang Wang. Large language models are implicitly topic models: Explaining and finding good demonstrations for in-context learning. *arXiv preprint arXiv:2301.11916*, 2023.

Albert Webson and Ellie Pavlick. Do prompt-based models really understand the meaning of their prompts? In Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz (eds.), *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2300–2344, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.167. URL https://aclanthology.org/2022.naacl-main.167/.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Noam Wies, Yoav Levine, and Amnon Shashua. The learnability of in-context learning. *Advances in Neural Information Processing Systems*, 2023.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit bayesian inference. *International Conference on Learning Representations*, 2022.

Seonghyeon Ye, Hyeonbin Hwang, Sohee Yang, Hyeongu Yun, Yireun Kim, and Minjoon Seo. Investigating the effectiveness of task-agnostic prefix prompt for instruction following. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19386–19394, 2024.

Ruiqi Zhang, Spencer Frei, and Peter L. Bartlett. Trained transformers learn linear models in-context. *Journal of Machine Learning Research*, 25(49):1–55, 2024.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate before use: Improving few-shot performance of language models. *International Conference on Machine Learning*, 139: 12697–12706, 2021.

A List of Common Notations

This appendix summarizes the notation used in the main text. Symbols that are local to a single proof are introduced where they first appear.

Sequences, tasks, and data.

- $\mathcal{V}$: the finite token vocabulary; $\mathcal{S}$ is the finite space of individual inputs, outputs, and intermediate states.
- $\mathcal{T}$: the finite set of latent tasks supported by pretraining; $\pi(t)$ is the pretraining prior on $t \in \mathcal{T}$.
- t^*: the target task in Section 3, where $t^* \in \mathcal{T}$; $t^\dagger$ is the target task in Section 4, where $t^\dagger \notin \mathcal{T}$ and $\pi(t^\dagger) = 0$.
- (x, y): an input–output pair; f_t is the task-specific mapping, so that $y = f_t(x)$ under a deterministic task t.
- x_{query} and y_{query}: the query input and its output, respectively.
- $P(\cdot)$ and $p(\cdot)$: a distribution and its probability mass or density function. In particular, $p(t \mid \boldsymbol{z})$ and $p(y_{\text{query}} \mid \boldsymbol{z})$ denote the latent-task posterior and posterior predictive distribution.
- $M(y \mid \boldsymbol{z})$: the output probability assigned by the actual pretrained model. The Bayesian quantities p and $\bar{p}$ below are its theoretical reference predictors.
- D_0: the support of the pretraining prompt distribution across all demonstration counts.

Prompts and instructions.

- $\boldsymbol{z}$ or $\boldsymbol{z}_N$: an ICL prompt containing N demonstrations and one query. The seven prompt formats are defined in Section 2.
- $\boldsymbol{i}$: an instruction sequence; $\widetilde{\boldsymbol{i}}$ denotes an instruction unsupported for the target task, and $\boldsymbol{i}_0$ denotes a supported anchor instruction. The notation $\boldsymbol{i}$ collects all instruction variables in a prompt when a format contains more than one.
- D: the supported instruction domain used in the supported instruction results of Section 3.
- L_i: the fixed maximum instruction length, and $\mathcal{V}^{\le L_i} := \bigcup_{m=0}^{L_i} \mathcal{V}^m$.
- $e : \mathcal{V} \cup \{\langle \text{pad} \rangle\} \to \mathbb{R}^d$: the token embedding map, with $e(\langle \text{pad} \rangle) = 0$.
- $E(\boldsymbol{i}) \in \mathbb{R}^{L_i \times d}$: the right-padded embedding matrix of instruction $\boldsymbol{i}$ defined in Section 2.
- $E_s = (1-s)E(\boldsymbol{i}_0) + sE(\widetilde{\boldsymbol{i}})$, $s \in [0, 1]$: the straight path in the padded instruction-embedding space from a supported anchor to an unsupported instruction.
- $r_t(\boldsymbol{i}) := \log\big(p(\boldsymbol{i} \mid t)/p(\boldsymbol{i} \mid t^*)\big)$: the task-relative instruction score on supported instructions. The notation $r_t(U)$, $U \in \mathbb{R}^{L_i \times d}$, denotes its fixed model-induced extension to instruction-embedding matrices for the continuous-path analysis.
- $\bar{p}$: the posterior or posterior predictive induced by a fixed analytic continuation at a zero-probability prompt. Ordinary p is used when the relevant conditional is determined directly on the support.

Instruction-level convergence.

- $\alpha > 0$: a demonstration-evidence rate separating the target task from its competitors; $\beta > 0$ denotes a generic N-independent prefactor in an exponential error bound.
- ε_M: a finite model-approximation or transport error between the actual model M and the theoretical reference predictor. For a shared unsupported instruction, $\varepsilon_{M,N}(\widetilde{\boldsymbol{i}})$ is the N-dependent error along the tied shared-template path, while $\overline{\varepsilon}_M(\widetilde{\boldsymbol{i}})$ denotes its optional uniform upper bound. The pathwise definitions are given in Appendix D.1.
- $\mathcal{L}_{\text{path}}$: the pathwise Lipschitz sensitivity of the continued instruction log-likelihood ratios, defined in (9).

- $\|\cdot\|_F$ denotes the Frobenius norm. In particular, $\|E(\widetilde{\boldsymbol{i}}) - E(\boldsymbol{i}_0)\|_F$ is the distance from the supported anchor to the unsupported instruction. Appendix F also denotes this distance by Δ_{inst}.
- $U_N \subseteq [N]$: the indices of demonstrations containing unsupported instructions in the mixed setting; Ω_n is the finite continuation cost associated with the n-th such instruction.

Agent execution.

- K: the fixed number of execution steps; $t_0, \ldots, t_{K-1} \in \mathcal{T}$ are the supported subtasks whose composition realizes $t^\dagger$.
- $x_n^{(k)}$ and $x^{(k)}$: the states after step k in demonstration n and in the query execution, respectively. In particular, $x^{(0)} = x_{\text{query}}$ and $x^{(k+1)} = f_{t_k}(x^{(k)})$.
- $\boldsymbol{z}_N^{(k)}(x)$: the step-$k$ ICL prompt evaluated at state x; $\boldsymbol{z}_N$ denotes the full agent-execution prompt.
- $\alpha_k, \beta_k > 0$: the exponential convergence constants for supported subtask t_k.

Padding.

- $\mathrm{Pad}_M(\boldsymbol{z})$: the position-preserving extension of prompt $\boldsymbol{z}$ to a fixed length M using $\langle \mathrm{pad} \rangle$ tokens.
- m: the number of added padding positions; Δ_{att} is the attention-logit gap between a selected informative position and the most competitive visible padding position.
- $\varepsilon_{\mathrm{pad}}$: the model-output log-probability perturbation caused by padding under the conditions of Proposition F.4.
- $\delta_{k,\mathrm{pad}}(N)$: the conditional-probability discrepancy between the original and padded step-k execution prompts. It is zero when the added positions are strictly future positions removed by the causal mask.

B Related Work

Existing theories rely on strong assumptions. A substantial body of theoretical work seeks to understand the mechanisms underlying ICL by analyzing how transformers implement learning procedures over context. At the mechanistic level, studies identify specific architectural patterns such as induction heads and function vectors—that enable example reuse and pattern matching across the context (Olsson et al., 2022; Elhage et al., 2021; Todd et al., 2024). Complementary algorithmic analyses show that, under linear or kernelized assumptions, ICL can be interpreted as implicitly gradient descent or kernel regression in activation space (Akyürek et al., 2023; von Oswald et al., 2023; Ahn et al., 2023; Mahankali et al., 2024), or as a form of task selection induced by the prior over pretraining tasks (Bai et al., 2023). While these mechanistic and algorithmic perspectives have significantly advanced our understanding of how ICL is computed, they typically rely on restrictive architectural assumptions, such as linear attention, single-layer or attention-only transformers, or single-head attention—and on idealized data distributions such as random generalized linear regression, which abstract away properties that are critical in practice (von Oswald et al., 2023; Mahankali et al., 2024; Zhang et al., 2024). A complementary line of work adopts a Bayesian or statistical inference perspective, interpreting ICL as approximate posterior inference over latent tasks conditioned on in-context demonstrations (Xie et al., 2022; Wang et al., 2023; Panwar et al., 2024). This framing provides a principled account of how generalization improves with more demonstrations, and has been extended to study the effects of noise, misspecification, and pretraining bias on ICL behavior (Gong et al., 2025). However, existing analyses within this framework uniformly assume a fixed and canonical context format, treating demonstrations as drawn from a coherent, well-specified data-generating process. As a result, they do not characterize how ICL behaves when the format of the context itself varies—a limitation that our work directly addresses.

Practical factors affect ICL generalization. Beyond mechanistic and statistical accounts, a large body of empirical work investigates how practical design choices in prompt construction affect ICL performance. A first line of work focuses on the role of demonstrations themselves: studies show that the number of demonstrations, their selection strategy, and the correctness of their labels each substantially influence downstream performance (Agarwal et al., 2024; Liu et al., 2022; Su et al.,

2023). Notably, Min et al. (2022) find that label correctness in demonstrations matters less than expected, suggesting that ICL generalizes by retrieving a latent task structure rather than learning directly from individual examples. A second line of work examines chain-of-thought prompting, showing that including intermediate reasoning steps in demonstrations can dramatically improve performance on complex tasks (Wei et al., 2022; Kojima et al., 2023; Nye et al., 2021). Subsequent theoretical work explains CoT's benefit in terms of expressivity: CoT enables deeper computation without additional layers (Feng et al., 2023), expands the model's recognition capacity for complex problem classes (Merrill & Sabharwal, 2024), or supports more serial computation than standard forward passes allow (Li et al., 2024). However, these expressivity-based accounts do not characterize how CoT-structured demonstrations affect ICL generalization as the number of reasoning steps grows, which is a gap our work addresses through a multi-step error propagation analysis. A third line of work studies sensitivity to prompt format and template design, finding that surface-level variations in instruction wording, demonstration ordering, and label presentation can lead to large performance swings (Ye et al., 2024; Zhao et al., 2021; Webson & Pavlick, 2022). Despite the breadth of these empirical findings, they remain largely descriptive: existing work does not provide a theoretical criterion for when format variation preserves successful generalization and when it fundamentally disrupts consistent task inference. Our work provides such a criterion by formally distinguishing prompt formats under which the posterior predictive converges exponentially with more demonstrations from those under which convergence necessarily fails.

Mechanistic explanations of ICL. A complementary line of work tries to explain ICL at the level of internal representations and computational mechanisms, examining how specific architectural components give rise to in-context behavior. At the representational level, studies identify induction heads and function vectors as key circuits that enable pattern matching and example reuse across the context window (Olsson et al., 2022; Elhage et al., 2021; Todd et al., 2024), and related work shows that multi-head softmax attention can implement regression behaviors in simplified settings (He et al., 2025; Li et al., 2025). At the algorithmic level, several studies argue that transformers implicitly perform gradient descent or kernel regression over in-context examples (Akyürek et al., 2023; von Oswald et al., 2023; Ahn et al., 2023), and related perspectives characterize ICL as algorithm selection or prior-induced regularization across tasks (Bai et al., 2023; Lu & Yu, 2025). A separate line of work examines whether standard pretraining objectives can produce the representational capacity assumed by these accounts, establishing conditions under which pretrained transformers exhibit ICL-like behavior (Kim & Suzuki, 2024; Kim et al., 2024). While these mechanistic and algorithmic explanations have significantly advanced our understanding of the computational substrate of ICL, they characterize how ICL is implemented within a fixed architecture and format but do not address when and why ICL generalizes across varying prompt formats, nor how practical design choices such as demonstration count, CoT structure, or instruction variation affect the quality of task inference. Our work is therefore complementary to this line, operating at the level of generalization theory rather than mechanistic realization.

C Probability Models

This appendix explains how the latent-task probabilities used in the analysis can be related to a Transformer evaluated on continuous instruction embeddings. The argument has three logically distinct parts. Population cross-entropy identifies the Bayes predictor on the support of the pretraining distribution; an analytic Transformer scoring map selects one particular extension away from that support; and a separate model-transport term measures the discrepancy between this reference and the actual pretrained model. In particular, the off-support extension is supplied by the model architecture and is not claimed to be statistically identified by pretraining data of zero probability.

C.1 What population cross-entropy identifies

Under the latent-task framework of Section 2, pretraining learns a conditional predictor $q(\cdot \mid \boldsymbol{z})$ by minimizing population log-loss

$$\mathcal{R}(q) := \mathbb{E}_{(\boldsymbol{z},y)\sim P}\big[-\log q(y \mid \boldsymbol{z})\big]. \tag{11}$$

For every predictor whose probabilities are positive on the support of P,

$$\mathcal{R}(q) = \mathcal{R}(p) + \mathbb{E}_{\boldsymbol{z}}\,\mathrm{KL}(p(\cdot \mid \boldsymbol{z}) \,\|\, q(\cdot \mid \boldsymbol{z}))\,. \tag{12}$$

Consequently, the population minimizer equals the Bayes posterior predictive almost surely on the pretraining-prompt support:

$$q^*(y \mid \boldsymbol{z}) = p(y \mid \boldsymbol{z}) = \sum_{t \in \mathcal{T}} p(y \mid x_{\text{query}}, t) p(t \mid \boldsymbol{z}), \qquad P_{\boldsymbol{z}}\text{-a.s.} \tag{13}$$

Equation (12) also makes the limitation precise: population risk places no restriction on $q(\cdot \mid \boldsymbol{z})$ when $P_{\boldsymbol{z}}(\boldsymbol{z}) = 0$. Thus cross-entropy optimality by itself neither defines nor identifies a predictor at an unsupported prompt.

For finite data, finite model capacity, or imperfect optimization, exact equality in (13) need not hold. This discrepancy is kept explicit through the supported-model gap r_0 in (31); it is not absorbed into the analytic continuation assumption.

C.2 A Transformer-induced reference extension

We distinguish the following objects throughout the paper:

(i) p is the latent-task data-generating distribution. Its symbolic instruction probabilities $p(\boldsymbol{i} \mid t)$ have their ordinary statistical meaning on the pretraining support.

(ii) p_{ref} is a hybrid Bayesian reference predictor. It agrees with the Bayes predictor on supported prompts and, on unsupported instruction embeddings, inserts fixed Transformer-induced instruction log-odds into the latent-task posterior while retaining the theoretical demonstration, prior, query, and task-conditional output factors.

(iii) M is the actual pretrained model. It need not equal p_{ref}; their discrepancy is controlled by ε_M in Appendix D.1.

Only instruction log-likelihood ratios, rather than a Lebesgue density over the entire instruction-embedding space, are required by the proofs. It is therefore both sufficient and conceptually cleaner to extend these ratios directly. Let a fixed Transformer-induced scoring map produce task scores

$$a_t(U), \qquad t \in \mathcal{T}, \qquad U \in \mathbb{R}^{L_i \times d}, \tag{14}$$

and define the reference instruction log-odds relative to t^* by

$$r_t(U) := a_t(U) - a_{t^*}(U) + c_t, \qquad t \neq t^*, \tag{15}$$

where c_t is independent of U. Equivalently, if the scores define $q_{\text{ref}}(t \mid U) = \mathrm{softmax}(a(U))_t$, then

$$\log \frac{q_{\text{ref}}(t \mid U)}{q_{\text{ref}}(t^* \mid U)} = a_t(U) - a_{t^*}(U). \tag{16}$$

The softmax normalizer cancels, so the extended log-odds inherit the regularity of the task scores directly.

The required connection to the latent-task probabilities is a supported calibration condition: on every supported symbolic instruction used by the analysis,

$$r_t(E(\boldsymbol{i})) = \log \frac{p(\boldsymbol{i} \mid t)}{p(\boldsymbol{i} \mid t^*)}. \tag{17}$$

For example, if $q_{\text{ref}}(t \mid E(\boldsymbol{i})) = p(t \mid \boldsymbol{i})$ on the supported domain, Bayes' rule gives

$$\log \frac{p(\boldsymbol{i} \mid t)}{p(\boldsymbol{i} \mid t^*)} = \log \frac{p(t \mid \boldsymbol{i})}{p(t^* \mid \boldsymbol{i})} - \log \frac{\pi(t)}{\pi(t^*)}, \tag{18}$$

so (17) follows by taking $c_t = -\log(\pi(t)/\pi(t^*))$. More generally, (17) is the factor-level calibration condition associated with the latent-task factorization used in the proofs. Output-level cross-entropy optimality in (13) motivates this idealized calibration but does not, by itself, identify a particular latent decomposition; the latter is part of the reference-model specification.

The Transformer-induced scoring map is not claimed to be statistically identified by next-token cross-entropy. It is a concrete architecture-induced reference extension used to instantiate and

quantitatively control the otherwise arbitrary finite off-support perturbation. Supported calibration is stated explicitly in (17) as a separate condition.

At an unsupported U, Equation (15) defines a model-selected value. We do not interpret it as the probability of observing a continuous embedding under the original discrete pretraining distribution. This distinction resolves the apparent conflict between $p(\widetilde{\boldsymbol{i}} \mid t^*) = 0$ and the fact that a Transformer nevertheless returns well-defined scores when fed $E(\widetilde{\boldsymbol{i}})$ or an interpolated instruction embedding.

Thus $r_t(U)$, rather than either individual factor $p(U \mid t)$, is the continuous object used in the analysis. It agrees with the ordinary likelihood ratio on the supported symbolic points by (17) and is defined by the scoring map off that support. For example, in the shared-template setting the continuation-induced task posterior is determined by the odds

$$\frac{p_{\mathrm{ref}}(t \mid \boldsymbol{z}_N(U))}{p_{\mathrm{ref}}(t^* \mid \boldsymbol{z}_N(U))} := B_t \exp(r_t(U) + \lambda_t), \qquad t \neq t^*, \tag{19}$$

followed by normalization over the finite task set. Here B_t is the demonstration likelihood ratio and λ_t contains the prior and query factors, as defined in (54) and (76). The associated reference output predictor is

$$p_{\mathrm{ref}}(y \mid \boldsymbol{z}_N(U)) := \sum_{t \in \mathcal{T}} p(y \mid x_{\mathrm{query}}, t) p_{\mathrm{ref}}(t \mid \boldsymbol{z}_N(U)). \tag{20}$$

Thus the Transformer supplies exactly the single off-support factor needed by the Bayesian concentration proof; the supported demonstration, prior, and query factors retain their original probabilistic meaning.

C.3 Analyticity of standard Transformer components

Fix the sequence length, positional convention, attention mask, and computational graph, and regard the instruction embedding U as the only continuous input. Affine maps, matrix products, residual additions, and fixed positional embeddings are analytic. The exponential, hyperbolic tangent, error function, exact GELU, and SiLU are real analytic. Hence attention softmax and an output softmax are real analytic at every real input, because their normalizing denominators are strictly positive on the real domain.

Layer normalization with a fixed $\epsilon > 0$ has the form

$$\mathrm{LN}(x) = \gamma \odot \frac{x - \mu(x)}{\sqrt{\sigma^2(x) + \epsilon}} + \beta. \tag{21}$$

Since $\sigma^2(x) + \epsilon \geq \epsilon > 0$ for real x, it is real analytic on the entire real domain. The same argument applies to RMS normalization with a strictly positive stabilizer. A causal mask is harmless when it is fixed: masked entries are omitted from the corresponding finite softmax, rather than treated as a variable arithmetic input.

Proposition C.1 (Analytic Transformer scoring map)**.** *Suppose that the fixed Transformer scoring map in (14) has finite depth and width and is constructed from the analytic operations above. Suppose also that every normalization uses a strictly positive stabilizer and that the sequence length, masks, and computational graph remain fixed. Then each map $U \mapsto a_t(U)$, and hence each $U \mapsto r_t(U)$, is real analytic on $\mathbb{R}^{L_i \times d}$.*

Proof. Each hidden state is obtained from the preceding hidden states by finitely many sums, products, quotients with nonzero real denominators, and compositions of real-analytic functions. These operations preserve real analyticity. An induction over the finitely many layers proves that every hidden state and output score is real analytic in U. Equation (15) then gives the result for the reference log-odds. □

Proposition C.1 is an architectural statement, not an assumption that follows from data.

C.4 From architectural analyticity to the finite continuation chain

Let

$$E_s = (1-s)E(\boldsymbol{i}_0) + sE(\widetilde{\boldsymbol{i}}), \qquad s \in [0,1], \tag{22}$$

be the path used in Theorem 3.3. The following result shows that the finite-chain hypothesis is automatic for the scoring-map-induced extension above.

Proposition C.2 (Analytic Transformer implies a finite continuation chain)**.** *Under Proposition C.1, for every $t \neq t^*$ the function*

$$s \longmapsto r_t(E_s) \tag{23}$$

is real analytic on an open real neighborhood of $[0,1]$ and admits a finite compatible chain of holomorphic continuations of the form required by Assumption D.15. The chain is independent of the number N of demonstrations.

Proof. The map $s \mapsto E_s$ is affine and $U \mapsto r_t(U)$ is real analytic, so their composition in (23) is real analytic on an open neighborhood of every point of $[0,1]$. Every real-analytic germ has a local holomorphic complexification. These complexifications agree wherever their domains overlap along the real interval, by uniqueness of analytic continuation. The resulting complex neighborhoods form an open cover of the compact interval $[0,1]$. Compactness gives a finite subcover; choosing a sufficiently fine increasing sequence of real centers $0 = s_0 < \cdots < s_L = 1$ and disks inside this subcover yields radii r_ℓ with $s_{\ell+1} - s_\ell < r_\ell$ and compatible holomorphic functions on the disks, exactly as in Assumption D.15. The function, path, and finite cover concern a single shared instruction template and do not depend on N. □

Complexifications of softmax, SiLU, and normalization may have zeros, poles, or branch points away from the real domain. This does not create an additional existence assumption in Proposition C.2: at every real point of the path, the real-analytic computation supplies some complex neighborhood free of such singularities, and compactness reduces these neighborhoods to a finite chain. Nearby complex singularities do, however, shrink the admissible radii and can enlarge the Cauchy constants M_ℓ and the cumulative continuation cost Ω defined in Appendix D.5. Thus architectural analyticity guarantees existence of the chain, whereas quantitative separation from the complex singular set governs the strength of the resulting bound.

For an abstract off-support log-odds extension not parameterized by such a Transformer scoring map, none of the preceding architectural argument is available; in that case the finite-continuation condition remains a substantive regularity hypothesis. The main theorem requires only finite, N-independent endpoint log-odds for the chosen extension; Proposition C.2 supplies a concrete model-induced construction and quantitative control of those endpoint values.

C.5 The actual model and the transport error

The construction does not assume that the actual pretrained model M equals the idealized reference p_{ref}. On supported prompts their initial discrepancy is r_0 from (31). Along a compact prompt path γ, Appendix D.1 gives

$$|M(y \mid \gamma(1)) - p_{\mathrm{ref}}(y \mid \gamma(1))| \le \underbrace{r_0 + (L_M + L_{\mathrm{ref}})\,\mathrm{len}(\gamma)}_{\varepsilon_M(\gamma)}. \tag{24}$$

The analytic continuation controls the reference term in the main theorems; ε_M records the logically separate approximation and transport error for the actual model. Analyticity guarantees that the displayed Lipschitz constants are finite on a fixed compact path, but does not guarantee that they, r_0, or ε_M are small. Appendix D.1.1 gives an additional quantitative result: supported-anchor calibration and length-independent network stability imply the local certificate $\bar{r} + (C_M + 2G)d$, uniformly over the specified length family. Appendix D.1.2 relates calibration to population excess loss under an explicit coverage condition, and Appendix D.1.3 gives a sharper certificate using variation of the model–reference residual. These sufficient conditions are separate from the existence of analytic continuation.

When the reference predictor can be evaluated, this discrepancy can also be studied empirically. For a grid $0 = u_0 < \cdots < u_K = 1$, define

$$\widehat{\varepsilon}_M := \max_{0\le k\le K} \sup_y |M(y \mid \gamma(u_k)) - p_{\mathrm{ref}}(y \mid \gamma(u_k))| . \tag{25}$$

If the largest path-segment length $\max_k \mathrm{len}(\gamma|_{[u_k,u_{k+1}]})$ is δ and the two predictors have pathwise Lipschitz constants L_M and L_{ref}, then every path point is within path length at most δ of a grid point and

$$\sup_{s\in[0,1]} \sup_y |M(y \mid \gamma(s)) - p_{\mathrm{ref}}(y \mid \gamma(s))| \le \widehat{\varepsilon}_M + (L_M + L_{\mathrm{ref}})\delta. \tag{26}$$

Without computable Lipschitz bounds, the grid experiment supplies evidence rather than a uniform certificate. Without access to an evaluable reference predictor, ε_M remains a theoretical approximation term and cannot be identified from ordinary test accuracy alone.

