Reinforcement Learning Improves Traversal of Parametric Knowledge in LLMs

Renfei Zhang[1], **Manasa Kaniselvan**[2*], **Rylan Schaeffer**[3], **Niloofar Mireshghallah**[1,*]
[1]Carnegie Mellon University, [2]MIT, [3]Stanford University
Correspondence: {az5, nmireshg}@andrew.cmu.edu

arXiv:2511.05933v3 [cs.CL] 4 Sep 2026

Abstract

Reinforcement learning (RL) is often credited with improving reasoning at the expense of factual knowledge. We instead find that reasoning models outperform their instruction-tuned versions on factual recall by accessing existing parametric knowledge more effectively. Across five model families, structured prompting—which explicitly guides models through hierarchical traversal—recovers most of this gap, suggesting that much of the missing knowledge is latent rather than absent. Controlled RL experiments further support this: training on unseen, non-extractable facts improves recall of held-out, frequent but previously inaccessible facts, ruling out simple data exposure. Decomposing the training objective further attributes this gain to iterated on-policy exploration. The same mechanism appears behaviorally and internally: the reasoning advantage grows with retrieval depth, while layerwise analysis finds similar factual representations but divergent query representations. Distilled models, in contrast, often imitate self-correction without acquiring the exploration needed for navigation. Together, these findings suggest that improving factual recall in LLMs depends not only on expanding what models know but also on teaching them to navigate it—motivating future post-training methods that optimize traversal.

1 Introduction

Large Language Models (LLMs) acquire vast parametric knowledge during pretraining, encoding facts and concepts across model parameters. Post-training techniques, including supervised fine-tuning (SFT), distillation, Reinforcement Learning from Human Feedback (RLHF) and RL, then transform these base models into instruction-following agents capable of complex reasoning (Bai et al., 2022; Wang et al., 2025b; Yu et al., 2026). While these methods improve reasoning and align with user preferences, a growing body of evidence reveals a trade-off known as the "alignment tax" (Askell et al., 2021; Lin et al., 2024; Sorensen et al., 2026): models sacrifice factual recall to optimize for other objectives, leading to degraded performance on knowledge-intensive benchmarks (Gekhman et al., 2024; Yuan et al., 2024).

However, existing work has examined a limited number of recall tasks and post-training methods. This leaves open two broader questions: ***do these degradation patterns generalize across the major forms of knowledge retrieval—entity-relations, taxonomies, and ontologies? And how do knowledge-recall abilities differ across distilled, instruction-tuned, and reasoning models*** [1]***?***

To address these questions, we study knowledge retrieval tasks which require models to navigate both explicit and implicit structures in parametric memory. Surprisingly, reasoning models can outperform their instruct counterparts by 13 and 24 percentage points on PopQA and MedConceptsQA respectively (Mallen et al., 2023; Shoham and Rappoport, 2024), directly challenging the conventional wisdom that RL sacrifices memorization for reasoning (Ghosh et al., 2024; Chu et al., 2025). We hypothesize these models succeed through systematic hierarchical navigation rather than direct recall. For example, in structured domains, such as medical code lookup, retrieval may require traversing a formal hierarchy (Figure 1): to identify that ICD-9-CM code 57.95 refers to "Replacement of indwelling urinary catheter," reasoning models systematically traverse the knowledge hierarchy (Chapter 11 → codes 57.0-57.99 → specific procedure). In contrast, instruct models often attempt direct recall, which often fails due to the

*Work partly done at Meta.

[1]Throughout, we use *instruct* and *reasoning* to denote publicly released model variants, whose recipes may combine multiple post-training stages. Comparisons between these variants are therefore observational and do not isolate the effect of RL. We reserve *SFT* and *RL* for the controlled knowledge-injection experiments described in Section 3.