C.6 Empirical validation of the model–reference approximation

We complement the preceding argument with a controlled experiment in which the latent-task probabilities and the hybrid reference predictor are evaluable. The purpose is not to test analyticity itself, which follows from the architectural conditions of Proposition C.1, but to test the substantive quantitative premise behind the model-level bounds: whether the actual Transformer remains close to the reference predictor along the continuation path, and whether this discrepancy stays controlled as the number of demonstrations increases.

Controlled setting. We use a finite latent-task benchmark with three modular-arithmetic tasks (addition, subtraction, and multiplication). The task prior, supported task-conditional instruction probabilities, demonstration likelihoods, and task-conditional output distributions are known, so the factors B_t and λ_t in (19) and the resulting p_{ref} can be computed directly. A finite Transformer with analytic components is trained by cross-entropy using supported instructions only, and an analytic task-scoring map supplies r_t. We repeat the experiment for five independently initialized models.

For a supported anchor $\boldsymbol{i}_0$ and an instruction endpoint $\widetilde{\boldsymbol{i}}$, we evaluate 51 equally spaced points on

$$E_s = (1-s)E(\boldsymbol{i}_0) + sE(\widetilde{\boldsymbol{i}}), \qquad s \in [0,1], \tag{27}$$

and use $N \in \{1, 2, 4, 8, 16, 32, 64\}$ demonstrations. To avoid confusion with the working radii ρ_ℓ used in the continuation proof, denote normalized endpoint distance by

$$\delta_{\mathrm{emb}} := \frac{\|E(\widetilde{\boldsymbol{i}}) - E(\boldsymbol{i}_0)\|_F}{\text{typical separation between supported instruction embeddings}}. \tag{28}$$

For each N, define the endpoint and pathwise model–reference gaps

$$d_N^{\mathrm{end}} := \sup_y |M(y \mid \gamma_N(1)) - p_{\mathrm{ref}}(y \mid \gamma_N(1))| , \tag{29}$$

$$d_N^{\mathrm{path}} := \max_{s\in\{0,.02,\ldots,1\}} \sup_y |M(y \mid \gamma_N(s)) - p_{\mathrm{ref}}(y \mid \gamma_N(s))| . \tag{30}$$

Thus d_N^{path} is an empirical effective transport error, rather than the analytic upper bound in (24).

Local-path discrepancies over the tested lengths. Table 3 reports the complete trajectories for two local paths. At $\delta_{\mathrm{emb}} = 0.25$, the pathwise gap remains between 0.058 and 0.125 over a 64-fold change in demonstration count. It is 0.099 at $N = 64$, approximately 20% below its value at $N = 1$. Moreover, the measured supported-anchor gap accounts for most of this discrepancy: across the displayed values of N, the additional pathwise gap $d_N^{\mathrm{path}} - r_0$ is only 0.007–0.037. This provides evidence of a small additional measured error on these short paths. Here the reported r_0 is an empirical anchor counterpart, not an evaluation of the supremum over the full supported domain in (31).

At $\delta_{\mathrm{emb}} = 0.50$, the pathwise gap remains between 0.139 and 0.316 and again ends below its $N = 1$ value. Neither trajectory is monotone, as expected for finite prompts and independently sampled demonstrations, but neither exhibits systematic amplification with N. These are precisely the local paths for which the model-level theorem is most informative: the Bayesian term decreases exponentially while the empirical transport term remains controlled.

Table 3: **Model–reference discrepancy on local paths.** Mean ± standard deviation over five independently trained models. The empirical supported-anchor gap (denoted r_0 in this table) is common to the endpoint-distance conditions.

N	r_0	d_N^{path} ($\delta_{\mathrm{emb}} = .25$)	d_N^{path} ($\delta_{\mathrm{emb}} = .50$)
1	0.088 ± 0.046	0.125 ± 0.081	0.253 ± 0.170
2	0.056 ± 0.078	0.069 ± 0.087	0.146 ± 0.151
4	0.064 ± 0.082	0.071 ± 0.080	0.139 ± 0.113
8	0.066 ± 0.059	0.099 ± 0.084	0.316 ± 0.175
16	0.035 ± 0.024	0.058 ± 0.038	0.182 ± 0.127
32	0.090 ± 0.102	0.115 ± 0.127	0.206 ± 0.179
64	0.071 ± 0.045	0.099 ± 0.086	0.192 ± 0.157

Table 4: **Dependence on normalized endpoint distance.** Pathwise model–reference gap at the smallest and largest tested demonstration counts. Mean ± standard deviation over five independently trained models; the last column is the ratio of the two means.

δ_{emb}	d_1^{path}	d_{64}^{path}	$d_{64}^{\mathrm{path}}/d_1^{\mathrm{path}}$
0.25	0.125 ± 0.081	0.099 ± 0.086	0.799
0.50	0.253 ± 0.170	0.192 ± 0.157	0.757
1.00	0.323 ± 0.270	0.459 ± 0.189	1.421
2.00	0.413 ± 0.229	0.543 ± 0.081	1.315
4.00	0.525 ± 0.057	0.562 ± 0.057	1.072

Distance controls the error floor. Table 4 expands the endpoint distance from one quarter to four times the typical supported-instruction separation. The comparison reveals a favorable and interpretable structure. For the two local paths, the gap at $N = 64$ is smaller than at $N = 1$. At larger distances, the gap increases between these two demonstration counts, but remains bounded and the increase is modest relative to the change caused by moving the endpoint farther from the supported anchor. For example, at $\delta_{\mathrm{emb}} = 4$, the gap changes from 0.525 to 0.562 as N increases from 1 to 64. Across distances, the dominant pattern is thus a distance-dependent error floor, as represented explicitly by $\varepsilon_{M,N}(\widetilde{\boldsymbol{i}})$ in the theorem, rather than systematic amplification with the number of demonstrations.

The endpoint and pathwise gaps are also nearly identical throughout the two local conditions and for $\delta_{\mathrm{emb}} = 1$ except at the smallest demonstration count. Hence the maximum discrepancy is generally attained near the unsupported endpoint, rather than at an unstable interior point of the interpolation. This behavior is consistent with the finite-chain picture: the path remains regular, while moving the endpoint farther from supported instructions increases the distance-dependent difficulty of model transport.

What the experiment validates. The measured anchor and path-grid discrepancies support accurate local tracking in this controlled setting over the tested demonstration counts. Larger distances generally produce larger discrepancies. These are means and standard deviations over five trained models, not worst-case bounds over models, demonstration sequences, or the continuum of path points. Small d_N^{path} does not establish that the analytic upper certificate $\varepsilon_{M,N}(\widetilde{\boldsymbol{i}})$ is small: the latter uses separate sensitivity bounds that the experiment does not certify. Nor does the tested length grid establish an almost-sure statement for all sufficiently large N along one infinite demonstration sequence.

Proposition D.7 identifies a sufficient regime in which the analytic certificate is uniformly small, but its network constants and calibration premises have not been certified for these trained models. A continuous-path certificate additionally requires a valid Lipschitz envelope in (26), or a certified residual-variation bound from Proposition D.9. Extending the diagnostic to real models also requires an evaluable reference; task accuracy alone does not identify the discrepancy. Thus the experiment provides finite-range evidence compatible with the sufficient conditions, rather than verification of their uniform hypotheses for general LLMs.

In summary, the setting makes two different commitments. Supported calibration connects the reference scores to the latent-task probabilities at prompts that pretraining can identify. Analytic Transformer architecture then selects a well-defined, finite-chain extension of those scores along a instruction-embedding path. The first commitment is statistical and support-restricted; the second is architectural and off-support. Their separation is essential: analyticity provides a principled extension, but does not turn zero-probability prompts into observations from the pretraining distribution.

D Proof of Section 3

D.1 From the reference predictor to the pretrained model

Let $M(y \mid \boldsymbol{z})$ denote the output probability of the actual pretrained model, and let $p_{\mathrm{ref}}(y \mid \boldsymbol{z})$ denote the theoretical reference: it equals the Bayesian predictor p on supported prompts and the fixed continued predictor $\bar{p}$ on unsupported prompts. Define

$$r_0 := \sup_{\boldsymbol{z} \in D_0} \sup_y |M(y \mid \boldsymbol{z}) - p(y \mid \boldsymbol{z})|. \tag{31}$$

Consider a compact instruction-embedding path γ from a supported prompt $\gamma(0)$ to a test prompt $\gamma(1)$. Suppose $M(y \mid \gamma(s))$ and $p_{\mathrm{ref}}(y \mid \gamma(s))$ are Lipschitz along this path, uniformly in y, with constants L_M and L_{ref}, and let $\mathrm{len}(\gamma)$ be its length. These constants are finite, for example, if both functions are real analytic on a neighborhood of the compact path. Define the model-transport error

$$\varepsilon_M(\gamma) := r_0 + (L_M + L_{\mathrm{ref}}) \,\mathrm{len}(\gamma). \tag{32}$$

Lemma D.1 (Transport from the supported domain)**.** *Under the preceding conditions,*

$$|M(y \mid \gamma(1)) - p_{\mathrm{ref}}(y \mid \gamma(1))| \le \varepsilon_M(\gamma). \tag{33}$$

For a supported test prompt, the constant path gives $\varepsilon_M = r_0$.

Proof. Insert and subtract the two functions at the supported endpoint:

$$\begin{aligned} &|M(y \mid \gamma(1)) - p_{\mathrm{ref}}(y \mid \gamma(1))| && (34)\\ &\quad \le |M(y \mid \gamma(0)) - p(y \mid \gamma(0))| + |M(y \mid \gamma(1)) - M(y \mid \gamma(0))| && (35)\\ &\qquad + |p_{\mathrm{ref}}(y \mid \gamma(1)) - p_{\mathrm{ref}}(y \mid \gamma(0))| && (36)\\ &\quad \le r_0 + (L_M + L_{\mathrm{ref}}) \,\mathrm{len}(\gamma). && (37) \end{aligned}$$

□

For the shared-template setting, fix the instruction-coordinate path

$$E_s = (1-s)E(\boldsymbol{i}_0) + sE(\widetilde{\boldsymbol{i}}), \qquad \gamma_N(s) := \boldsymbol{z}_N(E_s).$$

The copies of the repeated instruction in $\gamma_N(s)$ are tied to the same coordinate E_s. Let $L_{M,N}$ and $L_{\mathrm{ref},N}$ satisfy, for all $s, s' \in [0,1]$,

$$\sup_y |M(y \mid \boldsymbol{z}_N(E_s)) - M(y \mid \boldsymbol{z}_N(E_{s'}))| \le L_{M,N} \|E_s - E_{s'}\|_F, \tag{38}$$

$$\sup_y |p_{\mathrm{ref}}(y \mid \boldsymbol{z}_N(E_s)) - p_{\mathrm{ref}}(y \mid \boldsymbol{z}_N(E_{s'}))| \le L_{\mathrm{ref},N} \|E_s - E_{s'}\|_F. \tag{39}$$

Define

$$r_{0,N} := \sup_y |M(y \mid \boldsymbol{z}_N(\boldsymbol{i}_0)) - p(y \mid \boldsymbol{z}_N(\boldsymbol{i}_0))| \tag{40}$$

and the shared-template transport error

$$\varepsilon_{M,N}(\widetilde{\boldsymbol{i}}) := r_{0,N} + (L_{M,N} + L_{\mathrm{ref},N}) \|E(\widetilde{\boldsymbol{i}}) - E(\boldsymbol{i}_0)\|_F. \tag{41}$$

The repeated copies therefore do not introduce an artificial $\sqrt{N+1}$ factor into the path length; any dependence on their number is recorded in the pullback Lipschitz constants.

Corollary D.2 (Shared-template transport at fixed demonstration count)**.** *For every fixed N,*

$$\sup_y \left| M(y \mid \boldsymbol{z}_N(\widetilde{\boldsymbol{i}})) - \bar{p}(y \mid \boldsymbol{z}_N(\widetilde{\boldsymbol{i}})) \right| \le \varepsilon_{M,N}(\widetilde{\boldsymbol{i}}). \tag{42}$$

Proof. Apply the triangle-inequality argument of Lemma D.1 to γ_N, using $p_{\mathrm{ref}} = p$ at the supported anchor and the tied instruction coordinate to measure the path length. □

For the mixed setting, $\varepsilon_M(\boldsymbol{z}_N)$ is the quantity in (32) on a path that changes the unsupported instruction slots successively. It is finite for each fixed prompt but need not be uniform in N or converge to zero. Combining Lemma D.1 with the continuation bounds proved below gives the four model-level bounds stated in the main text.

D.1.1 Quantitative conditions for uniform local transport

The transport inequality is an upper certificate, not the actual discrepancy. Write

$$D_N := \sup_{s\in[0,1]} \sup_{y} |M(y \mid \boldsymbol{z}_N(E_s)) - p_{\mathrm{ref}}(y \mid \boldsymbol{z}_N(E_s))|.$$

Then $D_N \le \varepsilon_{M,N}$, but a small measured D_N does not imply a small value of the particular upper bound in (41). Even when the two predictors agree everywhere on a path, their separate Lipschitz constants can both be large. We give sufficient conditions for the upper bound itself to be uniform and locally small, followed by a sharper certificate based on their difference.

Why calibration and analyticity alone do not suffice. For finitely many distinct supported embeddings $a_1, \dots, a_m$ and an endpoint b different from all of them, the polynomial

$$q(E) := \prod_{j=1}^{m} \frac{\|E - a_j\|_F^2}{\|b - a_j\|_F^2}$$

vanishes on every a_j and equals one at b. Adding $Aq(E)$ to one output logit of an analytic predictor preserves its predictions at the supported embeddings but can change its prediction at b arbitrarily strongly as $|A|$ increases. This demonstrates the insufficiency of finite symbolic calibration and real analyticity alone; it does not assert exact representation of this polynomial by a particular fixed Transformer. Quantitative stability or further restrictions on the learned predictor are needed.

Length-independent stability in a tokenwise norm. For token representations $H = (h_1, \dots, h_m)$, set $\|H\|_{\infty,2} := \max_j \|h_j\|_2$. Comparisons below are between prompts of the same length and with the same mask; only the tied instruction changes. The bounds will be independent of that length.

Lemma D.3 (Stability of an attention head)**.** *Let $A_i(H) = \sum_{j\in I_i} a_{ij}(H) W_V h_j$, where I_i is a fixed nonempty mask and a_{ij} is the softmax of $\langle W_Q h_i, W_K h_j\rangle/\sqrt{d_k}$ over I_i. If $\|H\|_{\infty,2}, \|H'\|_{\infty,2} \le B$, and $q = \|W_Q\|_2$, $k = \|W_K\|_2$, $v = \|W_V\|_2$, then*

$$\|A(H) - A(H')\|_{\infty,2} \le v\left(1 + \frac{4qkB^2}{\sqrt{d_k}}\right)\|H - H'\|_{\infty,2}. \tag{43}$$

The bound is independent of the number of tokens and of $|I_i|$.

Proof. Put $\delta = \|H - H'\|_{\infty,2}$. Adding and subtracting a mixed inner product bounds each logit difference by $2qkB\delta/\sqrt{d_k}$. For a softmax vector a, its Jacobian maps w to $(a_j(w_j - \sum_l a_l w_l))_j$, whose ℓ_1 norm is at most $2\|w\|_\infty$. Integration along the segment of logits gives $\|a_i(H) - a_i(H')\|_1 \le 4qkB\delta/\sqrt{d_k}$. The change in values contributes at most $v\delta$, and the change in weights at most $vB\|a_i(H) - a_i(H')\|_1$. Take the maximum over i. A fixed additive position bias cancels in the logit difference. □

Assumption D.4 (Uniform architectural stability on the path family)**.** Let $\mathcal{N}$ be the set of admissible demonstration counts. The model has fixed depth and head counts. On all paths $\boldsymbol{z}_N(E_s)$, $N \in \mathcal{N}$, and all teacher-forced output prefixes of length at most a fixed $H_\star$:

1. Inputs to each attention head have a common token-norm bound B_ℓ independent of N, s, and the prefix. Projection norms are bounded independently of N. Heads have the form of Lemma D.3, or obey its estimate with uniform constants; bounded affine biases may be included in those constants.

2. Tokenwise normalization and feed-forward maps, output projections, position-processing maps, and residual and multi-head combination maps are Lipschitz on the relevant domains with constants independent of N. The input representation satisfies $\|H_N(E) - H_N(E')\|_{\infty,2} \le C_{\mathrm{in}}\|E - E'\|_F$ on the path with C_{in} independent of N.

3. Outputs use fixed decoding probabilities and have length at most $H_\star$, including any termination token. The generation kernel is defined at every teacher-forced prefix used above.

The input inequality holds with $C_{\mathrm{in}} = 1$ for direct insertion of tied embedding copies and unchanged additive positions. A positive fixed normalization stabilizer and bounded normalization parameters give finite tokenwise normalization constants. These observations do not establish all the other uniform bounds for every architecture or length regime. The resulting constants need not be numerically small, and greedy decoding is not a Lipschitz probability map.

Proposition D.5 (Uniform model sensitivity along tied paths)**.** *Under Assumption D.4, there is a finite C_M, independent of $N \in \mathcal{N}$, such that*

$$\sup_y |M(y \mid \boldsymbol{z}_N(E_s)) - M(y \mid \boldsymbol{z}_N(E_{s'}))| \le C_M\|E_s - E_{s'}\|_F.$$

Proof. Apply Lemma D.3 to each head. A residual map $H \mapsto H + F(H)$ has constant at most $1 + L_F$. Multi-head concatenation has constant at most the square root of the sum of squared head constants, followed by the output-projection norm. Composing these with the remaining uniform bounds gives a next-token logit constant $C_{\log}$ independent of N and the forced prefix. The softmax estimate above gives a probability-coordinate constant at most $2C_{\log}$. For a fixed block y, compare its autoregressive factors at the same forced prefixes. The difference of two products of numbers in $[0, 1]$ is at most the sum of the factor differences. Thus $C_M = 2H_\star C_{\log}$ suffices. □

Lemma D.6 (Uniform reference sensitivity)**.** *Let $d = \|E(\widetilde{\boldsymbol{i}}) - E(\boldsymbol{i}_0)\|_F$ and suppose $|\frac{d}{ds} r_t(E_s)| \le Gd$ for every competing task on the fixed path. Suppose the continued task weights are*

$$w_{t,N}(s) = \frac{c_{t,N}\exp(r_t(E_s))}{1 + \sum_{u \ne t^*} c_{u,N}\exp(r_u(E_s))},$$
$$w_{t^*,N}(s) = \frac{1}{1 + \sum_{u \ne t^*} c_{u,N}\exp(r_u(E_s))},$$

where $c_{t,N}$ are nonnegative and finite and do not depend on s. Then $L_{\mathrm{ref},N} \le 2G$ is an admissible choice for every N. A finite G exists when the finitely many continued scores are continuously differentiable on a neighborhood of the fixed compact path.

Proof. Put $a_t = p(y \mid x_{\text{query}}, t) \in [0, 1]$ and $a_N(s) = \sum_t w_{t,N}(s)a_t$. Differentiating the normalized weights yields

$$a'_N(s) = \sum_{t \ne t^*} w_{t,N}(s)(a_t - a_N(s))\frac{d}{ds} r_t(E_s).$$

Thus $|a'_N(s)| \le Gd\sum_{t \ne t^*} w_{t,N}(s) \le Gd$. Integration and $\|E_s - E_{s'}\|_F = d|s - s'|$ prove even the constant G; we retain $2G$ as a common upper bound. For $d = 0$ the path is constant. For $d > 0$, compactness and the finite task set bound the directional derivatives divided by d. □

Proposition D.7 (Uniform local transport certificate)**.** *Under Proposition D.5 and Lemma D.6, suppose the supported anchors satisfy $\sup_{N \in \mathcal{N}} r_{0,N} \le \bar{r}$. Admissible constants in (41) can then be chosen so that*

$$\sup_{N \in \mathcal{N}} D_N \le \sup_{N \in \mathcal{N}} \varepsilon_{M,N}(\widetilde{\boldsymbol{i}}) \le \bar{r} + (C_M + 2G)d. \tag{44}$$

For $\epsilon > \bar{r}$ and $C_M + 2G > 0$, a path with $d \le (\epsilon - \bar{r})/(C_M + 2G)$ has a certificate at most ϵ. If $C_M + 2G = 0$, its certificate is at most $\bar{r}$ at every admissible distance.

Proof. Apply the fixed-N transport argument to every initial path segment. Substitute $L_{M,N} \le C_M$ and $L_{\mathrm{ref},N} \le 2G$, then take the supremum over $N \in \mathcal{N}$. □

The proposition is deterministic conditional on the demonstration family. If its hypotheses hold on a probability-one event for prefixes of one infinite demonstration sequence, the conclusion is almost sure and uniform. For a fixed context window, $\mathcal{N}$ contains only lengths that fit; no $N \to \infty$ claim is then made. A common radius for a neighborhood of endpoints requires common C_M, G over that neighborhood. The proposition does not certify that the experimental endpoints fall inside a numerically useful radius.

D.1.2 Supported calibration from population loss and coverage

Proposition D.8 (Calibration on a covered anchor family)**.** *Let Q be a distribution on supported prompts with true conditional p, and use natural logarithms. Define its population excess log loss by*

$$\Delta := \mathbb{E}_{Z\sim Q} D_{\mathrm{KL}}(p(\cdot \mid Z) \| M(\cdot \mid Z)).$$

For a finite anchor family $\mathcal{A}$ with $Q(z) \geq q_{\min} > 0$ for all $z \in \mathcal{A}$,

$$\sup_{z\in\mathcal{A},y} |M(y \mid z) - p(y \mid z)| \leq \min\{1, \sqrt{\Delta/(2q_{\min})}\}. \tag{45}$$

If all anchors in Proposition D.7 belong to $\mathcal{A}$, its bound holds with $\bar{r} = \min\{1, \sqrt{\Delta/(2q_{\min})}\}$.

Proof. Nonnegativity gives $Q(z) D_{\mathrm{KL}}(p(\cdot \mid z) \| M(\cdot \mid z)) \leq \Delta$. For completeness, for discrete probability distributions P, R let $A = \{y : P(y) \geq R(y)\}$. The log-sum inequality gives $D_{\mathrm{KL}}(P\|R) \geq D_{\mathrm{KL}}(\mathrm{Ber}(P(A)) \| \mathrm{Ber}(R(A)))$. For fixed $b \in (0,1)$ the binary divergence as a function of a has second derivative $1/[a(1-a)] \geq 4$ and vanishes with zero first derivative at $a = b$. It is at least $2(a-b)^2$; boundary cases follow by limits. Since $P(A) - R(A) = \|P - R\|_{\mathrm{TV}}$, this proves Pinsker's inequality. Apply it at each anchor and bound a coordinate difference by total variation. □

Here Δ is excess *population* loss, not empirical training loss. A learning-theoretic analysis would also have to control estimation, approximation, and optimization error. Coverage is substantive: a probability distribution cannot assign a common positive mass to infinitely many distinct prompts. This finite-family corollary therefore does not establish uniform calibration over all lengths from one average pretraining-loss bound.

For a distributional alternative, suppose a target anchor law R_N is dominated by Q with $dR_N/dQ \leq C$ over the chosen lengths. Pinsker's inequality and Cauchy–Schwarz give

$$\mathbb{E}_{R_N} \sup_y |M(y \mid Z) - p(y \mid Z)| \leq \sqrt{C\Delta/2}.$$

With a common path-stability bound and path length at most d, expected off-support discrepancy is at most $\sqrt{C\Delta/2} + (C_M + 2G)d$. This is an expectation guarantee for each N, not an almost-sure bound on the supremum over N. The density-ratio premise must itself be justified.

D.1.3 A sharper certificate based on residual variation

Proposition D.9 (Residual-variation transport)**.** *Suppose $g_N(s,y) := M(y \mid \boldsymbol{z}_N(E_s)) - p_{\mathrm{ref}}(y \mid \boldsymbol{z}_N(E_s))$ is absolutely continuous in s for each output, with an integrable common envelope for its derivatives. Define*

$$\varepsilon^{\mathrm{res}}_{M,N} := r_{0,N} + \int_0^1 \sup_y |\partial_s g_N(s,y)|\, ds. \tag{46}$$

Then $D_N \leq \varepsilon^{\mathrm{res}}_{M,N}$. With the same Lipschitz constants as in (41), $\varepsilon^{\mathrm{res}}_{M,N} \leq \varepsilon_{M,N}$.

Proof. The fundamental theorem of calculus bounds $|g_N(s,y)|$ by $|g_N(0,y)| + \int_0^s |\partial_u g_N(u,y)|\, du$. Take suprema and extend the integral to $[0,1]$. The individual derivative bounds are $L_{M,N} d$ and $L_{\mathrm{ref},N} d$ almost everywhere, proving the second inequality. □

Every endpoint model bound obtained by transport remains valid with $\varepsilon^{\mathrm{res}}_{M,N}$ replacing the coarser certificate. A uniform bound on these residual certificates gives a uniform model bound without asserting a uniform bound on the original certificates. If the predictors agree throughout the path, the residual certificate is zero. Path-grid errors or sampled derivatives alone do not certify the integral; an envelope or a controlled discretization error is still required.

D.2 COMPLETE PROMPT FORMATS

We expand all seven rows of Table 1 below. The first axis is *task-conditional support*. An instruction may be familiar globally yet unsupported for the target task: support requires $p(i \mid t^*) > 0$, meaning that pretraining paired that rendering with t^* with positive probability. Unsupported instructions satisfy $p(\tilde{i} \mid t^*) = 0$. This is not a judgment of semantic correctness. A dialectal expression or private schema label can express the user's intended task perfectly while remaining unsupported by the pretraining distribution.

The second axis is *placement*. A single instruction appears once above the prompt; a repeated instruction is one realization copied into every demonstration and the query; varying instructions are separately generated for the individual slots. Formats 1, 3, and 5 draw their instructions from $P_{i|t^*}$ and are supported by construction. Formats 2, 4, and 6 use renderings outside that distribution's support. Format 7 begins with a repeated supported template and corrupts only a subset of its occurrences. These cases correspond respectively to ordinary user or system messages, record-level schemas, retrieval or multi-annotator pipelines, and partially corrupted data.

1. **Single supported instruction.** A single supported instruction i appears once at the start of the prompt and applies to all demonstrations and the query.

   $$\boldsymbol{z} = (i, (x_1, y_1), \ldots, (x_N, y_N), x_{\text{query}}).$$

   Concrete example. A customer-support system places *Classify each ticket as billing, technical, or account* once above several labeled tickets. This conventional instruction has been paired with ticket classification during pretraining and is modeled as a draw from $P_{i|t^*}$.

2. **Single unsupported instruction.** A single instruction $\tilde{i}$ outside the task-conditional support appears once at the start of the prompt. The absent case is recovered by setting $\tilde{i} = \langle\text{pad}\rangle$.

   $$\boldsymbol{z} = (\tilde{i}, (x_1, y_1), \ldots, (x_N, y_N), x_{\text{query}}),$$

   where $\tilde{i}$ is an unsupported instruction or a virtual token $\langle\text{pad}\rangle$.

   Concrete example. An Indian-English user writes *Please revert with a reply*, intending *revert* to mean "reply," and provides demonstrations mapping incoming emails to reply drafts. The wording is correct for the user, but if pretraining paired *revert* with operations such as "undo" and never with reply drafting, then $p(\tilde{i} \mid t^*) = 0$ in our idealized model.

3. **Repeated supported instruction.** One supported instruction i is copied into every demonstration and the query, so the model encounters the same realization $N + 1$ times.

   $$\boldsymbol{z} = ((i, x_1, y_1), \ldots, (i, x_N, y_N), (i, x_{\text{query}})).$$

   Concrete example. A JSON dataset includes `"task": "sentiment"` in every demonstration record and in the query record. The standard schema key is supported for sentiment classification, and all displayed occurrences are copies of one shared template rather than independent instruction samples.

4. **Repeated unsupported instruction.** One unsupported (or pad) instruction $\tilde{i}$ is copied into every demonstration and the query.

   $$\boldsymbol{z} = ((\tilde{i}, x_1, y_1), \ldots, (\tilde{i}, x_N, y_N), (\tilde{i}, x_{\text{query}})).$$

   Concrete example. A company repeatedly inserts `"operation": "Atlas"` into records for its internal ticket-routing service. The name was borrowed from an unrelated open-source project and was never paired with the company's current routing behavior during pretraining. Although the label is meaningful inside the company, it is unsupported for t^* and incurs only one shared instruction factor because the same label is copied throughout.

5. **Varying supported instructions.** Each demonstration is paired with an independently sampled supported rendering i_n, and the query carries another independently sampled rendering i_{query}.

   $$\boldsymbol{z} = ((i_1, x_1, y_1), \ldots, (i_N, x_N, y_N), (i_{\text{query}}, x_{\text{query}})).$$

   Concrete example. A retrieval system pools sentiment examples written by different annotators: *classify the sentiment*, *positive or negative?*, and *identify the review's tone*. These are distinct but familiar renderings sampled from $P_{i|t^*}$, so every slot is supported even though the surface wording varies.

6. **Varying unsupported instructions.** Each demonstration is paired with a distinct unsupported rendering $\tilde{i}_n$, and the query carries an analogous $\tilde{i}_{\text{query}}$. Each slot can therefore introduce a separate off-support cost.

$$\boldsymbol{z} = ((\tilde{i}_1, x_1, y_1), \ldots, (\tilde{i}_N, x_N, y_N), (\tilde{i}_{\text{query}}, x_{\text{query}})).$$

Concrete example. A multi-tenant retrieval pipeline combines ticket-routing demonstrations from companies that use private labels such as `triage_acme`, `route_7G`, and `dispatch_private`. Each tenant intends its label to denote the same routing task, but none of the private labels was paired with that task during pretraining. Unlike format 4, the prompt introduces a new unsupported label in each slot.

7. **Mixed supported and unsupported instructions.** Most slots retain one shared supported instruction i, while a subset contains distinct unsupported corruptions $\tilde{i}_n$:

$$\boldsymbol{z} = \big((i_n, x_n, y_n)_{n=1}^N, (i_{\text{query}}, x_{\text{query}})\big), \qquad i_n \in \{i, \tilde{i}_n\}.$$

Concrete example. A sentiment dataset normally repeats the standard field `sentiment`, but manual entry or OCR produces distinct typos such as `sentimnet` and `sentimemt` in a few records. The unchanged records retain a supported anchor, whereas each typo is treated as an unsupported corruption. This differs from format 6, in which every slot carries a distinct unsupported rendering.

D.3 Preliminaries

We begin by expanding the posterior distribution over tasks using Bayes' rule. For any task $t \in \mathcal{T}$, conditioning on the full prompt $\boldsymbol{z} = (\boldsymbol{i}, \{(x_n, y_n)\}_{n=1}^N, x_{\text{query}})$ yields

$$p(t \mid \boldsymbol{z}) = \frac{\pi(t)\, p(\boldsymbol{z} \mid t)}{\sum_{s \in \mathcal{T}} \pi(s)\, p(\boldsymbol{z} \mid s)}. \tag{47}$$

For instructions that lie in the support of the pretraining distribution, we use different but compatible factorizations for the repeated and varying formats. In formats 1 and 3, the instruction is sampled once and, in format 3, its displayed occurrences are deterministic copies of the same shared realization. Hence

$$p(\boldsymbol{z} \mid t) = p(\boldsymbol{i} \mid t)\, p(x_{\text{query}} \mid t) \prod_{n=1}^N p(x_n, y_n \mid t). \tag{48}$$

In format 5, the paraphrases $\boldsymbol{i}_1, \ldots, \boldsymbol{i}_N, i_{\text{query}}$ are independently sampled from the task-conditional supported instruction law. Conditional on the latent task, they are also independent of the demonstration pairs and query input, so

$$p(\boldsymbol{z} \mid t) = p(i_{\text{query}} \mid t)\, p(x_{\text{query}} \mid t) \prod_{n=1}^N p(\boldsymbol{i}_n \mid t) p(x_n, y_n \mid t). \tag{49}$$

To treat all three formats uniformly, define

$$W_n := \begin{cases} (x_n, y_n), & \text{formats 1 and 3,} \\ (\boldsymbol{i}_n, x_n, y_n), & \text{format 5,} \end{cases} \qquad \boldsymbol{i}_\circ := \begin{cases} \boldsymbol{i}, & \text{formats 1 and 3,} \\ i_{\text{query}}, & \text{format 5.} \end{cases} \tag{50}$$

For format 5 we therefore have $p(W_n \mid t) = p(\boldsymbol{i}_n \mid t) p(x_n, y_n \mid t)$. In all three cases,

$$p(\boldsymbol{z} \mid t) = p(\boldsymbol{i}_\circ \mid t)\, p(x_{\text{query}} \mid t) \prod_{n=1}^N p(W_n \mid t). \tag{51}$$

For notational simplicity, below we write $\boldsymbol{i}$ for the single non-demonstration instruction variable $\boldsymbol{i}_\circ$.

Substituting (51) into (47), we obtain

$$p(t \mid \boldsymbol{z}) = \frac{\pi(t) p(\boldsymbol{i} \mid t) p(x_{\text{query}} \mid t) \prod_{n=1}^N p(W_n \mid t)}{\sum_{s \in \mathcal{T}} \pi(s) p(\boldsymbol{i} \mid s) p(x_{\text{query}} \mid s) \prod_{n=1}^N p(W_n \mid s)}. \tag{52}$$

To make the dependence on the number of demonstrations explicit, we compare each task t against the true task t^*. For any $t \neq t^*$,

$$\frac{p(t \mid \boldsymbol{z})}{p(t^* \mid \boldsymbol{z})} = \frac{\pi(t)}{\pi(t^*)} \frac{p(\boldsymbol{i} \mid t)}{p(\boldsymbol{i} \mid t^*)} \frac{p(x_{\text{query}} \mid t)}{p(x_{\text{query}} \mid t^*)} \frac{\prod_{n=1}^{N} p(W_n \mid t)}{\prod_{n=1}^{N} p(W_n \mid t^*)}. \tag{53}$$

We define the task-evidence likelihood ratio

$$B_t := \frac{\prod_{n=1}^{N} p(W_n \mid t)}{\prod_{n=1}^{N} p(W_n \mid t^*)}, \tag{54}$$

and the non-demonstration ratio

$$A_t(\boldsymbol{i}) := \frac{\pi(t)}{\pi(t^*)} \frac{p(\boldsymbol{i} \mid t)}{p(\boldsymbol{i} \mid t^*)} \frac{p(x_{\text{query}} \mid t)}{p(x_{\text{query}} \mid t^*)}. \tag{55}$$

Thus

$$\frac{p(t \mid \boldsymbol{z})}{p(t^* \mid \boldsymbol{z})} = A_t(\boldsymbol{i}) B_t, \qquad t \neq t^*, \tag{56}$$

and

$$p(t^* \mid \boldsymbol{z}) = \frac{1}{1 + \sum_{s \neq t^*} A_s(\boldsymbol{i}) B_s}, \qquad p(t \mid \boldsymbol{z}) = \frac{A_t(\boldsymbol{i}) B_t}{1 + \sum_{s \neq t^*} A_s(\boldsymbol{i}) B_s}, \quad t \neq t^*. \tag{57}$$

Then we present two lemmas.

Lemma D.10 (Exponential Decay of Task-Evidence Likelihood Ratio)**.** *Defining*

$$\alpha := \tfrac{1}{2} \min_{t \neq t^*} \kappa_t > 0,$$

there exists $N_0 < \infty$ such that almost surely, for all $N \geq N_0$ and all $t \neq t^*$,

$$B_t \leq e^{-\alpha N}.$$

Proof. Fix any $t \neq t^*$. Assume the corresponding log-likelihood ratio is integrable. By the strong law of large numbers and Assumption 3.1,

$$\frac{1}{N} \sum_{n=1}^{N} \log \frac{p(W_n \mid t)}{p(W_n \mid t^*)} \xrightarrow{\text{a.s.}} -\mathrm{KL}\Big(P_{W|t^*} \,\|\, P_{W|t}\Big) =: -\kappa_t, \tag{58}$$

where $\kappa_t > 0$. For formats 1 and 3 this is the demonstration KL divergence; for format 5 it is the joint KL divergence of the independently sampled instruction–demonstration triples $W_n = (\boldsymbol{i}_n, x_n, y_n)$. Hence, almost surely, there exists $N_t < \infty$ such that for all $N \geq N_t$,

$$B_t \leq \exp\big(-(\kappa_t/2)\, N\big). \tag{59}$$

Since $\mathcal{T}$ is finite, letting

$$\alpha := \frac{1}{2} \min_{t \neq t^*} \kappa_t \;>\; 0, \tag{60}$$

we have (after enlarging $N_0 := \max_{t \neq t^*} N_t$) that almost surely, for all $N \geq N_0$ and all $t \neq t^*$,

$$B_t \leq e^{-\alpha N}. \tag{61}$$

□

Lemma D.11 (Uniform Boundedness of Non-Demonstration Ratio)**.** *There exists a finite constant $A_{\max}$ such that*

$$\sup_{\boldsymbol{i} \in D} \max_{t \neq t^*} A_t(\boldsymbol{i}) \leq A_{\max} < \infty.$$

Proof. Recall that

$$A_t(\boldsymbol{i}) = \frac{\pi(t)}{\pi(t^*)} \frac{p(\boldsymbol{i} \mid t)}{p(\boldsymbol{i} \mid t^*)} \frac{p(x_{\text{query}} \mid t)}{p(x_{\text{query}} \mid t^*)}.$$

The vocabulary is finite and instruction length is bounded by L_i, so the supported instruction set $D \subseteq \mathcal{V}^{\leq L_i}$ is finite. Moreover, $p(\boldsymbol{i} \mid t^*) > 0$ for every $\boldsymbol{i} \in D$ by the definition of support. Therefore each ratio $p(\boldsymbol{i} \mid t)/p(\boldsymbol{i} \mid t^*)$ is finite, and its maximum over D is finite. Since $\mathcal{T}$ is finite and x_{query} is fixed,

$$A_{\max} := \max_{t \neq t^*} \frac{\pi(t)}{\pi(t^*)} \frac{p(x_{\text{query}} \mid t)}{p(x_{\text{query}} \mid t^*)} \max_{\boldsymbol{i} \in D} \frac{p(\boldsymbol{i} \mid t)}{p(\boldsymbol{i} \mid t^*)} < \infty.$$

This proves the claim without a differentiability or gradient assumption. □

D.4 Proof of Theorem 3.2

We first recall the statement of the theorem for convenience.

Theorem D.12 (Formal model bound with supported instructions)**.** *Consider prompt formats whose instructions lie in the support of the pretraining distribution, including formats 1, 3, and 5. Under the generation mechanism in Section 2 and Assumption 3.1, there exist constants $\alpha, \beta > 0$ such that, almost surely, for all sufficiently large numbers of demonstrations N,*

$$\left|p(y_{query} \mid \boldsymbol{z}) - p(y_{query} \mid x_{query}, t^*)\right| \leq \beta \exp(-\alpha N). \tag{62}$$

Moreover, with r_0 defined in (31) and $\varepsilon_M := r_0$,

$$\left|M(y_{query} \mid \boldsymbol{z}) - p(y_{query} \mid x_{query}, t^*)\right| \leq \beta e^{-\alpha N} + \varepsilon_M. \tag{63}$$

Consequently, for any two supported instruction realizations $\boldsymbol{i}, \boldsymbol{i}'$ to which the same format-specific assumptions apply (a single shared realization in formats 1 and 3, or two independently sampled paraphrase sequences in format 5), we have

$$\left|p(y_{query} \mid \boldsymbol{z}(\boldsymbol{i})) - p(y_{query} \mid \boldsymbol{z}(\boldsymbol{i}'))\right| \leq 2\beta \exp(-\alpha N). \tag{64}$$

We now proceed with the proof.

Proof. By definition of the posterior predictive,

$$p(y_{\text{query}} \mid \boldsymbol{z}) \;=\; \sum_{t \in \mathcal{T}} p(y_{\text{query}} \mid x_{\text{query}}, t)\, p(t \mid \boldsymbol{z}). \tag{65}$$

Subtracting $p(y_{\text{query}} \mid x_{\text{query}}, t^*)$ and using $\sum_t p(t \mid \boldsymbol{z}) = 1$ yields

$$p(y_{\text{query}} \mid \boldsymbol{z}) - p(y_{\text{query}} \mid x_{\text{query}}, t^*) = \sum_{t \neq t^*} \big(p(y_{\text{query}} \mid x_{\text{query}}, t) - p(y_{\text{query}} \mid x_{\text{query}}, t^*)\big)\, p(t \mid \boldsymbol{z}). \tag{66}$$

Since $0 \leq p(y_{\text{query}} \mid x_{\text{query}}, t) \leq 1$, we have $|p(y_{\text{query}} \mid x_{\text{query}}, t) - p(y_{\text{query}} \mid x_{\text{query}}, t^*)| \leq 1$ for all t, hence

$$\left|p(y_{\text{query}} \mid \boldsymbol{z}) - p(y_{\text{query}} \mid x_{\text{query}}, t^*)\right| \;\leq\; \sum_{t \neq t^*} p(t \mid \boldsymbol{z}). \tag{67}$$

It remains to upper bound the posterior mass on wrong tasks. From Equation 57, for any $t \neq t^*$,

$$p(t \mid \boldsymbol{z}) \;=\; \frac{A_t(\boldsymbol{i}) B_t}{1 + \sum_{s \neq t^*} A_s(\boldsymbol{i}) B_s} \;\leq\; A_t(\boldsymbol{i}) B_t. \tag{68}$$

Summing over $t \neq t^*$ and applying Lemma D.10 and D.11, we obtain that almost surely for all $N \geq N_0$,

$$\sum_{t \neq t^*} p(t \mid \boldsymbol{z}) \;\leq\; \sum_{t \neq t^*} A_t(\boldsymbol{i}) B_t \;\leq\; (|\mathcal{T}| - 1)\, A_{\max}\, e^{-\alpha N}. \tag{69}$$

Combining Equation 67, 69 proves the first statement with $\beta := (|\mathcal{T}| - 1) A_{\max}$:

$$\left|p(y_{\text{query}} \mid \boldsymbol{z}) - p(y_{\text{query}} \mid x_{\text{query}}, t^*)\right| \leq \beta e^{-\alpha N}.$$

For the equivalent statement comparing two supported instruction realizations $\boldsymbol{i}, \boldsymbol{i}'$ covered by the statement, we use the triangle inequality:

$$\left|p(y_{\text{query}} \mid \boldsymbol{z}(\boldsymbol{i})) - p(y_{\text{query}} \mid \boldsymbol{z}(\boldsymbol{i}'))\right| \tag{70}$$

$$\leq \left|p(y_{\text{query}} \mid \boldsymbol{z}(\boldsymbol{i})) - p(y_{\text{query}} \mid x_{\text{query}}, t^*)\right| + \left|p(y_{\text{query}} \mid \boldsymbol{z}(\boldsymbol{i}')) - p(y_{\text{query}} \mid x_{\text{query}}, t^*)\right| \tag{71}$$

$$\leq 2\beta e^{-\alpha N}, \tag{72}$$

Finally, a supported prompt is the constant-path case of Lemma D.1, so $|M(y_{\text{query}} \mid \boldsymbol{z}) - p(y_{\text{query}} \mid \boldsymbol{z})| \leq r_0$. Combining this inequality with the first bound by the triangle inequality proves (63) and completes the proof. □

D.5 Extension to an Unsupported Shared Instruction

We consider prompt format 4, in which the same instruction is prepended to all demonstrations and to the query. Let

$$\boldsymbol{z}_N(\boldsymbol{i}) := \big((\boldsymbol{i}, x_1, y_1), \ldots, (\boldsymbol{i}, x_N, y_N), (\boldsymbol{i}, x_{\text{query}})\big). \tag{73}$$

D.5.1 The shared-instruction likelihood

Although the instruction appears $N+1$ times in the surface form of (73), these occurrences are copies of a single template chosen once for the whole prompt, and not $N+1$ independent draws. We therefore model the instruction through a latent template variable.

Assumption D.13 (Shared-instruction factorization)**.** There is a latent template τ, drawn once per prompt from a task-conditional law $P_{\tau|t}$. Conditional on $\tau = \boldsymbol{i}$, every displayed instruction slot is a deterministic copy of the same realization $\boldsymbol{i}$, while the demonstrations and query are conditionally independent of τ given the task. Thus, for every $t \in \mathcal{T}$

$$p(\boldsymbol{z}_N(\boldsymbol{i}) \mid t) = p(\boldsymbol{i} \mid t) \prod_{n=1}^{N} p(x_n, y_n \mid t)\, p(x_{\text{query}} \mid t). \tag{74}$$

The instruction likelihood enters (74) *once*. This is the substantive modelling choice of this subsection: seeing one template repeated $N+1$ times is one observation about the task, not $N+1$ of them. Consequently, for every $t \neq t^*$,

$$\frac{p(t \mid \boldsymbol{z}_N(\boldsymbol{i}))}{p(t^* \mid \boldsymbol{z}_N(\boldsymbol{i}))} = B_t \exp\big(r_t(\boldsymbol{i}) + \lambda_t\big), \tag{75}$$

where

$$r_t(\boldsymbol{i}) := \log \frac{p(\boldsymbol{i} \mid t)}{p(\boldsymbol{i} \mid t^*)}, \qquad \lambda_t := \log\left(\frac{\pi(t)}{\pi(t^*)} \frac{p(x_{\text{query}} \mid t)}{p(x_{\text{query}} \mid t^*)}\right), \tag{76}$$

and B_t is the demonstration likelihood ratio of (54).

D.5.2 Minimal endpoint condition and the role of analyticity

The exponential mechanism does not itself require analytic continuation. Let a chosen off-support reference extension assign each competitor a finite endpoint log-odds $c_t := \bar{r}_t(\widetilde{\boldsymbol{i}}) \in \mathbb{R}$, independent of N, and define its endpoint odds by

$$\bar{R}_{t,N} := B_t \exp(c_t + \lambda_t). \tag{77}$$

Then Lemma D.10 immediately gives, almost surely for all sufficiently large N,

$$\sum_{t \neq t^*} \bar{R}_{t,N} \leq (|\mathcal{T}| - 1) \max_{t \neq t^*} e^{c_t + \lambda_t} e^{-\alpha N}. \tag{78}$$

Since the predictive deviation is at most the total competing posterior odds, the reference predictor converges at the same exponential rate. Thus the minimal substantive statement for a shared instruction is robustness to an arbitrary fixed finite off-support odds perturbation: demonstration log-evidence grows linearly in N, whereas the instruction contributes only an N-independent constant.

The fixed-endpoint argument above establishes exponential convergence conditional on arbitrary finite constants c_t. The analytic-continuation theorem below does not improve the exponent α, which is determined by the demonstration likelihood ratios. Its additional content is to replace the unstructured endpoint constants by a Transformer-induced continuation from a supported anchor and to derive the explicit path-dependent bound $c_t = \phi_t(1) \leq \gamma_0 + \Omega$. Consequently, the unknown factor $\max_t e^{c_t}$ in (78) is controlled by $e^{\gamma_0 + \Omega}$. The Cauchy–Taylor chain further yields local propagation bounds, a comparison bound between two unsupported endpoints, and additive continuation costs used in the mixed and varying-instruction analyses. Thus, the proof burden of the analytic theorem lies in constructing and quantitatively controlling the off-support perturbation, rather than in establishing a different asymptotic domination mechanism. Analyticity does not make the endpoint semantics statistically identified by zero-probability pretraining data.

Let $\boldsymbol{i}_0$ be a supported anchor and $\widetilde{\boldsymbol{i}}$ an unsupported instruction. Write

$$E_s := (1-s)E(\boldsymbol{i}_0) + sE(\widetilde{\boldsymbol{i}}), \quad s \in [0,1], \qquad \Delta := \|E(\widetilde{\boldsymbol{i}}) - E(\boldsymbol{i}_0)\|_F, \tag{79}$$

in the fixed-length instruction-embedding space of Section 2, so $s = 0$ is supported and $s = 1$ is the unsupported endpoint. Since the Bayes conditional is not determined at a zero-probability prompt, the object studied below is the posterior predictive induced by a fixed chain of local Taylor expansions of the supported log-odds along (79).

Assumption D.14 (Supported anchor)**.** The anchor is a supported rendering of the target task with $p(\boldsymbol{i}_0 \mid t) > 0$ for every $t \in \mathcal{T}$. Among admissible supported renderings one may choose the most typical anchor, for example $\boldsymbol{i}_0 \in \arg\max_{\boldsymbol{i} \in D} p(\boldsymbol{i} \mid t^*)$ (such as "+" among supported addition renderings). The proof below uses only support and finiteness. We write

$$\gamma_0 := \max_{t \neq t^*} r_t(\boldsymbol{i}_0) < \infty, \tag{80}$$

which satisfies $\gamma_0 \leq 0$ whenever the selected anchor is task-favoring, i.e. $p(\boldsymbol{i}_0 \mid t) \leq p(\boldsymbol{i}_0 \mid t^*)$ for every $t \neq t^*$.

Under the shared reading γ_0 enters only additively in a constant, so no sign condition on it is required.

D.5.3 The chain hypothesis, and the constants it induces

The hypothesis below specifies a finite chain of overlapping complex disks along which the log-odds continue holomorphically. The constants used in the proof are defined from this chain in Lemma D.16.

For clarity, the new symbol $\phi_{t,\ell}$ denotes the local analytic continuation of the supported pathwise log-odds $s \mapsto r_t(E_s)$ on the ℓ-th disk of the chain. The index t specifies the competing task and ℓ specifies the local neighborhood. Agreement on overlapping disks makes these local functions pieces of one continued function along the whole path; we denote that resulting function by ϕ_t.