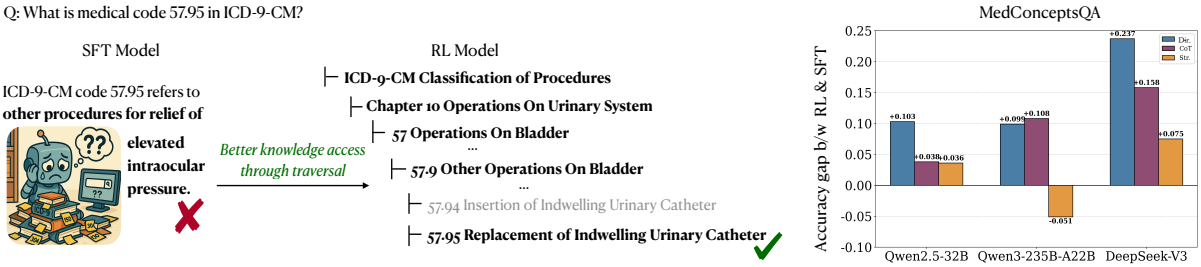


Figure 1: **RL improves LLM's parametric knowledge access through hierarchical traversal. (Left)** Example query "What is medical code 57.95 in ICD-9-CM?" The instruct model retrieves an incorrect description via direct memorization, while the reasoning model systematically traverses the knowledge hierarchy to arrive at the correct answer. Final answers and intermediate codes visited during reasoning are in bold. **(Right)** Majority-voted accuracy on MedConceptsQA with three prompt strategies: Dir. (direct answering), CoT (chain-of-thought), and Str. (structured hierarchy traversal). When instruct models are explicitly instructed to traverse the knowledge hierarchy before answering, their accuracy gap relative to reasoning models quickly narrows, sometimes even reverses. This indicates that instruct models possess the parametric knowledge but lack effective access methods to surface it.

vast code space. In flat factual domains, such as entity-relation QA, there is no explicit taxonomy, but reasoning models still benefit from implicit navigation by recalling related entities, relations, or contexts that lead to the answer. Therefore, we propose that **RL improves navigation of existing parametric knowledge rather than adding new factual content**.[2]

To disentangle knowledge acquisition from navigation, we design a series of experiments and uncover four findings:

- **Instruct models already possess the knowledge but lack the navigation policy to access it.** Inspired by work showing prompt optimization can match RL gains (Khattab et al., 2024; Agrawal et al., 2026; Ziems et al., 2025), we develop structured prompting that explicitly guides models through hierarchical traversal; it reduces the 24pp gap between DeepSeek-V3 and DeepSeek-R1 to 7pp on MedConceptsQA (Figure 1, right). Together with manual inspection of instruct model responses, this result suggests that information is present but only reliably accessible with proper navigation. Response-distribution analysis corroborates this: hierarchical navigation pushes the knowledge distribution from mostly incorrect to mostly correct, while reasoning models—having internalized the navigation policy—change little because they already operate near their performance ceiling. Applying RL to Qwen3-8B-/30B-A3B-Instruct on unseen facts further improves recall on held-out facts that are frequent but previously non-extractable, isolating RL's effect on access to latent parametric knowledge.
- **The RL gain on knowledge access mostly comes from iterated on-policy exploration.** Holding the data and hyperparameters fixed, we compare six post-training objectives spanning supervised finetuning, rejection-sampling, and group relative policy optimization (GRPO). Knowledge access improves when self-generated rollouts track the current policy, and the contrastive advantage of GRPO adds little once iteration is present.
- **Reasoning models' traversal advantage grows with retrieval depth.** We introduce two depth-stratified datasets, IPC-CAR and DBO-CAR, that require recalling the paths from a given pair of nodes to their nearest common ancestor. Using a path-matching score that measures traversal fidelity, we find the reasoning–instruct gap widens as depth increases—indicating that reasoning's advantage emerges specifically under greater retrieval complexity, a signature of learned traversal that scales with task complexity.
- **Post-training reshapes how models process questions while leaving factual representations intact.** To provide internal validation, we conduct layerwise representational analysis (Agarwal et al., 2024; Mukherjee et al., 2025; Skean et al., 2025; Huan et al., 2025) on matched query–answer pairs, com-

[2] *Knowledge access* (or *retrieval*) denotes eliciting facts in model parameters, whereas *hierarchical traversal* (or *navigation*) refers to accessing them through intermediate levels of knowledge hierarchy; Table 3 provides the complete mapping.

Table 1: Majority vote accuracy comparison of Instruct vs. Reasoning models on PopQA, IPC, and MedConceptsQA datasets across three prompt templates (QA, CoT, Structured). Δ rows show the reasoning advantage; orange shading intensity reflects gap magnitude. Bold indicates best within each model pair.