Assumption D.15 (Finite chain of analytic continuations)**.** There exist an integer $L < \infty$ and centers

$$0 = s_0 < s_1 < \cdots < s_L = 1 \tag{81}$$

with associated open complex disks $D_\ell := \{\zeta \in \mathbb{C} : |\zeta - s_\ell| < r_\ell\}$, $\ell = 0, \ldots, L-1$, all independent of N, such that

$$d_\ell := s_{\ell+1} - s_\ell < r_\ell, \qquad \ell = 0, \ldots, L-1, \tag{82}$$

and such that for every $t \neq t^*$ there are holomorphic functions $\phi_{t,\ell} : D_\ell \to \mathbb{C}$ with

(i) (compatibility) $\phi_{t,\ell} = \phi_{t,m}$ on $D_\ell \cap D_m$ whenever this intersection is nonempty;

(ii) (anchoring) $\phi_{t,0}(s) = r_t(E_s)$ on a nontrivial instruction-embedding interval adjacent to $s = 0$ on which the fixed extension is defined.

We write ϕ_t for the function these germs define on $[0,1]$, and $\phi_t(1)$ for the value at $s_L = 1$, which lies in D_{L-1} by (82).

Only D_0 concerns supported instructions; the remaining $L-1$ disks are the part of the hypothesis that describes continuation outside the support.

Lemma D.16 (Constants induced by the chain)**.** *Fix working radii $\rho_\ell \in (d_\ell, r_\ell)$ and set*

$$M_\ell := \max_{t \neq t^*} \max_{|\zeta - s_\ell| = \rho_\ell} \big|\phi_t(\zeta) - \phi_t(s_\ell)\big|, \qquad q_\ell := \frac{d_\ell}{\rho_\ell} \in (0,1). \tag{83}$$

Then, under Assumption D.15:

(a) every M_ℓ is finite, and $M_\ell, \rho_\ell, q_\ell$ are independent of N;

(b) for every $t \neq t^*$, every ℓ and every $k \geq 1$,

$$\big|\phi_t^{(k)}(s_\ell)\big| \;\leq\; \frac{M_\ell\, k!}{\rho_\ell^{\,k}}; \tag{84}$$

(c) every ϕ_t has real Taylor coefficients at every s_ℓ, and $\phi_t(s) \in \mathbb{R}$ for all $s \in [0,1]$.

Proof. (a) Since $\rho_\ell < r_\ell$, the circle $|\zeta - s_\ell| = \rho_\ell$ is a compact subset of D_ℓ, on which the holomorphic—hence continuous—function ϕ_t is bounded. The maximum over the finite set $\mathcal{T}$ of finitely many finite numbers is finite. None of D_ℓ, ρ_ℓ, ϕ_t depends on N, by Assumption D.15 and because r_t in (76) does not.

(b) Apply the Cauchy estimates on the circle of radius ρ_ℓ to the holomorphic function $\psi_t := \phi_t - \phi_t(s_\ell)$, which satisfies $\max_{|\zeta - s_\ell| = \rho_\ell} |\psi_t| = M_\ell$ by (83) and $\psi_t^{(k)} = \phi_t^{(k)}$ for $k \geq 1$.

(c) By (ii) the germ $\phi_{t,0}$ agrees with the real-valued function $s \mapsto r_t(E_s)$ on a nontrivial real interval, so all its derivatives at s_0 are real. If ϕ_t has real coefficients at s_ℓ, then by (i) its coefficients at the real point $s_{\ell+1}$ are obtained by term-by-term differentiation of a real power series, hence are real. Induction on ℓ gives the claim, and a power series with real coefficients is real on the real axis inside its disk of convergence. □

Inequality (84) is the quantitative form of analyticity at s_ℓ: by Cauchy–Hadamard it says the Taylor series there has radius of convergence at least ρ_ℓ with coefficients bounded by $M_\ell \rho_\ell^{-k}$. It is a consequence of Assumption D.15, not an additional hypothesis.

D.5.4 One Taylor step, and the cancellation of $k!$

The next lemma is the sole mechanism of the proof. It converts the local coefficient bound (84) at s_ℓ into a bound on the *increment* of ϕ_t across the link. The factorial $k!$ in (84) is exactly cancelled by the factorial in the Taylor coefficients, leaving a geometric series of ratio $q_\ell < 1$.

Lemma D.17 (Taylor transfer across one link)**.** *Under Assumption D.15, for every $t \neq t^*$ and every $\ell = 0, \ldots, L-1$:*

(a) the increment across the link satisfies

$$\eta_\ell := \max_{t \neq t^*} \big|\phi_t(s_{\ell+1}) - \phi_t(s_\ell)\big| \;\leq\; \frac{M_\ell \, q_\ell}{1 - q_\ell}; \tag{85}$$

(b) more generally, for $0 \leq u < 1$ and $|s - s_\ell| \leq u\rho_\ell$,

$$\big|\phi_t(s) - \phi_t(s_\ell)\big| \leq \frac{M_\ell \, u}{1 - u}; \tag{86}$$

(c) the germ at $s_{\ell+1}$ inherited from s_ℓ obeys

$$\big|\phi_t^{(k)}(s_{\ell+1})\big| \;\leq\; \frac{M_\ell}{1 - q_\ell} \cdot \frac{k!}{\big(\rho_\ell - d_\ell\big)^k}, \qquad k \geq 1. \tag{87}$$

Proof. (b) Since the Taylor series of ϕ_t at s_ℓ converges for $|s - s_\ell| < \rho_\ell$ by (84),

$$\big|\phi_t(s) - \phi_t(s_\ell)\big| \leq \sum_{k \geq 1} \frac{\big|\phi_t^{(k)}(s_\ell)\big|}{k!} \, |s - s_\ell|^k \leq \sum_{k \geq 1} \frac{1}{k!} \cdot \frac{M_\ell k!}{\rho_\ell^k} \, (u\rho_\ell)^k = M_\ell \sum_{k \geq 1} u^k = \frac{M_\ell u}{1 - u}. \tag{88}$$

The middle step is the cancellation of $k!$ against $1/k!$: without it the series would be $\sum_k k! \, u^k$, which diverges for every $u > 0$.

(a) Apply (b) with $u = q_\ell$, legitimate because $d_\ell = q_\ell \rho_\ell$ and $q_\ell < 1$, and take the maximum over the finite set $\mathcal{T}$.

(c) Differentiating the Taylor series of ϕ_t at s_ℓ term by term k times and evaluating at $s_{\ell+1}$,

$$\phi_t^{(k)}(s_{\ell+1}) = \sum_{j \geq k} \frac{\phi_t^{(j)}(s_\ell)}{(j-k)!} \, d_\ell^{\,j-k},$$

so by (84) and $\frac{(m+k)!}{m!} = k!\binom{m+k}{k}$,

$$\left|\phi_t^{(k)}(s_{\ell+1})\right| \le \frac{M_\ell}{\rho_\ell^k} \sum_{m\ge 0} \frac{(m+k)!}{m!} q_\ell^m = \frac{M_\ell\, k!}{\rho_\ell^k}\,(1-q_\ell)^{-(k+1)},$$

using $\sum_{m\ge 0}\binom{m+k}{k} q^m = (1-q)^{-(k+1)}$. Rearranging with $\rho_\ell(1-q_\ell) = \rho_\ell - d_\ell$ gives (87). □

Remark D.18 (The chain hypothesis is not vacuous, and not redundant). Part (c) shows that a germ at s_ℓ automatically induces a germ at $s_{\ell+1}$ with degraded constants $\big(M_\ell/(1-q_\ell),\ \rho_\ell - d_\ell\big)$: the radius shrinks by exactly the step length, which is the classical behaviour of Weierstrass continuation. Iterating from s_0 alone would therefore reach only $s < \rho_0$, recovering the single-germ case. Assumption D.15 is binding precisely when some center supplies constants better than those propagated by (c), i.e. when the log-odds regain analytic room further along the path. This is what allows $L > 1$ to travel a distance exceeding ρ_0.

Lemma D.19 (Additive propagation along the chain)**.** *Define the* cumulative continuation cost

$$\Omega := \sum_{\ell=0}^{L-1} \frac{M_\ell\, q_\ell}{1-q_\ell}. \tag{89}$$

Under Assumption D.15, $\Omega < \infty$ automatically, since the chain is finite and every M_ℓ is finite and every $q_\ell < 1$. Moreover, under Assumption D.14, for every $t \ne t^*$ and every $\ell = 0, \dots, L$,

$$\phi_t(s_\ell) \;\le\; \gamma_0 + \sum_{j=0}^{\ell-1} \eta_j \;\le\; \gamma_0 + \Omega. \qquad \textit{In particular} \qquad \phi_t(1) \le \gamma_0 + \Omega =: \Omega_0. \tag{90}$$

Proof. By Lemma D.16(c) all quantities are real. Telescoping,

$$\phi_t(s_\ell) = \phi_t(s_0) + \sum_{j=0}^{\ell-1}\big(\phi_t(s_{j+1}) - \phi_t(s_j)\big) \le r_t(\boldsymbol{i}_0) + \sum_{j=0}^{\ell-1}\eta_j \le \gamma_0 + \sum_{j=0}^{\ell-1}\eta_j,$$

and $\eta_j \le M_j q_j/(1-q_j)$ by Lemma D.17(a). Since every term is nonnegative, the partial sums are bounded by Ω. □

The propagation is *additive* because the chain acts on the log-odds. Had the same chain been run on the odds themselves, or on the posterior predictive, each link would have contributed a multiplicative factor and L links would produce an L-th power; this is the structural reason the present argument succeeds where amplification-based transfers do not.

D.5.5 Main result

Define the continued odds, odds mass and predictive deviation by

$$R_{t,N}(s) := B_t \exp\big(\phi_t(s) + \lambda_t\big), \qquad \sigma_N(s) := \sum_{t\ne t^*} R_{t,N}(s), \tag{91}$$

$$h_N(s) := \sum_{t\ne t^*} b_t\, \frac{R_{t,N}(s)}{1+\sigma_N(s)}, \qquad b_t := p(y_{\text{query}} \mid x_{\text{query}}, t) - p(y_{\text{query}} \mid x_{\text{query}}, t^*), \quad |b_t| \le 1. \tag{92}$$

At every s in the anchored initial instruction-embedding interval, $h_N(s) = p(y_{\text{query}} \mid \boldsymbol{z}_N(E_s)) - p(y_{\text{query}} \mid x_{\text{query}}, t^*)$ by (75); at $s = 1$ it is the value induced by the chain.

Theorem D.20 (Formal model bound for an unsupported shared instruction)**.** *Let Assumptions 3.1, D.13, D.14 and D.15 hold. Then almost surely, for all sufficiently large N,*

$$\left|h_N(1)\right| \;\le\; \widetilde{\beta}\, e^{-\alpha N}, \qquad \widetilde{\beta} := \beta_\circ e^{\Omega_0}, \quad \beta_\circ := (|\mathcal{T}|-1)\max_{t\ne t^*} e^{\lambda_t}, \tag{93}$$

with $\alpha = \frac{1}{2}\min_{t\ne t^*}\kappa_t$ as in Lemma D.10. Equivalently, the posterior predictive induced by the chain satisfies

$$\left|\bar{p}(y_{query} \mid \boldsymbol{z}_N(\widetilde{\boldsymbol{i}})) - p(y_{query} \mid x_{query}, t^*)\right| \le \widetilde{\beta} e^{-\alpha N}, \tag{94}$$

and, for the actual pretrained model,

$$\big|M(y_{query} \mid \boldsymbol{z}_N(\widetilde{\boldsymbol{i}})) - p(y_{query} \mid x_{query}, t^*)\big| \le \widetilde{\beta} e^{-\alpha N} + \varepsilon_{M,N}(\widetilde{\boldsymbol{i}}), \tag{95}$$

where $\varepsilon_{M,N}(\widetilde{\boldsymbol{i}})$ is defined by (41), and for any two unsupported instructions admitting chains with costs Ω_0, Ω_0',

$$\big|\bar{p}(y_{query} \mid \boldsymbol{z}_N(\widetilde{\boldsymbol{i}})) - \bar{p}(y_{query} \mid \boldsymbol{z}_N(\widetilde{\boldsymbol{i}}\,'))\big| \le 2\beta_\circ e^{\max\{\Omega_0,\Omega_0'\}} e^{-\alpha N}. \tag{96}$$

Proof. By Lemma D.10, almost surely for all sufficiently large N we have $B_t \le e^{-\alpha N}$ for every $t \ne t^*$. By Lemma D.19, $\phi_t(1) \le \Omega_0$, and by Lemma D.16(c) this is a real inequality. Hence

$$R_{t,N}(1) = B_t \exp\big(\phi_t(1) + \lambda_t\big) \le e^{\lambda_t} e^{\Omega_0} e^{-\alpha N}. \tag{97}$$

Summing over the $|\mathcal{T}| - 1$ competitors, $\sigma_N(1) \le \beta_\circ e^{\Omega_0} e^{-\alpha N}$. Since every $R_{t,N}(1)$ is real and nonnegative we have $1 + \sigma_N(1) \ge 1$, so by (92) and $|b_t| \le 1$,

$$\big|h_N(1)\big| \le \frac{\sigma_N(1)}{1 + \sigma_N(1)} \le \sigma_N(1) \le \beta_\circ e^{\Omega_0} e^{-\alpha N},$$

which is (93). Statement (96) follows by the triangle inequality as in Theorem D.12. Combining (94) with Corollary D.2 gives (95). □

Assumption D.21 (Uniform model–reference tracking along a shared-template path)**.** For the fixed unsupported instruction $\widetilde{\boldsymbol{i}}$, there exist N_0 and a finite constant $\bar{\varepsilon}_M(\widetilde{\boldsymbol{i}}) \ge 0$ such that, almost surely,

$$\sup_{N \ge N_0} \varepsilon_{M,N}(\widetilde{\boldsymbol{i}}) \le \bar{\varepsilon}_M(\widetilde{\boldsymbol{i}}). \tag{98}$$

Corollary D.22 (Uniform shared-template model tracking)**.** *If, in addition, Assumption D.21 holds, then almost surely for all sufficiently large N,*

$$\big|M(y_{query} \mid \boldsymbol{z}_N(\widetilde{\boldsymbol{i}})) - p(y_{query} \mid x_{query}, t^*)\big| \le \widetilde{\beta} e^{-\alpha N} + \bar{\varepsilon}_M(\widetilde{\boldsymbol{i}}), \tag{99}$$

and consequently

$$\limsup_{N \to \infty} \big|M(y_{query} \mid \boldsymbol{z}_N(\widetilde{\boldsymbol{i}})) - p(y_{query} \mid x_{query}, t^*)\big| \le \bar{\varepsilon}_M(\widetilde{\boldsymbol{i}}).$$

If, more strongly, $\varepsilon_{M,N}(\widetilde{\boldsymbol{i}}) \to 0$, then $M(y_{query} \mid \boldsymbol{z}_N(\widetilde{\boldsymbol{i}})) \to p(y_{query} \mid x_{query}, t^*)$ almost surely.

Proof. Substitute the uniform bound from Assumption D.21 into (95) and take the limit superior. The final claim follows from the same bound when both terms on its right-hand side vanish. □

Proposition D.7 supplies sufficient conditions for Assumption D.21, with the explicit choice $\bar{\varepsilon}_M = \bar{r} + (C_M + 2G)d$. Proposition D.8 further relates $\bar{r}$ to population excess log loss on a covered anchor family. Architectural analyticity alone supplies neither small calibration error nor a small transport constant. For a finite context window these results apply only to admissible N and do not imply an infinite-length limit.

Two features distinguish (93) from what the independent-occurrence reading would give. First, the decay rate is the undegraded α of Theorem D.12: passing to an unsupported instruction costs a constant factor, not a slower exponential. Second, no upper bound on Ω is required, so there is no fixed admissible distance; the cost of travelling further is paid once, in the constant.

Corollary D.23 (Certified distance under a uniform path-cost bound)**.** *Consider a collection of unsupported endpoints that share the supported anchor $\boldsymbol{i}_0$ and the fixed query contribution λ_t. For an endpoint $\widetilde{\boldsymbol{i}}$, let $\Delta(\widetilde{\boldsymbol{i}}) := \|E(\widetilde{\boldsymbol{i}}) - E(\boldsymbol{i}_0)\|_F$, and write $\Omega(\widetilde{\boldsymbol{i}})$ for the cumulative cost of its continuation chain. Suppose there is a constant $\mathcal{L}_{\mathrm{path}} > 0$, independent of N and of the endpoint, such that*

$$\Omega(\widetilde{\boldsymbol{i}}) \le \mathcal{L}_{\mathrm{path}} \Delta(\widetilde{\boldsymbol{i}}) \tag{100}$$

throughout this collection. Set $\beta_0 := \beta_\circ e^{\gamma_0}$. Then

$$|h_N(1)| \le \beta_0 \exp\Big(-\alpha N + \mathcal{L}_{\mathrm{path}} \Delta(\widetilde{\boldsymbol{i}})\Big). \tag{101}$$

Consequently, for any $\epsilon \in (0,1)$, $|h_N(1)| \le \epsilon$ whenever

$$\Delta(\widetilde{\boldsymbol{i}}) \le \frac{\alpha N - \log(\beta_0/\epsilon)}{\mathcal{L}_{\mathrm{path}}}. \tag{102}$$

Thus, within an endpoint family satisfying the same uniform path-cost bound, the certified distance grows linearly with N.

Proof. The proof of Theorem D.20 gives $|h_N(1)| \le \beta_\circ e^{\gamma_0 + \Omega(\widetilde{\boldsymbol{i}})} e^{-\alpha N}$. Applying (100) proves (101). Solving $\beta_0 e^{-\alpha N + \mathcal{L}_{\mathrm{path}}\Delta(\widetilde{\boldsymbol{i}})} \le \epsilon$ for $\Delta(\widetilde{\boldsymbol{i}})$ gives (102). □

D.6 Mixed Supported and Unsupported Instructions

We next consider a prompt in which most demonstrations display the same supported instruction while a few display corrupted, unsupported surface forms, for example misspellings. Write

$$\boldsymbol{z}_N := \big((\boldsymbol{i}_1, x_1, y_1), \ldots, (\boldsymbol{i}_N, x_N, y_N), (\boldsymbol{i}_q, x_{\text{query}})\big), \tag{103}$$

let $S_N \subseteq [N]$ denote the supported instruction indices and $U_N := [N] \setminus S_N$ the unsupported ones, and set

$$m_N := |S_N|, \qquad r_N := |U_N| = N - m_N. \tag{104}$$

All supported slots display the same instruction, which we denote $\boldsymbol{i}_S$; the unsupported slots display arbitrary distinct values $\{\boldsymbol{i}_n\}_{n \in U_N}$.

Assumption D.24 (Shared supported template, non-shared corruptions)**.** The m_N supported occurrences are renderings of one template chosen once for the whole prompt and therefore contribute a single supported factor $p(\boldsymbol{i}_S \mid t)$, exactly as in Assumption D.13; when $m_N = 0$, this factor is omitted. Each unsupported occurrence is treated as a separate instruction slot. On supported values the corresponding slot likelihoods factor independently of the demonstration pairs given the task; at unsupported realized values their log-likelihood ratios are defined by the fixed continuation chains introduced below.

This is the natural mixture of the shared reading for repeated supported copies and the independent-observation reading for distinct corruptions. The latter do not have literal Bayes likelihood ratios at zero-probability values; the object below is therefore a continuation-induced posterior, denoted by $\bar{p}$.

For $t \neq t^*$ define

$$\ell_{t,n} := \log \frac{p(x_n, y_n \mid t)}{p(x_n, y_n \mid t^*)}, \qquad r_t(\boldsymbol{i}) := \log \frac{p(\boldsymbol{i} \mid t)}{p(\boldsymbol{i} \mid t^*)} \tag{105}$$

for supported $\boldsymbol{i}$. For every unsupported slot $n \in U_N$, let $\bar{r}_{t,n}(\boldsymbol{i}_n)$ be the endpoint value of its specified finite analytic continuation chain. For the query define

$$\bar{\lambda}_{t,q} := \log\left(\frac{\pi(t)}{\pi(t^*)} \frac{p(x_{\text{query}} \mid t)}{p(x_{\text{query}} \mid t^*)}\right) + \begin{cases} r_t(\boldsymbol{i}_q), & \boldsymbol{i}_q \text{ supported}, \\ \bar{r}_{t,q}(\boldsymbol{i}_q), & \boldsymbol{i}_q \text{ unsupported}, \end{cases} \tag{106}$$

where $\bar{r}_{t,q}$ is obtained from a fixed finite chain in the unsupported case. We then define the continuation-induced odds by

$$\log \bar{R}_{t,N} := \bar{\lambda}_{t,q} + \sum_{n=1}^{N} \ell_{t,n} + r_t(\boldsymbol{i}_S) + \sum_{n \in U_N} \bar{r}_{t,n}(\boldsymbol{i}_n), \tag{107}$$

and normalize them as

$$\bar{p}(t^* \mid \boldsymbol{z}_N) := \frac{1}{1 + \sum_{s \neq t^*} \bar{R}_{s,N}}, \qquad \bar{p}(t \mid \boldsymbol{z}_N) := \frac{\bar{R}_{t,N}}{1 + \sum_{s \neq t^*} \bar{R}_{s,N}}. \tag{108}$$

The induced posterior predictive is

$$\bar{p}(y_{\text{query}} \mid \boldsymbol{z}_N) := \sum_{t \in \mathcal{T}} p(y_{\text{query}} \mid x_{\text{query}}, t)\, \bar{p}(t \mid \boldsymbol{z}_N). \tag{109}$$

The shared supported contribution $r_t(\boldsymbol{i}_S)$ is a single N-independent term; each distinct unsupported occurrence contributes one continued log-ratio.

Assumption D.25 (Supported template and local continuation of the corruptions)**.** For every $t \neq t^*$ the shared supported template satisfies

$$r_t(\boldsymbol{i}_S) \leq 0, \tag{110}$$

and for every $n \in U_N$ there exist a supported anchor $\boldsymbol{i}_n^0$ with $r_t(\boldsymbol{i}_n^0) \leq 0$ and a finite chain of analytic continuations connecting $\boldsymbol{i}_n^0$ to $\boldsymbol{i}_n$, in the sense of Assumption D.15.

Condition (110) is the anchor condition (80) applied to the shared template, and is a single inequality rather than one per slot. The natural choice of anchor for a misspelling is the uncorrupted rendering itself, $\boldsymbol{i}_n^0 = \boldsymbol{i}_S$, in which case the continuation distance $\|\boldsymbol{i}_n - \boldsymbol{i}_S\|_2$ is the size of the typo.

Lemma D.26 (Continuation cost of a corrupted occurrence)**.** *Under Assumption D.25, let $\Omega_{t,n}$ denote the cumulative continuation cost (89) of the chain reaching $\boldsymbol{i}_n$ for the task pair (t, t^*), and set*

$$\Omega_n := \max_{t \neq t^*} \Omega_{t,n} < \infty. \tag{111}$$

Then, for every $n \in U_N$ and every $t \neq t^*$,

$$\bar{r}_{t,n}(\boldsymbol{i}_n) \;\leq\; \Omega_{t,n} \;\leq\; \Omega_n. \tag{112}$$

Proof. Apply Lemma D.19 to the chain connecting $\boldsymbol{i}_n^0$ to $\boldsymbol{i}_n$: it gives $\bar{r}_{t,n}(\boldsymbol{i}_n) \leq \gamma_0 + \Omega_{t,n}$ with $\gamma_0 = \max_{t \neq t^*} r_t(\boldsymbol{i}_n^0) \leq 0$ by Assumption D.25. Finiteness of $\Omega_{t,n}$ is Lemma D.16(a), and the maximum over the finite set $\mathcal{T}$ is finite. □

The following theorem separates the task information carried by the examples from the continuation burden created by the corrupted occurrences.

Theorem D.27 (Formal mixed supported/unsupported instruction bound)**.** *Let Assumptions D.24 and D.25 hold. Suppose that, almost surely for all sufficiently large N, there is a sequence $a_N \geq 0$ such that simultaneously for all $t \neq t^*$,*

$$\sum_{n=1}^{N} \ell_{t,n} \leq -a_N. \tag{113}$$

Then, almost surely for all sufficiently large N,

$$\frac{\bar{p}(t \mid \boldsymbol{z}_N)}{\bar{p}(t^* \mid \boldsymbol{z}_N)} \leq e^{\bar{\lambda}_{t,q}} \exp\left(-a_N + \sum_{n \in U_N} \Omega_n\right), \qquad t \neq t^*, \tag{114}$$

and consequently, with $\beta_q := (|\mathcal{T}| - 1) \max_{t \neq t^*} e^{\bar{\lambda}_{t,q}}$,

$$\left|\bar{p}(y_{\textit{query}} \mid \boldsymbol{z}_N) - p(y_{\textit{query}} \mid x_{\textit{query}}, t^*)\right| \leq \beta_q \exp\left(-a_N + \sum_{n \in U_N} \Omega_n\right). \tag{115}$$

For the actual pretrained model,

$$\left|M(y_{\textit{query}} \mid \boldsymbol{z}_N) - p(y_{\textit{query}} \mid x_{\textit{query}}, t^*)\right| \leq \beta_q \exp\left(-a_N + \sum_{n \in U_N} \Omega_n\right) + \varepsilon_M(\boldsymbol{z}_N), \tag{116}$$

where $\varepsilon_M(\boldsymbol{z}_N)$ is the pathwise transport error in (32). In particular, continued posterior predictive convergence follows whenever

$$a_N - \sum_{n \in U_N} \Omega_n \longrightarrow +\infty. \tag{117}$$

Proof. From (107), (113), (110) and Lemma D.26,

$$\log \frac{\bar{p}(t \mid \boldsymbol{z}_N)}{\bar{p}(t^* \mid \boldsymbol{z}_N)} \;\leq\; \bar{\lambda}_{t,q} - a_N + \underbrace{r_t(\boldsymbol{i}_S)}_{\leq 0} + \sum_{n \in U_N} \Omega_n \;\leq\; \bar{\lambda}_{t,q} - a_N + \sum_{n \in U_N} \Omega_n,$$

which is (114). Since $\bar{p}(t^* \mid \boldsymbol{z}_N) \leq 1$, we have $\bar{p}(t \mid \boldsymbol{z}_N) \leq \bar{p}(t \mid \boldsymbol{z}_N)/\bar{p}(t^* \mid \boldsymbol{z}_N)$; summing over the $|\mathcal{T}| - 1$ competing tasks bounds the continued posterior mass outside t^*, and (115) follows by repeating the calculation of (67) with $\bar{p}$ in place of p. The triangle inequality and Lemma D.1 then give (116). □

Remark D.28 (Consistency with Assumption D.13). Under Assumption D.24 the m_N supported occurrences contribute the single term $r_t(\boldsymbol{i}_S)$ of (107), so Lemma D.11 applies to them with an N-independent constant. Only the continued corrupted-slot contributions accumulate, and they do so at the rate $\sum_{n\in U_N}\Omega_n$ rather than at rate N. Setting $U_N=\varnothing$ recovers the shared-supported case of Theorem D.12, and setting $S_N=\varnothing$ with a single repeated corruption recovers Theorem D.20.

We now specialize Theorem D.27 to the setting used in the main text: the surface instruction may be unsupported, but its associated pair (x_n, y_n) is still generated by the true task and remains informative. In this case Lemma D.10 gives $a_N=\alpha N$ directly.

Corollary D.29 (Evidence-preserving typos)**.** *Under the hypotheses of Theorem D.27, suppose* $\sum_{n=1}^{N}\ell_{t,n}\le -\alpha N$ *almost surely eventually, uniformly over* $t\neq t^*$*. Then*

$$\left|\bar{p}(y_{\textit{query}}\mid \boldsymbol{z}_N)-p(y_{\textit{query}}\mid x_{\textit{query}},t^*)\right|\le \beta_q\exp\left(-\alpha N+\sum_{n\in U_N}\Omega_n\right), \tag{118}$$

so convergence holds whenever

$$\limsup_{N\to\infty}\frac{1}{N}\sum_{n\in U_N}\Omega_n<\alpha. \tag{119}$$

If the corruption costs are uniformly bounded, $\Omega_n\le\Omega$*, then*

$$\left|\bar{p}(y_{\textit{query}}\mid \boldsymbol{z}_N)-p(y_{\textit{query}}\mid x_{\textit{query}},t^*)\right|\le \beta_q\exp\Big(-N\Big[\alpha-\Big(1-\tfrac{m_N}{N}\Big)\Omega\Big]\Big). \tag{120}$$

Proof. Apply Theorem D.27 with $a_N=\alpha N$, and use $\sum_{n\in U_N}\Omega_n\le r_N\Omega=(N-m_N)\Omega$ for the last display. □

Remark D.30 (Asymptotic regimes). Under Corollary D.29:

1. if $r_N=O(1)$, a fixed number of typos changes only the prefactor, $\left|\bar{p}(y_{\text{query}}\mid \boldsymbol{z}_N)-p(y_{\text{query}}\mid x_{\text{query}},t^*)\right|\le\beta_q e^{O(1)}e^{-\alpha N}$;
2. if $r_N=o(N)$ and $\sup_n\Omega_n<\infty$, then $-\alpha N+\sum_{n\in U_N}\Omega_n=-\alpha N+o(N)$, so for every $\varepsilon>0$ the deviation is eventually $O(e^{-(\alpha-\varepsilon)N})$;
3. if $r_N/N\to\rho\in[0,1]$ and $r_N^{-1}\sum_{n\in U_N}\Omega_n\to\overline{\Omega}$, the effective exponent is $\alpha_{\text{eff}}=\alpha-\rho\overline{\Omega}$. Thus $\rho\overline{\Omega}<\alpha$ gives exponential convergence, equality gives no decay from the present bound, and $\rho\overline{\Omega}>\alpha$ is not covered.

D.7 Varying unsupported instructions

Format 6 is the special case of the preceding framework in which every per-demonstration instruction is a distinct unsupported realization. Thus $S_N=\varnothing$, $U_N=[N]$, and the prompt is

$$\boldsymbol{z}_N^{(6)}:=\big((\widetilde{\boldsymbol{i}}_1,x_1,y_1),\dots,(\widetilde{\boldsymbol{i}}_N,x_N,y_N),(\widetilde{\boldsymbol{i}}_q,x_{\text{query}})\big). \tag{121}$$

For each n, let $\bar{r}_{t,n}(\widetilde{\boldsymbol{i}}_n)$ be the endpoint value supplied by a fixed finite analytic-continuation chain from a supported anchor, and absorb the single query-instruction continuation into $\bar{\lambda}_{t,q}$. The continuation-induced odds therefore satisfy the exact identity

$$\log\frac{\bar{p}(t\mid \boldsymbol{z}_N^{(6)})}{\bar{p}(t^*\mid \boldsymbol{z}_N^{(6)})}=\bar{\lambda}_{t,q}+\sum_{n=1}^{N}\ell_{t,n}+\sum_{n=1}^{N}\bar{r}_{t,n}(\widetilde{\boldsymbol{i}}_n),\qquad t\neq t^*. \tag{122}$$

Unlike format 4, the continuation burden is paid once for every distinct instruction realization and can therefore grow linearly with N.

Corollary D.31 (Conditional bound for format 6)**.** *Suppose that the demonstration pairs remain informative, so that almost surely, for all sufficiently large* N *and all* $t\neq t^*$*,*

$$\sum_{n=1}^{N}\ell_{t,n}\le-\alpha N, \tag{123}$$

and suppose every unsupported instruction admits a finite continuation chain with cost Ω_n as in Lemma D.26. Then

$$\left|\bar{p}(y_{query} \mid \boldsymbol{z}_N^{(6)}) - p(y_{query} \mid x_{query}, t^*)\right| \le \beta_q \exp\left(-\alpha N + \sum_{n=1}^{N} \Omega_n\right). \tag{124}$$

Consequently, convergence is guaranteed if

$$\alpha N - \sum_{n=1}^{N} \Omega_n \longrightarrow +\infty, \tag{125}$$

or, more concretely, if

$$\limsup_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} \Omega_n < \alpha. \tag{126}$$

The finite-chain assumption alone only gives $\Omega_n < \infty$ separately for each n and does not imply either condition.

Proof. Set $S_N = \varnothing$ and $U_N = [N]$ in Theorem D.27, and take $a_N = \alpha N$ using (123). This gives (124). Conditions (125) and (126) make its right-hand side tend to zero. □

The preceding corollary gives only a sufficient condition. The next result shows that this limitation is intrinsic: under the assumptions used so far, the continued instruction evidence may cancel or dominate the demonstration evidence.

Theorem D.32 (No unconditional convergence guarantee for format 6)**.** *Assume the query contribution $\bar{\lambda}_{t,q}$ is fixed, the demonstrations are i.i.d. from $P_{X,Y|t^*}$, and the log-likelihood ratios are integrable. For every $t \neq t^*$ define the average continued instruction pressure*

$$c_{t,N} := \frac{1}{N} \sum_{n=1}^{N} \bar{r}_{t,n}(\widetilde{\boldsymbol{i}}_n). \tag{127}$$

Then almost surely

$$\frac{1}{N} \log \frac{\bar{p}(t \mid \boldsymbol{z}_N^{(6)})}{\bar{p}(t^* \mid \boldsymbol{z}_N^{(6)})} = c_{t,N} - \kappa_t + o(1), \qquad t \neq t^*, \tag{128}$$

where $\kappa_t = \mathrm{KL}(P_{X,Y|t^*} \| P_{X,Y|t}) > 0$. Consequently:

(i) if

$$\limsup_{N\to\infty} c_{t,N} < \kappa_t \qquad \text{for every } t \neq t^*, \tag{129}$$

then $\bar{p}(t^* \mid \boldsymbol{z}_N^{(6)}) \to 1$ exponentially fast and the continued posterior predictive converges to $p(y_{query} \mid x_{query}, t^*)$;

(ii) if $\mathcal{T} = \{t^*, t^\dagger\}$ and

$$\liminf_{N\to\infty} c_{t^\dagger,N} > \kappa_{t^\dagger}, \tag{130}$$

then

$$\bar{p}(t^* \mid \boldsymbol{z}_N^{(6)}) \longrightarrow 0, \qquad \bar{p}(t^\dagger \mid \boldsymbol{z}_N^{(6)}) \longrightarrow 1. \tag{131}$$

Hence, whenever $p(y_{query} \mid x_{query}, t^\dagger) \neq p(y_{query} \mid x_{query}, t^*)$,

$$\left|\bar{p}(y_{query} \mid \boldsymbol{z}_N^{(6)}) - p(y_{query} \mid x_{query}, t^*)\right| \longrightarrow \left|p(y_{query} \mid x_{query}, t^\dagger) - p(y_{query} \mid x_{query}, t^*)\right| > 0. \tag{132}$$

Therefore the finite analytic-continuation assumptions by themselves do not imply convergence for format 6.

Proof. Divide (122) by N. Since the query term is fixed, $\bar{\lambda}_{t,q}/N \to 0$. By the strong law of large numbers,

$$\frac{1}{N}\sum_{n=1}^{N} \ell_{t,n} \xrightarrow{\text{a.s.}} -\kappa_t. \tag{133}$$

Together with (127), this proves (128).

Under (129), finiteness of $\mathcal{T}$ gives a $\delta > 0$ such that, almost surely for all sufficiently large N,

$$\log \frac{\bar{p}(t \mid \boldsymbol{z}_N^{(6)})}{\bar{p}(t^* \mid \boldsymbol{z}_N^{(6)})} \le -\delta N \qquad \text{for all } t \neq t^*. \tag{134}$$

Summing the odds over the finite competing-task set yields $1 - \bar{p}(t^* \mid \boldsymbol{z}_N^{(6)}) = O(e^{-\delta N})$, and the posterior-predictive claim follows exactly as in (67).

Under (130), (128) gives

$$\log \frac{\bar{p}(t^\dagger \mid \boldsymbol{z}_N^{(6)})}{\bar{p}(t^* \mid \boldsymbol{z}_N^{(6)})} \longrightarrow +\infty \tag{135}$$

linearly in N. In the two-task case this implies (131); substituting it into the definition (109) gives (132). □

E Proof of Section 4

This appendix formalizes the two claims in Section 4.1: the Bayesian latent-task reference for direct ICL cannot assign posterior mass to a zero-prior target task, whereas its fixed-length, explicitly composed reference can converge by invoking pretraining-supported subtasks. This is not an impossibility result for an arbitrary Transformer. A guarantee for extracted step prompts does not by itself imply the same guarantee for a full-context autoregressive rollout. We quantify this distinction by a rollout discrepancy, prove the actual-model bound without exact rollout compatibility, and recover exact compatibility as a special case. The reference-rollout claim is stated separately.

E.1 Reduction of an autoregressive CoT rollout to supported steps

Fix an integer K that does not depend on N. For every $k = 0, \dots, K-1$, fix a dependency set $J_k \subseteq \{0, \dots, k\}$ with $k \in J_k$, independent of N, and a fixed history map $\mathrm{enc}_k : \mathcal{S}^{|J_k|} \to \mathcal{S}$ of the selected states into one step input. For any trajectory, write

$$h^{(k)} := \mathrm{enc}_k\Big((x^{(j)})_{j\in J_k}\Big), \qquad x^{(k+1)} := f_{t_k}(h^{(k)}), \qquad t_0, \dots, t_{K-1} \in \mathcal{T}. \tag{136}$$

This finite-history recursion defines the target map by $f_{t^\dagger}(x) := x^{(K)}$ when $x^{(0)} = x$. The special case $J_k = \{k\}$ recovers ordinary composition of one-step maps. For a query x_{query}, define its correct intermediate states and histories recursively by

$$x_q^{(0)} := x_{\text{query}}, \qquad h_q^{(k)} := \mathrm{enc}_k\Big((x_q^{(j)})_{j\in J_k}\Big), \qquad x_q^{(k+1)} := f_{t_k}(h_q^{(k)}), \quad k = 0, \dots, K-1. \tag{137}$$

Thus $x_q^{(K)} = f_{t^\dagger}(x_{\text{query}})$. Let $\mathcal{H}^{(k)}$ denote the set of encoded histories $h_q^{(k)}$ reachable at step k as the initial query ranges over the finite state space $\mathcal{S}$.

The full prompt $\boldsymbol{z}_N$ contains the demonstration trajectories $(x_n^{(0)}, \dots, x_n^{(K)})$. For each step k, let

$$h_n^{(k)} := \mathrm{enc}_k\Big((x_n^{(j)})_{j\in J_k}\Big), \qquad \boldsymbol{z}_N^{(k)}(h) := \Big(\boldsymbol{i}^{(k)}, \big(h_n^{(k)}, x_n^{(k+1)}\big)_{n=1}^{N}, h\Big) \tag{138}$$

be the corresponding single-step ICL prompt.

Assumption E.1 (Supported extracted-step domains)**.** Each $\boldsymbol{i}^{(k)}$ is supported for t_k, and every correct step prompt in (138) lies in the corresponding pretraining-supported step domain. Any use of a padded model evaluation must separately justify the local-model bound below and retain its padding discrepancy.

Assumption E.2 (Exact finite-history rollout compatibility for a specified predictor)**.** For a specified predictor R (either p or M), every $k = 0, \ldots, K-1$, every output u, and every correct prefix $x_q^{(0:k)}$, the next-token-block conditional of the full CoT rollout agrees with the conditional induced by the extracted single-step prompt:

$$R\Big(\widehat{x}_q^{(k+1)} = u \mid \boldsymbol{z}_N, \widehat{x}_q^{(0:k)} = x_q^{(0:k)}\Big) = R\Big(u \mid \boldsymbol{z}_N^{(k)}(h_q^{(k)})\Big). \tag{139}$$

An assertion for $R = p$ is distinct from one for $R = M$; neither implies the other. Conditionals are understood as generation kernels evaluated at the specified prefix, including prefixes with zero rollout probability.

Assumption E.2 is a sufficient condition for transferring a step-prompt bound with no compatibility correction. It is a substantive finite-history conditional-sufficiency assumption: on the correct prefix, the history–next-state demonstrations and the selected history $h_q^{(k)}$ contain all information from the full CoT prompt that is relevant to the next step. The states selected by J_k are not being treated as padding. Rather, under the deterministic subtask model the correct process satisfies

$$p\big(x_q^{(k+1)} \mid x_q^{(0:k)}, t_k\big) = p\big(x_q^{(k+1)} \mid h_q^{(k)}, t_k\big), \tag{140}$$

because $x_q^{(k+1)} = f_{t_k}(h_q^{(k)})$. Thus the selected finite history, rather than the current state alone, summarizes the correct query prefix for the target transition. Equation (139) requires the model conditional to respect the same sufficiency property. This does not follow from virtual padding: padding accounts only for the future query positions that have not yet been generated. States outside J_k may still appear in the full prompt, but the assumption requires that they provide no additional information about the next step once $h_q^{(k)}$ is given. If this finite-history sufficiency fails, then the equality in (139) can fail and an additional approximation error must be included in the final bound. In particular, deterministic sufficiency for the target transition does not establish sufficiency for a learned model's conditional. Enlarging J_k to include the complete query history also does not remove dependence on other demonstrations, absolute positions, or prompt formatting; the enlarged step domain must still satisfy supportedness and identifiability.

Quantifying the failure of exact compatibility. Write $u_k^* := f_{t_k}(h_q^{(k)})$ and define

$$Q_{k,N}^M(u \mid x_q^{(0:k)}) := M(\widehat{x}_q^{(k+1)} = u \mid \boldsymbol{z}_N, \widehat{x}_q^{(0:k)} = x_q^{(0:k)}), \tag{141}$$

$$P_{k,N}^M(u \mid h_q^{(k)}) := M(u \mid \boldsymbol{z}_N^{(k)}(h_q^{(k)})). \tag{142}$$

Both kernels are distributions over the same next-state token blocks, including a failure outcome for malformed or nonterminating outputs if needed. The full kernel includes all actually retained context. For fixed demonstrations, take suprema over correct query prefixes and define

$$\rho_{k,N} := \sup_{x_q^{(0:k)}} \|Q_{k,N}^M(\cdot \mid x_q^{(0:k)}) - P_{k,N}^M(\cdot \mid h_q^{(k)})\|_{\mathrm{TV}}, \tag{143}$$

$$\rho_{k,N}^+ := \sup_{x_q^{(0:k)}} [P_{k,N}^M(u_k^* \mid h_q^{(k)}) - Q_{k,N}^M(u_k^* \mid x_q^{(0:k)})]_+. \tag{144}$$

Here $[a]_+ = \max\{a, 0\}$ and $\|P - Q\|_{\mathrm{TV}} = \frac{1}{2}\sum_u |P(u) - Q(u)|$. In particular $0 \le \rho_{k,N}^+ \le \rho_{k,N} \le 1$. The one-sided discrepancy measures only a decrease in correct-transition probability caused by the full context. Additional context that improves that probability does not incur a penalty. Neither quantity is assumed small by definition. The actual-model theorem below uses $\rho_{k,N}^+$ and therefore also holds with the more conservative $\rho_{k,N}$.

Virtual padding for the step-k length mismatch. At step k, the current input may be shorter than its pretraining reference for two closely related reasons. First, the suffix $x_q^{(k+1)}, \ldots, x_q^{(K)}$ has not yet been generated. Second, the user's prompt may simply be shorter than the longer contexts commonly observed during pretraining. These are the same dimensional-alignment problem: in both cases we choose the pretraining reference length M_k and pad the shorter step prompt to that length. The only difference is where the padding tokens are inserted. Let

$$\mathrm{Pad}_{M_k}\Big(\boldsymbol{z}_N^{(k)}(h)\Big) \tag{145}$$

denote the resulting prompt after inserting m_k virtual padding tokens while leaving all informative tokens unchanged and preserving their relative order. The padded step prompt and the pretraining prompt then belong to the same fixed-length ambient space.

When the padding occupies only future positions, a causal mask gives those positions zero attention weight, so the padding has no effect on the current conditional. For padding inserted elsewhere to match a longer pretraining context, or in an abstract fixed-length attention model without an exact mask, the effect is only approximately zero. Lemma F.2 controls the attention perturbation by the aggregate padding mass, of order

$$m_k e^{-\Delta_{\mathrm{att},k}}, \tag{146}$$

where $\Delta_{\mathrm{att},k}$ is the attention-logit gap between an informative token and the most competitive padding token. Thus a large amount of weakly masked padding can still be non-negligible.

To record this approximation explicitly, define the padding discrepancy

$$\delta_{k,\mathrm{pad}}(N) := \sup_{h,u} \left| M\Big(u \mid \boldsymbol{z}_N^{(k)}(h)\Big) - M\Big(u \mid \mathrm{Pad}_{M_k}(\boldsymbol{z}_N^{(k)}(h))\Big) \right|. \tag{147}$$

Under exact causal masking this discrepancy is zero for padding placed only in future positions. Under the attention-stability conditions of Appendix F.3, it is small when the induced network-output perturbation is small. Combining the padded reference bound with the model-transport inequality in Section 3, the corresponding unpadded model error is at most

$$\sup_{h\in\mathcal{H}^{(k)}} \left[1 - M\Big(f_{t_k}(h) \mid \boldsymbol{z}_N^{(k)}(h)\Big)\right] \le \beta_k e^{-\alpha_k N} + \varepsilon_M^{(k)}(N) + \delta_{k,\mathrm{pad}}(N). \tag{148}$$

Here $\varepsilon_M^{(k)}(N)$ is an upper bound, uniform over $h \in \mathcal{H}^{(k)}$, on the transport certificates in (32) for the step-k prompts being compared with the reference. For supported unpadded step prompts one may take r_0; otherwise the supremum over the finitely many correct-history paths suffices at fixed N. The reference bound must apply to the same local or padded prompts used in this comparison; inserting padding alone does not establish supportedness or reference invariance. A smaller residual-variation certificate from Proposition D.9 may be used when its hypotheses hold uniformly over these histories.

Including the one-sided rollout discrepancy, the sequential argument below gives

$$\text{final CoT error} \le \sum_{k=0}^{K-1} \left(\beta_k e^{-\alpha_k N} + \varepsilon_M^{(k)}(N) + \delta_{k,\mathrm{pad}}(N) + \rho_{k,N}^{+}\right). \tag{149}$$

The formal theorem below retains both corrections explicitly. Padding controls the local prompt versus its padded version; it does not control the full prompt versus the extracted local prompt.

For deterministic subtasks, define the step error by

$$\epsilon_k(N) := \sup_{h\in\mathcal{H}^{(k)}} \left[1 - p\Big(f_{t_k}(h) \mid \boldsymbol{z}_N^{(k)}(h)\Big)\right], \tag{150}$$

where $\mathcal{H}^{(k)}$ is the set of encoded histories reachable at step k under the correct recursion. The desired next state for history h is explicitly $f_{t_k}(h)$.

Lemma E.3 (Uniform supported-step convergence)**.** *Suppose that, for every k, the extracted pairs $\big(h_n^{(k)}, x_n^{(k+1)}\big)_{n=1}^{N}$ follow the supported single-step generation mechanism and satisfy Assumption 3.1 for task t_k. Then there are constants $\alpha_k, \beta_k > 0$, independent of N, such that, almost surely for all sufficiently large N,*

$$\epsilon_k(N) \le \beta_k e^{-\alpha_k N}. \tag{151}$$

Proof. Fix k and $h \in \mathcal{H}^{(k)}$. Theorem 3.2, applied to the supported step task t_k and the correct output $f_{t_k}(h)$, gives constants $\alpha_{k,h}, \beta_{k,h} > 0$ and an almost-sure finite threshold $N_{k,h}$ such that

$$1 - p\Big(f_{t_k}(h) \mid \boldsymbol{z}_N^{(k)}(h)\Big) \le \beta_{k,h} e^{-\alpha_{k,h} N}$$

for all $N \ge N_{k,h}$. Because J_k is finite and each state has bounded length over the finite vocabulary $\mathcal{V}$, the reachable encoded-history set $\mathcal{H}^{(k)}$ is finite. We may therefore take

$$\alpha_k := \min_{h\in\mathcal{H}^{(k)}} \alpha_{k,h} > 0, \qquad \beta_k := \max_{h\in\mathcal{H}^{(k)}} \beta_{k,h} < \infty, \qquad N_k := \max_{h\in\mathcal{H}^{(k)}} N_{k,h} < \infty.$$

The intersection of the finitely many probability-one events also has probability one. Taking the supremum over $h \in \mathcal{H}^{(k)}$ proves (151) for all $N \ge N_k$. □

E.2 Formal rollout bound and its exact-compatibility specialization

Theorem E.4 (Model bound with rollout discrepancy)**.** *Let the fixed-length finite-history decomposition (136), Assumption E.1, and the hypotheses of Lemma E.3 hold, with the local-model bound (148). If the theoretical rollout additionally satisfies Assumption E.2 for $R = p$, then almost surely for all sufficiently large N and every output y,*

$$\left|p(y \mid \boldsymbol{z}_N) - p(y \mid x_{query}, t^\dagger)\right| \le \sum_{k=0}^{K-1} \beta_k e^{-\alpha_k N} \le K\beta_{\max} e^{-\alpha_{\min} N}, \tag{152}$$

where $\beta_{\max} := \max_k \beta_k$ and $\alpha_{\min} := \min_k \alpha_k > 0$. For the actual pretrained model, define

$$\eta_k(N) := \beta_k e^{-\alpha_k N} + \varepsilon_M^{(k)}(N) + \delta_{k,\mathrm{pad}}(N) + \rho_{k,N}^{+}. \tag{153}$$

Without requiring exact compatibility for either M or a theoretical full rollout, the actual full autoregressive rollout satisfies, almost surely for all sufficiently large N and every y,

$$\begin{aligned}\left|M(y \mid \boldsymbol{z}_N) - p(y \mid x_{query}, t^\dagger)\right| &\le 1 - \prod_{k=0}^{K-1} (1 - \min\{1, \eta_k(N)\}) \\ &\le \min\left\{1, \sum_{k=0}^{K-1} \eta_k(N)\right\}.\end{aligned} \tag{154}$$

Proof. For the reference claim, assume compatibility only for $R = p$. Let $\widehat{x}_q^{(0)} := x_{\text{query}}$ and let $\widehat{x}_q^{(1)}, \ldots, \widehat{x}_q^{(K)}$ be the states generated by the full autoregressive CoT rollout. For each k, let $\widehat{h}_q^{(k)} := \mathrm{enc}_k((\widehat{x}_q^{(j)})_{j \in J_k})$. Define

$$E_k := \left\{\widehat{x}_q^{(k+1)} = f_{t_k}(\widehat{h}_q^{(k)})\right\}, \qquad F_k := \bigcap_{j=0}^{k} E_j, \tag{155}$$

with F_{-1} equal to the whole sample space.

On F_{k-1}, induction using (137) gives $\widehat{x}_q^{(j)} = x_q^{(j)}$ for all $j \le k$, and in particular $\widehat{h}_q^{(k)} = h_q^{(k)} \in \mathcal{H}^{(k)}$. Assumption E.2 and (150) therefore imply

$$\mathbb{P}(E_k^c \mid F_{k-1}, \boldsymbol{z}_N) = 1 - p\Big(f_{t_k}(h_q^{(k)}) \mid \boldsymbol{z}_N^{(k)}(h_q^{(k)})\Big) \tag{156}$$

$$\le \epsilon_k(N). \tag{157}$$

No independence between the rollout steps is required. The conditional statement is used only when $\mathbb{P}(F_{k-1} \mid \boldsymbol{z}_N) > 0$; otherwise the joint event $E_k^c \cap F_{k-1}$ has zero probability and the recursion below is immediate.

Using $F_k^c = F_{k-1}^c \cup (E_k^c \cap F_{k-1})$ and the disjointness of this union,

$$\mathbb{P}(F_k^c \mid \boldsymbol{z}_N) \le \mathbb{P}(F_{k-1}^c \mid \boldsymbol{z}_N) + \mathbb{P}(E_k^c \mid F_{k-1}, \boldsymbol{z}_N)\mathbb{P}(F_{k-1} \mid \boldsymbol{z}_N) \tag{158}$$

$$\le \mathbb{P}(F_{k-1}^c \mid \boldsymbol{z}_N) + \epsilon_k(N). \tag{159}$$

Iterating this inequality yields

$$\mathbb{P}(F_{K-1}^c \mid \boldsymbol{z}_N) \le \sum_{k=0}^{K-1} \epsilon_k(N). \tag{160}$$

On F_{K-1}, every step is correct and

$$\widehat{x}_q^{(K)} = f_{t^\dagger}(x_{\text{query}}).$$

Consequently,

$$p(f_{t^\dagger}(x_{\text{query}}) \mid \boldsymbol{z}_N) \ge 1 - \sum_{k=0}^{K-1} \epsilon_k(N). \tag{161}$$

Because the target-task conditional is deterministic, $p(y \mid x_{\text{query}}, t^\dagger) = \mathbb{1}\{y = f_{t^\dagger}(x_{\text{query}})\}$. If $y = f_{t^\dagger}(x_{\text{query}})$, equation (161) bounds $1 - p(y \mid \boldsymbol{z}_N)$. If $y \neq f_{t^\dagger}(x_{\text{query}})$, then $p(y \mid \boldsymbol{z}_N)$ is at most the failure probability in (160). Thus, in both cases,

$$\left|p(y \mid \boldsymbol{z}_N) - p(y \mid x_{\text{query}}, t^\dagger)\right| \leq \sum_{k=0}^{K-1} \epsilon_k(N). \tag{162}$$

Finally apply Lemma E.3 and use the finiteness of K:

$$\sum_{k=0}^{K-1} \epsilon_k(N) \leq \sum_{k=0}^{K-1} \beta_k e^{-\alpha_k N} \leq K \beta_{\max} e^{-\alpha_{\min} N}.$$

This proves (152).

For the model-level claim, start independently with states sampled from M. On a correct prefix, (144) gives

$$1 - Q_{k,N}^M(u_k^* \mid x_q^{(0:k)}) \leq 1 - P_{k,N}^M(u_k^* \mid h_q^{(k)}) + \rho_{k,N}^+.$$

Together with (148), this implies, whenever $\mathbb{P}_M(F_{k-1} \mid \boldsymbol{z}_N) > 0$,

$$\mathbb{P}_M(E_k^c \mid F_{k-1}, \boldsymbol{z}_N) \leq \min\{1, \eta_k(N)\}.$$

Set $a_k = \min\{1, \eta_k(N)\}$. Multiplication of the conditional success bounds yields $\mathbb{P}_M(F_{K-1} \mid \boldsymbol{z}_N) \geq \prod_k (1 - a_k)$. If a preceding success event has probability zero, the recursive inequality $\mathbb{P}_M(F_k \mid \boldsymbol{z}_N) \geq (1 - a_k)\mathbb{P}_M(F_{k-1} \mid \boldsymbol{z}_N)$ remains valid without conditioning on a null event. The elementary inequality $1 - \prod_k (1 - a_k) \leq \sum_k a_k$ gives the additive bound. On F_{K-1} the final output is $f_{t^\dagger}(x_{\text{query}})$, so the same two-case argument for $y = f_{t^\dagger}(x_{\text{query}})$ and $y \neq f_{t^\dagger}(x_{\text{query}})$ proves (154). □

Corollary E.5 (Execution through explicit step calls)**.** *Suppose an executor constructs $\boldsymbol{z}_N^{(k)}(h)$ at each step and calls the same model with exactly that context, position convention, and decoding settings. It passes the generated state to the next step and retains no additional model-visible history or cache. Its kernel equals $P_{k,N}^M$ by construction, so $\rho_{k,N} = \rho_{k,N}^+ = 0$. Consequently its final error is at most*

$$\min\left\{1, \sum_{k=0}^{K-1} [\beta_k e^{-\alpha_k N} + \varepsilon_M^{(k)}(N) + \delta_{k,\text{pad}}(N)]\right\}.$$

Proof. The executor's transition kernel is precisely the model kernel on the constructed step prompt, including on every correct prefix. Substitute zero discrepancy into Theorem E.4. □

Scope of the main-text bound. The bound displayed in Theorem 4.2 is the specialization with zero one-sided rollout discrepancy and zero padding discrepancy (or with such corrections explicitly included in the step error). This is guaranteed by the step-call protocol above with supported, unpadded prompts, or follows under exact model compatibility and exact padding invariance. For ordinary full-context autoregressive execution, the general bound is (154); supported subtasks alone do not remove its rollout correction. An exactly composed theoretical reference may be defined using the local reference kernels, but this construction is not an identity with an arbitrary full-context reference predictor.

For fixed K, vanishing reference errors and vanishing model, padding, and one-sided rollout corrections imply convergence of the actual final output. Uniformly small nonzero corrections give an error floor bounded by their sum. No independence between steps is needed. Errors following an incorrect prefix need not be controlled: the proof already charges the first failure, and later recovery can only reduce the final error relative to this bound.

Assessing compatibility empirically. On matched demonstrations and a teacher-forced correct prefix, evaluate both the full-context kernel and the extracted-step kernel. In finite-state experiments their total variation and correct-state probability difference can be computed directly; in large-output experiments the latter tests the one-sided quantity without estimating the whole distribution. Varying history, unrelated demonstrations, positions, and formatting while keeping the local transition fixed

probes the discarded information. A comparison with explicit step calls separates execution-protocol effects from subtask learning. Measurements on sampled prefixes are diagnostics, not supremum certificates. The inequality between final error and the sum of full-rollout conditional errors holds even without compatibility, so verifying that inequality alone does not validate the local-to-full transfer.

E.3 Growing horizons and comparison of supplied plans

The sequential argument is nonasymptotic in the number of steps: for each finite K it bounds final error whenever the corresponding local error bounds hold. Fixed K was used to obtain a common almost-sure onset of those bounds by taking a maximum over finitely many steps and histories. To allow $K = K(N)$, one must control the local bounds simultaneously along the growing plan; pointwise convergence for each fixed step does not suffice.

For each N, let Π_N be a supplied, valid finite-history plan with $K_N := K(N) \geq 1$ steps, realizing a deterministic target $f_N^\dagger$. The target, operations, and history maps may depend on N. The extracted prompts must satisfy the same supported-domain conditions as above, and the actual executor must be specified. Define its conditional final error by

$$e_N := 1 - M_{\Pi_N}(f_N^\dagger(x_{\text{query}}) \mid \boldsymbol{z}_N).$$

Here M_{Π_N} is the final-output law of that executor, which may use full-context rollout or explicit step calls. All claims below are uniform over queries if their local premises are uniform over the associated correct prefixes. Since the target is deterministic, e_N also bounds every output-probability coordinate difference from the target law.

Corollary E.6 (Growing-horizon execution)**.** *Suppose that for each N, on an event G_N in the demonstration randomness, the following local bounds hold simultaneously for every $k < K_N$ and every correct history of Π_N:*

$$1 - P_{k,N}^M(u_k^* \mid h_q^{(k)}) \leq \beta_{k,N} e^{-\alpha_{k,N} N} + \varepsilon_{M,N}^{(k)} + \delta_{k,\text{pad}}(N), \tag{163}$$

where $\alpha_{k,N}, \beta_{k,N} > 0$. With the one-sided rollout discrepancy from (144), put

$$c_{k,N} := \varepsilon_{M,N}^{(k)} + \delta_{k,\text{pad}}(N) + \rho_{k,N}^+, \tag{164}$$

$$a_{k,N} := \min\{1, \beta_{k,N} e^{-\alpha_{k,N} N} + c_{k,N}\}, \tag{165}$$

$$a_N := \min_{k<K_N} \alpha_{k,N}, \qquad b_N := \max_{k<K_N} \beta_{k,N}, \qquad C_N := \sum_{k<K_N} c_{k,N}. \tag{166}$$

Then on G_N,

$$e_N \leq 1 - \prod_{k=0}^{K_N-1} (1 - a_{k,N}) \leq \min\{1, K_N b_N e^{-a_N N} + C_N\}. \tag{167}$$

In particular, sufficient conditions for $e_N \to 0$ on sample paths on which G_N holds eventually are

$$a_N N - \log K_N - \log b_N \longrightarrow +\infty, \qquad C_N \longrightarrow 0. \tag{168}$$

If $c_{k,N} \leq \bar{c}_N$ uniformly, the second condition is implied by $K_N \bar{c}_N \to 0$.

Proof. On G_N, the definition of $\rho_{k,N}^+$ transfers (163) to a conditional failure bound $a_{k,N}$ for the full executor on a correct prefix. The success-event recursion in Theorem E.4, applied at this particular N for K_N steps, gives the product bound. The union bound and $\sum_{k<K_N} \beta_{k,N} e^{-\alpha_{k,N} N} \leq K_N b_N e^{-a_N N}$ give the second inequality. Under (168) its right-hand side tends to zero. No independence between steps is used. □

When are the simultaneous premises justified? An almost-sure version requires that G_N hold eventually on a common probability-one event. This does not follow merely from almost-sure convergence for every fixed k: step-dependent onset thresholds may grow faster than N along newly introduced steps. One sufficient construction is a fixed finite library of supported local prompt problems, with a fixed finite collection of admissible encoded histories. For each library problem,

use length-N prefixes of the same underlying infinite demonstration stream, and instantiate steps by reusing those local problems. If each library problem satisfies Lemma E.3, the minimum rate, maximum prefactor, and maximum onset threshold over the finite library and histories are uniform for any number of repetitions. Explicit step calls with a fixed bank of demonstrations for each library problem provide one such protocol; arbitrary trajectory-extracted demonstration marginals need not have this property. This establishes the simultaneous reference part of the local bound; model transport, padding, and rollout discrepancy still require their own controls. Enlarging history encodings with K_N, or drawing new datasets at every N, is not covered by this finite-library argument without additional estimates.

Alternatively, suppose a finite-sample analysis gives $\Pr(G_N^c) \le \delta_N$. Then (167) holds with probability at least $1 - \delta_N$. If its right-hand side is bounded on G_N by a deterministic $B_N \in [0, 1]$, then

$$\mathbb{E}[e_N] \le B_N + (1 - B_N)\delta_N \le B_N + \delta_N. \tag{169}$$

Indeed, split the expectation over G_N and its complement and use $e_N \le 1$. If each of the K_N step bounds, already uniform over its correct histories, fails with probability at most d_N, a union bound gives $\delta_N \le \min\{1, K_N d_N\}$ without independence. A per-history guarantee would require a further union bound or another uniformity argument. When $\sum_N \delta_N < \infty$, the first Borel–Cantelli lemma gives eventual validity almost surely. Together with (168) this recovers almost-sure convergence; $\delta_N \to 0$ alone yields the corresponding convergence-in-probability statement when $B_N \to 0$.

Explicit scaling regimes. Suppose $a_N \ge \alpha > 0$ and $b_N \le \beta < \infty$ uniformly. For a composed reference, or an ideal executor with zero corrections, $\log K_N = o(N)$ is sufficient. More generally, for $K_N = O(e^{\gamma N})$ the reference contribution vanishes when $\gamma < \alpha$. For actual-model execution the correction sum is equally important:

- If $K_N = O(N^q)$ and $\bar{c}_N = O(N^{-p})$, with $0 \le q < p$, then $e_N = O(N^q e^{-\alpha N} + N^{q-p})$ on the simultaneous good events.
- If $K_N = O(e^{\gamma N})$ and $\bar{c}_N = O(e^{-\lambda N})$, then $e_N = O(e^{-(\alpha-\gamma)N} + e^{-(\lambda-\gamma)N})$ for $0 \le \gamma < \min\{\alpha, \lambda\}$.
- A fixed positive upper bound on $\bar{c}_N$ does not provide a vanishing growing-horizon certificate, even if the reference identification term vanishes. This is a limitation of the guarantee, not a lower bound on every model's error.

These conclusions require the uniform rates and prefactors just stated. If longer plans introduce less identifiable operations or more difficult query states, the general condition involving a_N, b_N must be used instead.

The accumulation can be unavoidable. Consider an executor which, conditional on a correct prefix, makes an error with probability q_N at every step, and whose error state is absorbing with an incorrect final output. Its exact final error is $1 - (1 - q_N)^{K_N}$. It tends to zero if and only if $K_N q_N \to 0$: sufficiency follows from the union bound, and necessity from $1 - (1 - q_N)^{K_N} \ge 1 - e^{-K_N q_N}$. Thus constant conditional error cannot support arbitrary horizon growth in general. This example concerns actual conditional failures, not the possibly loose upper certificates $c_{k,N}$.

Demonstration complexity for a requested horizon. For a supplied horizon K and tolerance $0 < \epsilon < 1$, uniform constants α, β and a correction bound $\bar{c}_N$ give the sufficient conditions

$$N \ge \max\{0, \alpha^{-1} \log(2K\beta/\epsilon)\}, \qquad \bar{c}_N \le \frac{\epsilon}{2K}, \tag{170}$$

provided the simultaneous local bounds are valid at that N. Substitution in (167) gives $e_N \le \epsilon$. Only the identification contribution has logarithmic dependence on K; the required accuracy of the corrections tightens as $1/K$. Moreover, N counts demonstrations per step, not total tokens. A full prompt containing N trajectories of length K generally has length of order NK for bounded-size states. The asymptotic regimes presume that the required contexts and states remain admissible. Explicit step calls can avoid storing all trajectories in each call, but still incur K model invocations.

Comparing plans of different lengths. For supplied plans Π realizing the same target at a given N, define the untruncated certificate

$$S_N(\Pi) := \sum_{k<K(\Pi)} [\beta_{\Pi,k,N} e^{-\alpha_{\Pi,k,N} N} + c_{\Pi,k,N}]. \tag{171}$$

Each plan has $e_N(\Pi) \le \min\{1, S_N(\Pi)\}$ whenever its premises hold. Consequently a longer plan may have a smaller certificate if its steps are more identifiable or have smaller corrections. For example, with zero corrections and homogeneous constants within each of two plans, a longer plan Π_ℓ has a smaller untruncated certificate than Π_s exactly when

$$\alpha_\ell - \alpha_s > \frac{1}{N} \log \frac{K(\Pi_\ell)\beta_\ell}{K(\Pi_s)\beta_s}.$$

This follows by taking logarithms of their positive exponential certificates. It compares guarantees, not actual errors, and does not imply that the shortest plan or the plan with the best bound is always empirically best. Selecting among candidate plans using the same data requires simultaneous validity over the candidates; validity for each fixed plan alone does not justify adaptive selection.

Separation from plan discovery. These results analyze a supplied plan family and do not establish that a model can discover a valid decomposition. To record the missing component, let a planner output $\widehat{\Pi}_N$, and let A_N be the event that it realizes the intended target using admissible supported steps and satisfies the required execution certificates. If $\Pr(A_N^c) \le \xi_N$ and, conditional on the planner output and its data, execution error on A_N is at most a deterministic $B_N \in [0,1]$, then the law of total probability gives

$$\Pr(\text{final failure}) \le \xi_N + (1-\xi_N) B_N.$$

No independence of planning and execution is needed, but the conditional execution guarantee is essential. Bounding ξ_N or designing a planner that makes it small is a separate problem not solved by the composition theorem. The growing-horizon result strengthens execution scaling without assuming that plan discovery follows from support of the component operations.

F Formal Justification of the Padding Reduction

This appendix formalizes the fixed-length padding reduction used in two places in the paper. First, the instruction path in Section 3.2.1 must compare instructions containing different numbers of tokens in one common Euclidean space. Second, the step-wise CoT analysis in Appendix E sometimes compares a partially generated prompt with a longer pretraining reference. These are instances of the same dimensional-alignment problem, although the padding tokens occupy different positions. We define the common padding operator, quantify its effect on attention, and state precisely what this reduction does and does not establish.

F.1 Two alignment problems

The first problem is *instruction-length alignment*. The path

$$E_s = (1-s)E(\boldsymbol{i}_0) + sE(\widetilde{\boldsymbol{i}}), \qquad s \in [0,1], \tag{172}$$

requires $E(\boldsymbol{i}_0)$ and $E(\widetilde{\boldsymbol{i}})$ to belong to the same space, even when $\boldsymbol{i}_0$ and $\widetilde{\boldsymbol{i}}$ contain different numbers of tokens.

The second problem is *rollout-length alignment*. At CoT step k, the prompt $\boldsymbol{z}_N^{(k)}(h)$ may be shorter than the reference prompt used for the corresponding pretraining-supported subtask. One reason is that the future query states $x_q^{(k+1)}, \dots, x_q^{(K)}$ have not yet been generated; another is that the selected pretraining reference may simply contain a longer context. Virtual padding places both prompts in a common fixed-length ambient space.

Padding creates a common representation by definition. Its effect on the model, however, is small only under additional masking or attention-separation conditions. The results below make this distinction explicit.

F.2 RIGHT-PADDED INSTRUCTION REPRESENTATIONS

Recall the token embedding map $e : \mathcal{V} \cup \{\langle \mathrm{pad} \rangle\} \to \mathbb{R}^d$ from Section 2, with $e(\langle \mathrm{pad} \rangle) = 0$. For $\boldsymbol{i} = (v_1, \ldots, v_m)$ with $m \le L_i$, its right-padded representation is

$$E(\boldsymbol{i}) := \big[e(v_1), \ldots, e(v_m), 0, \ldots, 0\big]^\top \in \mathbb{R}^{L_i \times d}. \tag{173}$$

Thus every instruction occupies the same L_i-position instruction slot.

Remark F.1 (Why the position convention matters). Inserting padding before an informative token and then reindexing the prompt can change that token's absolute or rotary position. A zero token embedding does not prevent this positional change. Right-padding is therefore the canonical position-preserving convention for the instruction sequence itself: it leaves the indices of all instruction tokens unchanged. When the instruction is embedded into a larger prompt, we additionally assume either that the prompt reserves an L_i-position instruction slot or that the position identifiers of the subsequent informative tokens are preserved. Without one of these alignment conventions, the informative representations can differ already at the input layer, and the zero-initial-perturbation condition used below need not hold.

For a general prompt $\boldsymbol{z}$, let $\mathrm{Pad}_M(\boldsymbol{z})$ denote its position-preserving extension to length M: informative tokens and their relative order are unchanged, and the added positions contain $\langle \mathrm{pad} \rangle$. In the applications below, the added positions are either trailing positions, reserved instruction-slot positions, or masked future positions.

F.3 ATTENTION STABILITY UNDER PADDING

We first isolate one attention head at a fixed informative query position. Let $\mathcal{I}$ and $\mathcal{P}$ index the informative and padding positions, respectively, with $|\mathcal{P}| = m$. The following result conditions on the query, keys, and values of the informative positions being aligned between the two computations. This condition is automatic for the first layer under the position convention above and is imposed layer by layer in the network-level result that follows.

Write x_i for the attention logit and v_i for the value vector at position i, and define

$$y = \frac{1}{Z} \sum_{i \in \mathcal{I}} e^{x_i} v_i, \qquad y' = \frac{1}{Z + Z_{mathrmp}} \left(\sum_{i \in \mathcal{I}} e^{x_i} v_i + \sum_{p \in \mathcal{P}} e^{x_p} v_p \right), \tag{174}$$

where $Z := \sum_{i \in \mathcal{I}} e^{x_i}$ and $Z_{mathrmp} := \sum_{p \in \mathcal{P}} e^{x_p}$.

Lemma F.2 (Attention stability under padding)**.** *Suppose $\|v_i\|_2 \le C_V$ at every informative and padding position. Let*

$$x_{\mathrm{p,max}} := \max_{p \in \mathcal{P}} x_p, \qquad \Delta_{\mathrm{att}} := x_j - x_{\mathrm{p,max}} > 0 \tag{175}$$

for some informative position $j \in \mathcal{I}$. Then

$$\|y' - y\|_2 \le 2C_V \frac{Z_{\mathrm{p}}}{Z + Z_{\mathrm{p}}} \le 2C_V \frac{Z_{\mathrm{p}}}{Z} \le 2C_V m e^{-\Delta_{\mathrm{att}}}. \tag{176}$$

Consequently, for any $\varepsilon > 0$, the condition

$$m \le \frac{\varepsilon e^{\Delta_{\mathrm{att}}}}{2C_V} \tag{177}$$

is sufficient for $\|y' - y\|_2 \le \varepsilon$.

Proof. Let $A := \sum_{i \in \mathcal{I}} e^{x_i} v_i$ and $B := \sum_{p \in \mathcal{P}} e^{x_p} v_p$. Subtracting the two expressions in (174) gives

$$y' - y = \frac{B}{Z + Z_{\mathrm{p}}} - \frac{Z_{\mathrm{p}} A}{Z(Z + Z_{\mathrm{p}})}. \tag{178}$$

Because $\|A\|_2 \le C_V Z$ and $\|B\|_2 \le C_V Z_{\mathrm{p}}$, the triangle inequality yields

$$\|y' - y\|_2 \le 2C_V \frac{Z_{\mathrm{p}}}{Z + Z_{\mathrm{p}}}. \tag{179}$$

Moreover, $Z_{\mathrm{p}} \le m e^{x_{\mathrm{p,max}}}$ and $Z \ge e^{x_j}$, and hence $Z_{\mathrm{p}}/Z \le m e^{-\Delta_{\mathrm{att}}}$. This proves (176); (177) follows immediately. □

Remark F.3 (Exactly masked future positions). In the first rollout-alignment case, padding occupies positions representing future query states. If these positions lie strictly after the current prediction position, the causal mask assigns them attention logit $-\infty$. Then $Z_{\mathrm{p}} = 0$, and the perturbation in Lemma F.2 is exactly zero. The quantitative bound is needed only when padding remains visible to an informative query, as may happen in an instruction slot or when matching a longer visible pretraining context.

F.4 From attention perturbations to output log-probabilities

Lemma F.2 concerns a single attention call. To obtain a model-level statement, the perturbation must be propagated through all layers. We state this step in a form that accommodates multi-head attention, output projections, residual connections, normalization, and MLP blocks.

Consider a Transformer with H blocks. After aligning the informative positions, let $\eta^{(\ell)}$ be the maximum norm of the hidden-state difference over those positions after block ℓ. Assume that, on the compact set of hidden states reached by the two forward passes,

(i) block ℓ is K_ℓ-Lipschitz with respect to perturbations of the informative hidden states; and

(ii) with the informative input states held fixed, adding the padding positions changes the informative outputs of block ℓ by at most ε_ℓ.

Lemma F.2, combined with the head-concatenation and attention-output projection norms, gives a bound of the form

$$\varepsilon_\ell \le C_{\mathrm{att},\ell} m e^{-\Delta_{\mathrm{att},\ell}} \tag{180}$$

whenever the value norms and attention gaps are controlled uniformly at that layer.

Proposition F.4 (Transformer-output stability under padding)**.** *Suppose the position-alignment condition of Remark F.1 holds, so that $\eta^{(0)} = 0$. Under conditions* (i)–(ii)*,*

$$\eta^{(H)} \le \sum_{\ell=0}^{H-1} \varepsilon_\ell \prod_{r=\ell+1}^{H-1} K_r, \tag{181}$$

where an empty product equals one. If the output readout is K_{out}-Lipschitz from the hidden-state norm to the ℓ_∞ logit norm, then, for every output token u,

$$|\log M(u \mid \mathrm{Pad}_M(\boldsymbol{z})) - \log M(u \mid \boldsymbol{z})| \le \varepsilon_{\mathrm{pad}} := 2K_{\mathrm{out}} \sum_{\ell=0}^{H-1} \varepsilon_\ell \prod_{r=\ell+1}^{H-1} K_r. \tag{182}$$

Proof. At block ℓ, add and subtract the padded block output evaluated at the unpadded informative hidden states. Conditions (i)–(ii) then give

$$\eta^{(\ell+1)} \le K_\ell \eta^{(\ell)} + \epsilon_\ell. \tag{183}$$

Starting from $\eta^{(0)} = 0$ and iterating this recursion proves (181). For logit vectors a, b,

$$|log\,\mathrm{softmax}(a)_u - \log \mathrm{softmax}(b)_u| \le 2\|a - b\|_\infty, \tag{184}$$

because the log-sum-exp map is one-Lipschitz in the ℓ_∞ norm. Combining this fact with the readout bound proves (182). □

The products of the block Lipschitz constants in (181) are worst-case amplification factors and need not be tight. The substantive quantity inherited from Lemma F.2 is $me^{-\Delta_{\mathrm{att},\ell}}$: many weakly masked padding positions can produce a non-negligible perturbation, whereas a sufficiently large attention gap suppresses their aggregate effect exponentially.

F.5 Consequence for the instruction path

Corollary F.5 (A common space for instructions)**.** *If $|\boldsymbol{i}_0|, |\widetilde{\boldsymbol{i}}| \le L_i$, then*

$$\Delta_{\mathrm{inst}} := \|E(\widetilde{\boldsymbol{i}}) - E(\boldsymbol{i}_0)\|_F \tag{185}$$

is well defined, and the entire path in (172) lies in $\mathbb{R}^{L_i \times d}$. The number of padding positions in either endpoint is at most L_i and is therefore independent of the number N of demonstrations.

The corollary supplies the dimensional fact required by the finite-continuation condition; it does not itself prove the existence of the analytic continuation assumed there. In the paper's formulation, the continued instruction likelihood is defined directly on the padded instruction-embedding space, so no extra padding term is required in the continuation bound. If an implementation instead compares padded model evaluations with their unpadded counterparts, Proposition F.4 provides an N-independent endpoint perturbation. Such a fixed perturbation changes only the prefactor in Theorem 3.3, not its exponential rate in N.

F.6 Consequence for the CoT rollout

Recall from Appendix E that the padding discrepancy at step k is

$$\delta_{k,\mathrm{pad}}(N) := \sup_{h,u} \left| M\Big(u \mid \boldsymbol{z}_N^{(k)}(h)\Big) - M\Big(u \mid \mathrm{Pad}_{M_k}(\boldsymbol{z}_N^{(k)}(h))\Big) \right|. \tag{186}$$

Corollary F.6 (Padding contribution to the CoT bound)**.** *Suppose Proposition F.4 holds uniformly over the encoded histories h and outputs u in (186). Then*

$$\delta_{k,\mathrm{pad}}(N) \le \varepsilon_{\mathrm{pad}}^{(k)}(N), \tag{187}$$

where $\varepsilon_{\mathrm{pad}}^{(k)}(N)$ is obtained from (182) using the step-k padding count and layerwise attention gaps. Consequently, the CoT error obeys

$$\left|M(y_{\textit{query}} \mid \boldsymbol{z}_N) - p(y_{\textit{query}} \mid x_{\textit{query}}, t^\dagger)\right| \le \sum_{k=0}^{K-1} \Big(\beta_k e^{-\alpha_k N} + \varepsilon_M^{(k)}(N) + \delta_{k,\mathrm{pad}}(N)\Big). \tag{188}$$

If the added positions represent only masked future states, then Remark F.3 gives $\delta_{k,\mathrm{pad}}(N) = 0$. If the step prompt is itself supported, then $\varepsilon_M^{(k)}(N) = r_0$, and (188) reduces to the bound in Theorem 4.2.

Proof. For probabilities $a, b \in (0, 1]$, the mean-value theorem applied to the exponential function on $(-\infty, 0]$ gives $|a - b| \le |\log a - \log b|$. Hence (182), uniformly over x and u, implies (187). Adding this discrepancy and the model-transport bound from Section 3 to the supported-step error, and repeating the union-bound argument used in the proof of Theorem 4.2 gives (188). The exactly masked case follows from Remark F.3. □

For visible padding used to match a longer pretraining context, neither the padding count nor the attention gap is automatically favorable; the additional term in (188) must then be retained. In particular, the fact that a prompt becomes longer as N grows does not by itself prove that the padded and unpadded conditionals become closer. The original The padding-free form of Theorem 4.2 is recovered only when padding is exactly masked or when $\delta_{k,\mathrm{pad}}(N)$ is otherwise negligible at the desired rate.

G Experiments

In this section we provide detailed introduction of our experiments setup and additional experimental results to supplement those in the main text.

G.1 Pretraining Dependence

To isolate the role of *intrinsic ICL capability*, we construct three variants of `Llama3.2-3B` that differ only in the amount and *format* of additional pretraining on a six-digit addition corpus. Throughout, we use the same evaluation protocol in the main text (6–10 digit addition).

Base (Weak). We use the original `Llama3.2-3B` checkpoint without any additional training.

Fine-tuned (Strong). We randomly generate 20,000 six-digit addition instances and format each instance as a supervised instruction-following sample in JSONL, each training example is serialized as

```
Instruction: ...  Input: ...  Response: ...
```

and we fine-tune the model using LoRA on this dataset.

Fine-tuned (Medium). We use the *same* 20,000 generated instances, but concatenate all serialized training examples into a single long text sequence and fine-tune on this single sequence. This construction preserves the token-level training signal while substantially reducing the effective diversity and sampling of training contexts, leading to a weaker improvement in intrinsic ICL capability compared to the sample-level training above.

Both **Strong** and **Medium** variants are trained with the same LoRA configuration and optimizer settings. We apply LoRA to attention and MLP projection modules. We train a causal LM objective with labels equal to the input tokens (standard next-token prediction over the serialized prompt). Unless otherwise stated, we use a maximum sequence length of 512 tokens with padding/truncation.

G.2 Demonstration Choice

In this section, our goal is to isolate how the informativeness of demonstrations affects in-context generalization, under the same prompt template, the same query distribution, and the same number of demonstrations.

Tasks. We consider two retrieval-style tasks constructed as "$(x, ?, y)$" completion: (i) **Sports**: x is a country and y is its designated sport label in our benchmark; (ii) **People**: x is a company and y is a canonical person name associated with that company in our benchmark (e.g., a founder). The model is queried with an unseen x_{test} and asked to predict the corresponding y_{test}.

Ambiguous vs. identifying demonstration pools. For each task, we build two disjoint pools of demonstration pairs: *ambiguous* and *identifying*. Both pools contain 10 labeled pairs (x, y). Intuitively, the ambiguous pool contains examples that can be explained by multiple plausible latent rules (hence providing weaker evidence about which rule the prompt instantiates), whereas the identifying pool contains examples that more strongly pin down a unique rule. Importantly, the *test set* (query distribution) is kept the same across the two conditions; only the demonstration pool differs.

Prompt template. Given a demonstration set $\{(x_j, y_j)\}_{j=1}^{k}$, we format the prompt as

$$\texttt{Question:}\quad (x_1, ?, y_1)\texttt{, ...,}\ (x_k, ?, y_k)\texttt{.}\quad x_{\text{test}}, ?,$$

and instruct the model to output a single answer token span in English (sport name for Sports; person name for People).

Evaluation. For each task and each condition (ambiguous / identifying), we generate 100 prompts per random seed by sampling $k \in \{2, 4, 8\}$ uniformly and sampling k demonstrations uniformly without replacement from the corresponding pool. The query pair $(x_{\text{test}}, y_{\text{test}})$ is sampled from a fixed test list with ground-truth labels. Model outputs are parsed into JSON and normalized by lowercasing and removing punctuation; accuracy is computed by exact match with the normalized ground truth. We repeat the entire procedure for 5 random seeds and report mean $\pm$ standard deviation across seeds.

Results. Tables 5–6 summarize the accuracy under the two demonstration pools as a function of k on `Qwen-235B-A22B` and `Qwen-30B-A3B`, respectively. Across both tasks, the identifying pool consistently yields higher accuracy than the ambiguous pool, and the performance gap generally increases with more demonstrations.

Table 5: **Improved Demonstration Experiment.** Task: sports, repeated 5 times with different random seeds.

Demo number	Ambiguous Acc (235B)	Identifying Acc (235B)	Ambiguous Acc (30B)	Identifying Acc (30B)
2	0.265 ± 0.073	0.421 ± 0.080	0.122 ± 0.029	0.176 ± 0.062
4	0.267 ± 0.084	0.542 ± 0.048	0.151 ± 0.078	0.410 ± 0.059
8	0.202 ± 0.037	0.602 ± 0.043	0.164 ± 0.039	0.559 ± 0.074

Table 6: **Improved Demonstration Experiment.** Task: People, repeated 5 times with different random seeds.

Demo number	Ambiguous Acc (235B)	Identifying Acc (235B)	Ambiguous Acc (30B)	Identifying Acc (30B)
2	0.713 ± 0.074	0.940 ± 0.049	0.735 ± 0.083	0.908 ± 0.059
4	0.708 ± 0.096	0.931 ± 0.027	0.775 ± 0.037	0.950 ± 0.058
8	0.648 ± 0.008	0.845 ± 0.044	0.820 ± 0.027	0.990 ± 0.001

G.3 Controlled Validation of Demonstration Evidence Rates

The preceding natural-language experiments vary whether demonstrations identify the intended task. We complement them with a controlled modular-arithmetic experiment in which the amount of identifying evidence is known exactly. This isolates the evidence-accumulation mechanism in Assumption 3.1 and tests the exponential concentration predicted by Theorem 3.2.

Controlled task family. We use three latent tasks on $\mathbb{Z}_{17}$, $t \in \{\mathrm{add}, \mathrm{sub}, \mathrm{mul}\}$, with addition as the true task. For an input $x = (a, b)$, the three clean outputs are $a + b$, $a - b$, and ab modulo 17. Outputs follow a symmetric noise model with probability 0.95 on the task-consistent answer and $0.05/16$ on each other answer, and the task prior is uniform. The ambiguous distribution q_{amb} is concentrated on $(0, 0)$, where all three tasks agree. The identifying distribution q_{id} is uniform over the 242 inputs for which the three outputs are pairwise distinct. We continuously control evidence through

$$q_\lambda = (1 - \lambda) q_{\mathrm{amb}} + \lambda q_{\mathrm{id}}, \qquad \lambda \in \{0, .25, .5, .75, 1\}. \tag{189}$$

For either competing task t, the population evidence rate is available in closed form:

$$\alpha_t(\lambda) = \mathbb{E}_{x \sim q_\lambda,\, y \sim p(\cdot \mid x, t^*)} \log \frac{p(y \mid x, t^*)}{p(y \mid x, t)} = 5.4133106\, \lambda. \tag{190}$$

Thus λ changes only the frequency of identifying examples, giving a direct experimental control over the rate appearing in the theory.

Reference evidence accumulation. For each λ and $N \in \{1, 2, 4, 8, 16, 32, 64\}$, we sample 2,000 independent prompts and measure the cumulative log-likelihood ratio $S_{N,t} = \sum_{n=1}^{N} \log[p(y_n \mid x_n, t^*)/p(y_n \mid x_n, t)]$. Table 7 compares the exact rate in Equation (190) with the fitted slope of $\mathbb{E}[S_{N,t}]$ against N. The two competing tasks give nearly identical slopes, every fit has $R^2 > 0.99999$, and all empirical slopes are within 0.8% of the exact population value. At $\lambda = 0$, the cumulative evidence is identically zero, as predicted.

Table 7: **Controlled evidence-rate recovery.** Fitted slopes of cumulative evidence versus N; intervals are 95% bootstrap confidence intervals over prompts.

λ	Exact $\alpha(\lambda)$	Subtraction competitor	Multiplication competitor
0.25	1.3533	1.3428 [1.3298, 1.3554]	1.3432 [1.3298, 1.3551]
0.50	2.7067	2.7144 [2.6981, 2.7292]	2.7134 [2.6988, 2.7315]
0.75	4.0600	4.0681 [4.0533, 4.0830]	4.0679 [4.0531, 4.0847]
1.00	5.4133	5.4141 [5.4070, 5.4212]	5.4127 [5.4051, 5.4198]

Posterior and predictive concentration. Using the exact Bayesian reference predictor, increasing either N or λ sharply reduces task-posterior error. Table 8 shows that the ordering induced by the analytical evidence rate is reproduced throughout the sweep. For example, at $\lambda = 0.25$ the posterior error falls from 0.503 at $N = 1$ to 1.08×10^{-8} at $N = 64$, while larger evidence rates reach the numerical floor earlier. The corresponding predictive error has the same ordering and concentration pattern. This provides a direct finite-sample realization of the theory's exponential evidence mechanism.

Table 8: **Reference task-posterior error.** Mean posterior mass on the two competing tasks over 2,000 prompts.

λ	$N=1$	2	4	8	16	32	64
0.25	.5033	.3742	.2205	.07523	.01243	2.61×10^{-4}	1.08×10^{-8}
0.50	.3541	.1892	.04979	.003138	3.40×10^{-4}	3.90×10^{-16}	$<10^{-16}$
0.75	.1998	.05201	.005197	2.52×10^{-4}	1.18×10^{-13}	$<10^{-16}$	$<10^{-16}$
1.00	.04103	.006837	2.56×10^{-4}	2.99×10^{-13}	$<10^{-16}$	$<10^{-16}$	$<10^{-16}$

Recovery by a learned Transformer. Finally, we train five independently initialized Transformers to infer the latent task from the same demonstrations. Each model has two layers, width 64, four attention heads, GELU activations, and is trained on uniformly sampled modular-arithmetic inputs; evaluation uses 200 fresh prompts per model, λ, and N (35,000 measurements in total). As shown in Table 9, the learned models recover the same central dependence on demonstration evidence: error decreases strongly with N, and larger λ produces faster task identification. The rank correlation between the analytical rate $\alpha(\lambda)$ and negative model error is 1.0 for every $N \le 32$ and remains 0.9 at $N = 64$. At full identification ($\lambda = 1$), mean error decreases from 0.4994 to 0.0028.

Table 9: **Learned-Transformer task error.** Means over five model seeds and 200 evaluation prompts per model and condition.

λ	$N=1$	2	4	8	16	32	64
0.25	.6598	.6285	.5576	.4520	.2868	.1321	.0483
0.50	.6265	.5662	.4375	.2531	.1067	.0293	.0086
0.75	.5625	.4458	.2952	.1444	.0485	.0150	.0093
1.00	.4994	.3471	.1828	.0630	.0139	.0031	.0028

Taken together, the exact calculation, reference-predictor simulation, and learned-model experiment consistently support the paper's central setting: identifying demonstrations contribute a positive per-example evidence rate, that evidence accumulates linearly with context length, and the resulting task and predictive errors contract rapidly with N. The controlled design also shows that the advantage of identifying demonstrations in the natural-language experiments is explained by information about the latent task rather than by demonstration count alone.

G.4 Empirical Validation of Effective Instruction Sample Size

The shared-template factorization used for repeated instructions treats one instruction realization copied across several displayed positions as one task-conditional draw. By contrast, independently varying realizations may provide separate pieces of instruction evidence. We directly test whether a Transformer can learn this distinction from the cross-position correlation structure of its training distribution. The experiment independently varies the total number N of displayed instruction positions and the number K of distinct realizations occupying those positions. Each realization is copied exactly N/K times. Thus $K = 1$ is the shared-template endpoint and $K = N$ is the independent-realization endpoint.

Tasks and paired measurement. We use the three latent tasks

$$f_0(a,b) = (a+b) \bmod 17, \qquad f_1(a,b) = (a-b) \bmod 17, \qquad f_2(a,b) = ab \bmod 17. \quad (191)$$

Every demonstration is $(0,0) \mapsto 0$, which is equally compatible with all three tasks. Consequently, demonstrations provide no task-identifying information, and the model must infer the task from the instruction realizations alone. Query inputs are selected so that the three tasks produce distinct answers.

For each configuration we measure the paired task-margin shift

$$C_\theta(N,K) := m_\theta(z^{\mathrm{mismatch}}_{N,K}) - m_\theta(z^{\mathrm{anchor}}_{N,K}), \qquad m_\theta(z) := \log p_\theta(y_{\mathrm{sub}} \mid z) - \log p_\theta(y_{\mathrm{add}} \mid z). \quad (192)$$

The paired prompts have identical N, K, cluster identities, repeat counts, cluster ordering, demonstrations, query, positions, attention mask, and nuisance embedding coordinates. They differ only in the task-likelihood coordinates of the instructions: the anchor favors addition and the mismatch favors subtraction. This pairing isolates instruction-induced task pressure from sequence length and demonstration content.

Instruction realizations. Each realization has four coordinates,

$$U_{c,j} = \left[\log \frac{p(c \mid t=1)}{p(c \mid t=0)}, \log \frac{p(c \mid t=2)}{p(c \mid t=0)}, r\cos\theta_j, r\sin\theta_j\right], \tag{193}$$

where $c \in \{0,1,2\}$ is the instruction class, $j \in \{1,\dots,64\}$ is its alias identity, $r = 0.5$, and $\theta_j = 2\pi j/64$. The first two coordinates contain task evidence; the last two distinguish aliases but contain no task information. We set $p(c = t \mid t) = 0.65$ and $p(c \neq t \mid t) = 0.175$, so a matching realization has single-sample log-likelihood ratio $\log(0.65/0.175) \simeq 1.312$. All 64 aliases within a class have the same task likelihood.

Training regimes. We train under three distributions with identical single-position instruction marginals, latent-task distribution, query distribution, sequence-length distribution, architecture, optimization, and total training-token count. They differ only in cross-position correlation:

1. *Shared-template*: one realization $U \sim P(U \mid t)$ is sampled and copied to all N positions, so the independent instruction sample size is one.
2. *Hierarchical*: $K \in \{1,2,4,\dots,N\}$ is sampled, K realizations are drawn independently from $P(U \mid t)$, and each is copied N/K times. The exact instruction likelihood therefore contains K, rather than N, independent factors.
3. *Independent-occurrence*: all N displayed realizations are sampled independently, giving the endpoint $K = N$ during training.

The model is not given K, cluster labels, or cluster boundaries; it must recover the effective sample structure from equality and variation among its input embeddings.

Model and evaluation. The predictor is a two-layer pre-norm Transformer encoder with hidden width 64, four attention heads, FFN width 128, GELU activations, and no dropout. It is trained for 1,600 AdamW steps with batch size 128, learning rate 8×10^{-4}, and weight decay 10^{-4}. We train five initialization seeds and output a 17-way query-answer distribution. In the principal experiment the target is the exact latent-task posterior mapped to the three distinct query answers. A separate experiment uses sampled hard answer labels.

We evaluate every $N \in \{1,2,4,8,16,32,64\}$ and every $K \in \{1,2,4,\dots,N\}$ dividing N, using 20 held-out query seeds and 16 realization-identity pools. This yields 44,800 measurements per training regime and 134,400 in total. We fit

$$C_\theta(N,K) = a + b_N N + b_K K \tag{194}$$

and compute 95% confidence intervals using 2,000 model-seed/prompt-seed cluster-bootstrap replicates.

Results. Table 10 shows a clean separation between displayed copies and distinct realizations. For hierarchical training, the coefficient of K is strictly positive, whereas no residual dependence on N is statistically distinguishable after controlling for K. The shared-template control has no significant K coefficient. Moreover, the estimated distinct-realization response is ordered with the amount of independence in the training distribution:

$$b_K^{\text{shared}} < b_K^{\text{hierarchical}} < b_K^{\text{occurrence}}. \tag{195}$$

The hierarchical point estimate is 10.8 times the shared-control estimate, and the confidence intervals for the hierarchical and shared coefficients do not overlap. Thus the same Transformer architecture learns different effective instruction sample sizes solely from cross-position correlations.

The hard-label replication confirms that the central result is not an artifact of posterior distillation. Under hierarchical training it gives

$$b_K^{\text{hierarchical}} = 0.000187, \qquad 95\%\ \text{CI} = [0.000097,\ 0.000283], \tag{196}$$

Table 10: **Effective instruction sample size.** Coefficients in $C_\theta(N, K) = a + b_N N + b_K K$ for posterior-distilled models. Intervals are 95% cluster-bootstrap confidence intervals.

Training regime	b_N	95% CI	b_K	95% CI
Shared-template	-0.000197	$[-0.000794,\ 0.000395]$	0.000030	$[-0.000002,\ 0.000062]$
Hierarchical	0.001244	$[-0.001079,\ 0.003472]$	0.000328	$[0.000207,\ 0.000447]$
Independent-occurrence	-0.001733	$[-0.007019,\ 0.003765]$	0.001230	$[0.000960,\ 0.001518]$

while the shared-template coefficient remains statistically indistinguishable from zero, $b_K^{\text{shared}} = 0.000053$ with 95% CI $[-0.000011, 0.000116]$. The fitted hierarchical b_N is likewise not distinguishable from zero. A distance sweep further recovers the selective K response at both medium and full mismatch strength: at interpolation $s = 0.5$, $b_K = 0.000116$ with 95% CI $[0.000024, 0.000213]$, and at $s = 1$, $b_K = 0.000187$ with 95% CI $[0.000097, 0.000282]$.

These results empirically support the factorization used in the theory. When training presents repeated slots as deterministic copies of one shared draw, the learned predictor treats them as a single effective instruction sample. When training contains clustered independent realizations, task pressure grows with the number of distinct draws rather than with the number of displayed copies. The experiment therefore validates the substantive distinction between the shared-template and varying-realization settings: their different scalings can be learned by a Transformer from correlation structure alone, even when all single-position marginals and all non-instruction aspects of the training distribution are held fixed.

G.5 Prompt Sensitivity vs. Demonstration Count

We study how token-level numerical predictions respond to changes in the in-context operator pattern while keeping the underlying query answer fixed. We evaluate `Qwen3-30B-A3B` model and `Qwen3-235B-A22B` model.

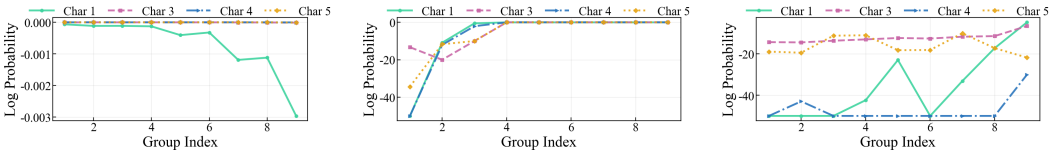
Prompt regimes. Each evaluation prompt consists of a fixed-format arithmetic query paired with a fixed target answer (e.g., `1345`). We construct three regimes that differ only in the operator pattern appearing in the demonstrations: (i) *addition-only*, where all in-context examples use +; (ii) *multiplication-only*, where all examples use $\times$; and (iii) *mixed-symbol*, where different operators are interleaved across demonstrations. Across all regimes, the query problem and its correct final answer remain identical.

Relation to theoretical support. Because the Qwen pretraining corpus and its latent task labels are unavailable, we cannot determine whether any of these operator–task pairings has literal zero conditional probability. Accordingly, this experiment tests consistent versus varying instruction mismatch, rather than directly verifying $p(\boldsymbol{i} \mid t^*) = 0$. The same fixed-cost mechanism remains relevant when $p(\boldsymbol{i} \mid t^*) > 0$ but is extremely small: a finite corpus may then contain too few such pairings for the model to learn the association reliably, making the instruction effectively unsupported for the learned model, while its fixed instruction pressure can still be overcome by accumulating demonstration evidence. It is natural to expect prompts that use a multiplication symbol while requiring addition outputs to be much rarer in pretraining than ordinary aligned uses, but we do not treat this expectation as a verified property of Qwen's training distribution.

Token-level score extraction. For each prompt, we extract token-level scores corresponding to the correct digits of the target answer. Concretely, for a target digit string $a_1 a_2 \cdots a_L$, we record, at each output position ℓ, the model-assigned log-probability of the correct digit a_ℓ. These per-digit log-probabilities serve as a fine-grained proxy for the model's confidence in the correct numerical prediction at each position. Specifically, in the figures, Char $1, 3, 4, 5$ denotes the score trace associated with the $1, 2, 3, 4$-th digit position of the fixed target answer `1345`.

Findings. Figure 2 and 3 present the token-level score trajectories for the `Qwen3-235B-A22B` and `Qwen3-30B-A3B` models, respectively, under the three operator regimes. Corresponding score gradients are shown in Figure 4 and 5. In homogeneous-symbol regimes (addition-only or multiplication-only), per-digit score trajectories are relatively smooth and exhibit small discrete

gradients, indicating stable symbolic behavior. In contrast, the mixed-symbol regime produces larger variance and more frequent sign changes in the gradients, suggesting increased instability when the operator distribution shifts within the context. These patterns are consistent across model scales, although the larger `Qwen3-235B-A22B` model exhibits systematically smaller gradient magnitudes, indicating improved robustness to symbolic heterogeneity.



(a) Equivalent Instructions (b) Consistent Instructions (c) Inconsistent Instructions

Figure 2: **Posterior confidence under instruction variation.** (`Qwen-235B-A22B-Instruct`) (a) Equivalent instructions yield stable, high-confidence predictions. (b) Incorrect but consistent instructions allow confidence to recover as more demonstrations are added. (c) Incorrect and inconsistent instructions lead to persistently low and unstable confidence despite increasing context length.

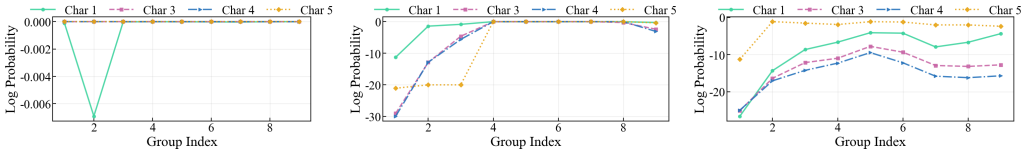


Figure 3: **Posterior confidence under instruction variation.** (`Qwen-30B-A3B-Instruct`) (a) Equivalent instructions yield stable, high-confidence predictions. (b) Incorrect but consistent instructions allow confidence to recover as more demonstrations are added. (c) Incorrect and inconsistent instructions lead to persistently low and unstable confidence despite increasing context length.

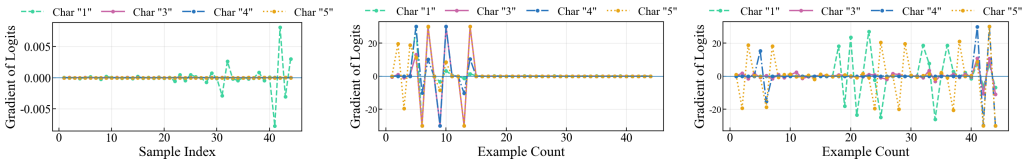


Figure 4: **Discrete gradients of log probability.** (`Qwen-235B-A22B-Instruct`) (a) Equivalent instructions yield small and stable gradients. (b) Incorrect but consistent instructions show gradual stabilization. (c) Incorrect and inconsistent instructions produce large and irregular gradients.

G.6 Controlled Agent Execution through Explicit Intermediate States

We study a fixed-plan arithmetic agent implemented with `Llama3.2-3B`. At execution step k, the agent reads its current state $x^{(k)}$, applies the operation prescribed by the plan, and emits the updated state $x^{(k+1)}$. Because the language model serializes these states and updates as an autoregressive text trace, the implementation uses chain-of-thought (CoT)-style prompting. Here CoT is not a separate theoretical object: it is the textual interface through which the agent exposes its state trajectory and executes one transition at a time. This correspondence makes intermediate states and step-level errors observable and lets us test the agent-execution mechanism in Section 4.1 and Theorem 4.2. We compare two agent plans for the same multi-digit addition objective. They have the same correct terminal state but differ in whether their component operations are supported by training.

In-distribution agent plan. In the *in-distribution* plan, every state transition invokes a well-defined arithmetic operation, such as digit-wise decomposition, carry propagation, or aggregation. To ensure

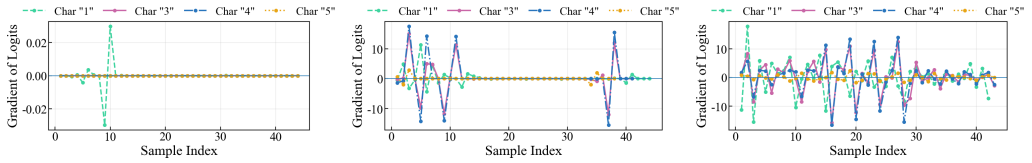


Figure 5: **Discrete gradients of log probability.** (`Qwen-30B-A3B-Instruct`) (a) Equivalent instructions yield small and stable gradients. (b) Incorrect but consistent instructions show gradual stabilization.(c) Incorrect and inconsistent instructions produce large and irregular gradients.

that the agent can execute these operations in isolation, we generate 10,000 training examples for *each* operation and fine-tune the model on the resulting step-level datasets before evaluation. Each example is formatted as an instruction–input–output triple, analogous to the pretraining setup in Appendix G.1.

Step 1: $1234561 = 1234560 + 1$
Step 2: $1323212 = 1323210 + 2$
Step 3: $1234560 + 1323210 \rightarrow (123456 + 132321) \times 10$
Step 4: $1 + 2$
Step 5: Combine results.

Each step corresponds to a sub-computation the model can reliably perform in isolation.

Out-of-distribution agent plan. The *out-of-distribution* plan reaches the same correct terminal state, but it requires intermediate operations on which the model has *not* been explicitly trained. No additional fine-tuning is performed for these operations; the agent must execute the prescribed state transitions from the plan supplied in the prompt.

Step 1: $1234561 = 1000000 + 234561$
Step 2: $1323212 = 1000000 + 323212$
Step 3: Add corresponding components. $1000000 + 1000000 = 2000000, 234561 + 323212 = 557773$
Step 4: Combine results.

Although the plan is algebraically correct, its intermediate transitions do not correspond to well-supported arithmetic routines in the model's training distribution and are therefore unreliable during execution.

Training Setup. All fine-tuning for the in-distribution agent plan uses the same LoRA configuration, optimizer, and training hyperparameters as in Appendix G.1. In particular, the architecture, learning rate, number of epochs, batch size, and quantization settings are kept fixed. This isolates the effect of step-level intrinsic capability from other confounding factors.

Evaluation. We evaluate the step-wise log probabilities to make the empirical measurement consistent with the pathwise perspective in Section 4.2. Let $f_s(\boldsymbol{z})$ denote the model logits at execution step s, and let $\ell_s(\boldsymbol{z}) = g(f_s(\boldsymbol{z}))$ be the corresponding cross-entropy loss on the correct step output. If f_s is $L_{f,s}$-Lipschitz and the cross-entropy map g is L_g-Lipschitz on the bounded logit region considered in the experiment, then for two prompts $\boldsymbol{z}_1, \boldsymbol{z}_2$ along the same step-wise path,

$$|\ell_s(\boldsymbol{z}_1) - \ell_s(\boldsymbol{z}_2)| \leq L_g \|f_s(\boldsymbol{z}_1) - f_s(\boldsymbol{z}_2)\| \leq L_g L_{f,s} \|\boldsymbol{z}_1 - \boldsymbol{z}_2\|.$$

Thus, if prompts are varied along a fixed step-wise path, gradients of the correct-output log probability provide a proxy for the effective Lipschitz sensitivity of that transition. Our plots instead compare different execution steps, which generally implement different functions; their variation is therefore a qualitative stability diagnostic rather than a direct estimate of a within-step Lipschitz constant. Under the in-distribution agent plan, each transition invokes an operation that is better represented during training, so the log-probability profile is expected to vary more smoothly across steps. In contrast, the out-of-distribution plan retains difficult operations in the execution trajectory, leading to sharper step-level variations and greater effective step-wise sensitivity.

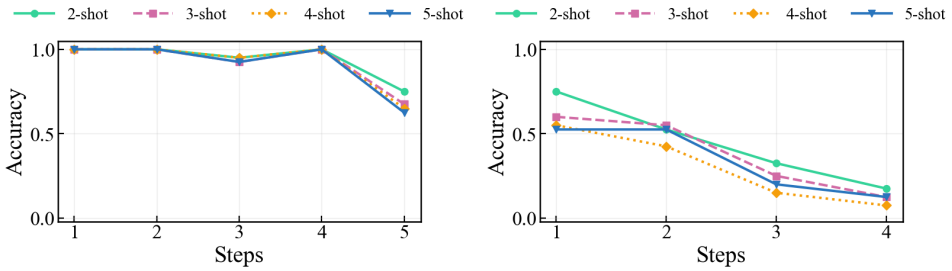


Figure 6: Agent transition accuracy. Left: ID plan; right: OOD plan.

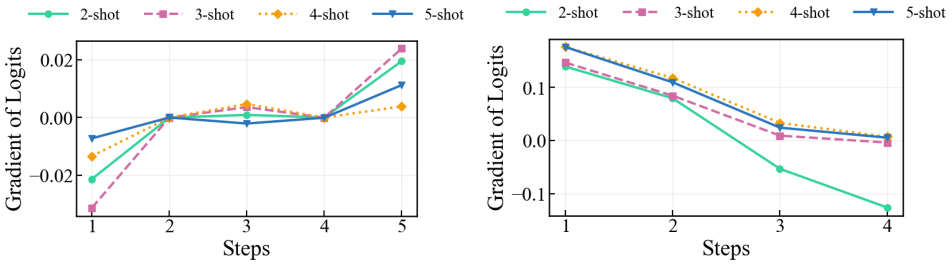


Figure 7: Step-wise agent log-probability gradients. Left: ID plan; right: OOD plan.

Results. We report three types of measurements: (i) final accuracy, (ii) step-wise accuracy, and (iii) step-wise log-probabilities, which serve as a proxy for the effective Lipschitz behavior at each step.

Final accuracy. Consistent with Figure 1c, the in-distribution (ID) agent plan substantially improves performance over direct prediction, whereas the out-of-distribution (OOD) plan degrades performance and is often worse than executing no plan. This matches Theorem 4.2: an agent can realize an unsupported target when its planned operations are supported and identifiable, whereas one poorly identified or poorly approximated transition can dominate the execution trajectory.

Step-wise accuracy. To further understand this behavior, we measure the correctness of every state transition in the agent trajectory. Under the ID plan, early transitions are executed almost perfectly and most errors occur only at final aggregation. Under the OOD plan, accuracy degrades progressively, with each subsequent state transition less reliable than the previous one.

This progressive degradation is consistent with the error-propagation structure in Theorem 4.2, whose final-error bound sums the identification and model-transport errors of the individual steps. When intermediate steps are unreliable, these contributions accumulate and lead to poor final performance.

Step-wise Lipschitz behavior (via log-probabilities). To connect these observations with the pathwise theory, we examine the average log-probability assigned to the correct token at each step. As explained above, the cross-step profile diagnoses stability along the realized agent trajectory; it does not by itself estimate a within-step pathwise Lipschitz constant.

In the ID setting, the log-probabilities remain stable across steps, providing evidence that no individual supported subtask introduces a pronounced loss spike. This is consistent with the favorable regime of Theorem 4.2, in which every step has a controlled identification and model-transport error.

In contrast, in the OOD setting, step-wise accuracy decreases substantially and the log-probability profile exhibits larger fluctuations. This indicates that some intermediate computations are unreliable, exactly the failure mode captured by the step-wise sum in Theorem 4.2. When a substep is poorly identified or has a large model-transport error, its contribution to the sum can dominate the final result. The experiments are therefore consistent with the theory's qualitative prediction that an unreliable step can degrade the final answer; isolating the two error sources individually would require controlled within-step interventions.

Table 11: **Controlled validation of supported agent composition.** Probabilistic error under direct prediction and stochastic agent rollout, together with the sum of the supported-agent conditional transition errors. Values are mean ± standard deviation over five models and twelve prompt seeds; every measurement includes all 17 query states.

N	Direct error	Supported-agent rollout	OOD-agent rollout	$\sum_{k=0}^{2} q_k(N)$
1	0.964 ± 0.023	0.456 ± 0.302	0.877 ± 0.103	0.531 ± 0.385
2	0.973 ± 0.020	0.185 ± 0.170	0.865 ± 0.098	0.194 ± 0.178
4	0.974 ± 0.021	0.072 ± 0.091	0.877 ± 0.113	0.074 ± 0.093
8	0.966 ± 0.022	0.038 ± 0.048	0.900 ± 0.078	0.038 ± 0.048
16	0.963 ± 0.023	0.016 ± 0.018	0.904 ± 0.054	0.016 ± 0.018
32	0.966 ± 0.022	0.0088 ± 0.0035	0.892 ± 0.061	0.0088 ± 0.0035
64	0.965 ± 0.021	0.0078 ± 0.0036	0.898 ± 0.042	0.0079 ± 0.0036

Controlled validation of supported agent composition. We additionally conduct a controlled finite-task experiment that tests the central mechanism of Theorem 4.2 directly. The agent follows an explicit three-operation plan, and its intermediate states are fully observable, so its conditional transition errors can be compared directly with the theorem's agent-execution bound. All computations are performed modulo 17. The training distribution contains only the three supported tasks

$$f_0(x) = x + 1, \qquad f_1(x) = 2x, \qquad f_2(x) = x - 3, \tag{197}$$

whereas the composed target

$$f^{\dagger}(x) = 2x - 1 = f_2 \circ f_1 \circ f_0(x) \pmod{17} \tag{198}$$

is never included as a direct training task. Demonstrations contain 8% observation noise, so competing-task likelihood ratios have common support; the query and task outputs remain deterministic, avoiding an artificial final error floor.

We compare three execution procedures. *Direct ICL* asks for $f^{\dagger}(x)$ as one indivisible mapping. The *supported agent* follows the three-operation plan

$$x \xrightarrow{x+1} x^{(1)} \xrightarrow{2x} x^{(2)} \xrightarrow{x-3} y. \tag{199}$$

The equal-length *OOD agent* instead follows

$$x \xrightarrow{x+2} x^{(1)} \xrightarrow{2x-2} x^{(2)} \xrightarrow{x-3} y, \tag{200}$$

whose first two subtasks never occur during training, although their composition still gives the same correct target $2x - 1$. This equal-length control separates support of the planned operations from the mere presence or length of an explicit execution trajectory.

We train five independently initialized models and use 12 prompt seeds. For each model–prompt pair, we evaluate all 17 query states at $N \in \{1, 2, 4, 8, 16, 32, 64\}$. Each reported configuration therefore averages over 60 model–prompt measurements, with all query states included in each measurement. The complete evaluation contains 840 aggregate records.

The contrast is sharp. The supported-agent rollout error decreases from 0.456 to 0.0078, a factor of approximately 58, and its greedy rollout error is zero for $N \geq 32$. Direct ICL remains near 0.96 error, consistent with the fact that the held-out composed task cannot be selected directly from the training-supported task set. The equal-length OOD agent remains between 0.865 and 0.904 error. Thus the improvement cannot be explained merely by emitting intermediate states or by plan length: it appears precisely when the plan consists of supported, identifiable operations, as required by Theorem 4.2.

Step-wise error propagation. For the supported agent plan, let F_{k-1} be the event that all preceding transitions are correct and define the conditional step-error probability

$$q_k(N) := P_M(E_k^c \mid F_{k-1}). \tag{201}$$

We compute these probabilities under the correct prefix and evaluate the exact stochastic autoregressive rollout distribution, rather than relying only on greedy decoding. The recursion used in the proof

of Theorem 4.2 gives

$$e_{\text{rollout}}(N) \le \sum_{k=0}^{2} q_k(N). \tag{202}$$

All 840 evaluated records satisfy this inequality. This checks error accumulation for the rollout kernel actually evaluated; the inequality itself does not require exact compatibility with an extracted local prompt. The bound is non-vacuous and becomes nearly tight once the individual step errors are small. Using unrounded means,

$$\begin{aligned} N = 4: &\quad 0.0725 \le 0.0736, \\ N = 16: &\quad 0.0161 \le 0.0162, \\ N = 64: &\quad 0.00783 \le 0.00785. \end{aligned} \tag{203}$$

This tightness is expected in the low-error regime because simultaneous-error corrections are higher order in the q_k. The experiment therefore shows that the theorem's step-wise error decomposition is not merely a loose worst-case accounting device in this controlled setting: the measured conditional errors closely explain the final rollout error. It does not separately establish that extracted-prompt error bounds transfer to a full-context kernel without a correction. That requires matched evaluations of the two kernels in (143) or (144). For an executor that explicitly invokes the extracted step prompts with no additional retained context, the zero-discrepancy guarantee follows instead from Corollary E.5.

Actual-model tracking of the reference predictor. Finally, we evaluate the discrepancy between the Transformer and the Bayesian reference predictor on the supported steps. The maximum step-wise model–reference gap decreases from 0.582 at $N = 1$ to 0.027 at $N = 64$. Hence the observed concentration is not confined to the idealized Bayesian reference: as demonstration evidence accumulates, the actual Transformer also tracks that reference increasingly closely. The theorem keeps this discrepancy explicit through the terms $\varepsilon_M^{(k)}(N)$ and does not require them to vanish; their empirical decrease is an additional favorable property of the learned model in this experiment.

G.7 Continuous Soft-Prompt Paths in a Pretrained Qwen3 Model

We finally test the pathwise perspective of Section 4.2 directly in a pretrained open-weight language model. Unlike the preceding discrete prompt comparisons, this experiment bypasses tokenization after constructing the path: interpolated instruction embeddings are injected into `Qwen3-4B-Instruct-2507` through `inputs_embeds`. The model is evaluated in BF16 without quantization or parameter updates.

Task and paths. The task is addition modulo 17. Each prompt contains one instruction, N addition demonstrations, and one query, with $N \in \{1, 2, 4, 8, 16, 32\}$. We use eight independently sampled prompts for each N and choose every query so that addition, subtraction, and multiplication give distinct answers. We pre-specify two paths. The first interpolates from the supported instruction "Perform addition modulo 17." to the mismatched instruction "Perform subtraction modulo 17." The second interpolates from the same supported anchor to a synthetic soft-OOD endpoint whose token-wise embedding norms are matched to those of the anchor. For both paths,

$$E_s = (1 - s)E(\boldsymbol{i}_0) + sE(\widetilde{\boldsymbol{i}}), \qquad s \in \{0, .05, \dots, 1\}. \tag{204}$$

The resulting evaluation contains 2,016 path points.

Both symbolic endpoint instructions occupy the same seven token slots and differ only in the task word. We first obtain the ordinary input embeddings from the model embedding layer, replace only these seven instruction slots by E_s, and pass the resulting tensor directly to the model. Sequence length, attention mask, position IDs, demonstrations, query, and every embedding outside the instruction span remain fixed along a path. At $s = 0$, the maximum logit difference between the `inputs_embeds` computation and the ordinary `input_ids` computation is exactly zero; the same check at $s = 1$ for the symbolic path also gives zero. No evaluated prompt produces a NaN or infinite value. These checks verify that the experiment follows a genuine continuous path through the embedding space rather than a sequence of retokenized discrete prompts.

Table 12: **Empirical sensitivity along genuine Qwen3 soft-prompt paths.** Values average the finite-difference sensitivities within each path region and then across the six demonstration counts. Both paths show a substantially smoother neighborhood around the supported anchor than the intermediate task-transition region.

Path	Region	$\widehat{L}^{\log p}$	$\widehat{L}^{\mathrm{err}}$	$\widehat{L}^{\mathrm{sub}}$
Symbolic mismatch	Anchor $[0, .25]$	3.668	0.166	3.708
	Middle $[.25, .75]$	20.674	0.911	26.458
	Endpoint $[.75, 1]$	7.518	0.076	7.431
Synthetic soft-OOD	Anchor $[0, .25]$	1.221	0.053	1.198
	Middle $[.25, .75]$	6.805	0.249	6.143
	Endpoint $[.75, 1]$	3.447	0.112	2.962

Output probabilities and empirical path sensitivity. We compute the full teacher-forced continuation log-probability $\ell_t(s, N)$ for the addition, subtraction, and multiplication answers, including all answer tokens, and define the candidate-normalized probability

$$q_t(s, N) := \frac{\exp(\ell_t(s, N))}{\sum_{t' \in \{\mathrm{add,sub,mul}\}} \exp(\ell_{t'}(s, N))}. \tag{205}$$

For adjacent path points, we measure the path-restricted finite-difference sensitivity of the correct-output log-probability by

$$\widehat{L}_j^{\log p}(N) := \frac{\left|\log M(y^* \mid \boldsymbol{z}_N(E_{s_{j+1}})) - \log M(y^* \mid \boldsymbol{z}_N(E_{s_j}))\right|}{\|E_{s_{j+1}} - E_{s_j}\|_F}. \tag{206}$$

We analogously compute $\widehat{L}_j^{\mathrm{err}}$ for $1 - q_{\mathrm{add}}$ and $\widehat{L}_j^{\mathrm{sub}}$ for the subtraction log-odds $\ell_{\mathrm{sub}} - \ell_{\mathrm{add}}$. These are empirical sensitivities restricted to the specified paths, precisely matching the local quantity suggested by the pathwise analysis rather than a global Lipschitz constant.

A locally smooth supported-instruction neighborhood. The two independently specified paths exhibit the same geometric structure. On the symbolic mismatch path, the middle-to-anchor sensitivity ratios are 5.64, 5.50, and 7.14 for the correct-output log-probability, candidate error, and subtraction log-odds, respectively. On the synthetic soft-OOD path, the corresponding ratios are 5.58, 4.66, and 5.13. Thus, across both a semantically meaningful competing endpoint and a synthetic endpoint, Qwen3 has a comparatively smooth neighborhood around the supported addition instruction, separated from the alternative endpoint regime by a much more sensitive transition region.

This experiment supplies a direct model-level realization of the soft-prompt geometry used in the theory. The path is formed inside the actual embedding space of a pretrained language model, and the measured output variations show that the pathwise sensitivity distinguishes a stable supported neighborhood from the transition to alternative instruction interpretations. It therefore complements the discrete Qwen3 prompt experiments by exposing the continuous local structure that those comparisons cannot observe.

H LIMITATIONS

Our results are conditional characterizations rather than unconditional claims about every pretrained language model. The latent-task framework assumes a finite task family, task-conditional generation, and the format-specific factorizations used in the proofs. In particular, repeated copies of one instruction are modeled as a single shared template, whereas distinct corruptions are treated as separate instruction slots. These choices isolate the effect of prompt structure, but a real pretraining distribution need not admit this exact decomposition. Task identifiability and the integrability conditions used for posterior concentration are likewise sufficient conditions and need not hold for ambiguous or overlapping tasks.

The treatment of unsupported instructions requires a distinction between what is identified by data and what is selected by the model. Population cross-entropy identifies the Bayes predictor only on

the pretraining support and does not determine a latent-task decomposition or an off-support value. We therefore impose supported calibration and use a Transformer-induced scoring map to define the reference instruction log-odds away from symbolic support, as described in Appendix C. Analytic Transformer components guarantee the existence of the finite continuation chain along a fixed compact soft-prompt path, but they do not imply that the selected extension is semantically correct or close to an unknown data-generating law off support. Moreover, nearby complex singularities can make the continuation radii small and the constants M_ℓ and Ω large, even when the chain exists. Appendix D.1.1 provides a quantitative local model-transport certificate under supported-anchor calibration and length-independent architectural stability. Its constants may be too large to certify an experimentally useful radius. Small average pretraining loss alone does not imply worst-case anchor accuracy: the loss-to-calibration result requires coverage, and its finite-family version does not extend to all prompt lengths. Fixed-context models are covered only over admissible lengths, not by an infinite-context asymptotic claim.

The condition $p(i \mid t^*) = 0$ is a statement about the latent pretraining data-generating distribution, rather than a quantity that can generally be verified for a real LLM. Complete pretraining corpora and their sampling mechanisms are usually unavailable. Even access to a finite corpus would show only whether a particular instruction–task pairing occurs in that realization, not whether its population probability is exactly zero. Consequently, experiments with real pretrained models can only use reasonable proxies—such as the provenance, novelty, or apparent absence of a pairing— to motivate that an instruction is unsupported or extremely rare for the target task. Such evidence does not certify the literal zero-support condition, so empirical "unsupported" instructions should be interpreted as plausible analogues of the theoretical regime.