Model	Type	PopQA			IPC			MedConceptsQA		
		QA	CoT	Struct.	QA	CoT	Struct.	QA	CoT	Struct.
Qwen2.5-32B	Instruct	0.252	0.288	0.310	0.300	**0.370**	**0.400**	0.379	0.475	0.469
	Reasoning	**0.386**	**0.380**	**0.375**	**0.350**	0.360	0.360	**0.482**	**0.513**	**0.505**
	Δ	**+0.134**	**+0.092**	**+0.065**	**+0.050**	**-0.010**	**-0.040**	**+0.103**	**+0.038**	**+0.036**
Qwen3-8B	Instruct	0.205	0.252	**0.262**	0.266	0.277	0.372	0.338	0.337	0.406
	Reasoning	**0.237**	**0.267**	**0.262**	**0.298**	**0.394**	**0.394**	**0.400**	**0.424**	**0.421**
	Δ	**+0.032**	**+0.015**	**+0.000**	**+0.032**	**+0.117**	**+0.022**	**+0.062**	**+0.087**	**+0.015**
Qwen3-30B-A3B	Instruct	0.231	0.279	0.302	0.309	0.319	0.436	0.398	0.404	**0.489**
	Reasoning	**0.288**	**0.317**	**0.312**	**0.426**	**0.372**	**0.457**	**0.498**	**0.506**	0.476
	Δ	**+0.058**	**+0.037**	**+0.010**	**+0.117**	**+0.053**	**+0.021**	**+0.100**	**+0.102**	**-0.013**
Qwen3-235B-A22B	Instruct	0.429	0.491	0.541	0.372	0.383	**0.596**	0.542	0.548	**0.631**
	Reasoning	**0.494**	**0.562**	**0.567**	**0.457**	**0.404**	0.574	**0.641**	**0.656**	0.580
	Δ	**+0.065**	**+0.071**	**+0.026**	**+0.085**	**+0.021**	**-0.022**	**+0.099**	**+0.108**	**-0.051**
DeepSeek-V3	Instruct	0.430	0.505	**0.569**	0.415	0.596	0.553	0.541	0.632	0.717
	Reasoning	**0.513**	**0.557**	**0.569**	**0.638**	**0.670**	**0.574**	**0.778**	**0.790**	**0.792**
	Δ	**+0.083**	**+0.052**	**+0.000**	**+0.223**	**+0.074**	**+0.021**	**+0.237**	**+0.158**	**+0.075**
DeepSeek-V3.1	Instruct	0.429	0.506	0.565	0.468	0.500	**0.606**	0.526	0.595	**0.785**
	Reasoning	**0.429**	**0.508**	**0.567**	**0.479**	**0.511**	0.585	**0.794**	**0.805**	0.774
	Δ	**+0.003**	**+0.002**	**+0.002**	**+0.011**	**+0.011**	**-0.021**	**+0.268**	**+0.210**	**-0.011**
Kimi-K2	Instruct	0.507	0.533	0.566	0.511	0.521	0.543	0.595	0.634	0.750
	Reasoning	**0.580**	**0.662**	**0.652**	**0.575**	**0.585**	**0.564**	**0.834**	**0.829**	**0.830**
	Δ	**+0.073**	**+0.129**	**+0.086**	**+0.064**	**+0.064**	**+0.021**	**+0.239**	**+0.195**	**+0.080**
Kimi-K2.5	Instruct	0.507	0.573	0.649	0.521	0.500	**0.553**	0.699	0.734	0.875
	Reasoning	**0.597**	**0.700**	**0.682**	**0.574**	**0.574**	**0.585**	**0.905**	**0.915**	**0.911**
	Δ	**+0.100**	**+0.127**	**+0.033**	**+0.053**	**+0.074**	**+0.032**	**+0.206**	**+0.181**	**+0.036**

paring queries (e.g., "What is medical code 57.95?") against declarative statements (e.g., "Code 57.95 refers to urinary catheter replacement"). We find that declarative statements maintain high cosine similarity between base and reasoning models, while queries diverge. This asymmetry shows that post-training techniques such as RL primarily transform how models process questions while leaving factual knowledge representations intact (Figure 6).

- **Distillation lacks the exploration behavior needed to navigate latent knowledge.** Contrary to prior findings that distillation can outperform RL on reasoning tasks (Guo et al., 2025; Yue et al., 2025), distilled models often underperform reasoning models on knowledge recall, sometimes falling below base and instruct models. Manual inspection of model outputs shows that they imitate self-correction without truly abandoning incorrect hypotheses, lacking the trial-and-error exploration RL instills. This prevents them from navigating the hierarchy to surface latent knowledge.

In summary, our experiments show that reasoning models excel at knowledge-retrieval tasks because they learn hierarchical navigation strategies, rather than simply acquiring new knowledge. Future methods should focus not only on injecting or distilling knowledge, but also on learning how to better retrieve and navigate the knowledge already encoded in model parameters.

2 Structured Prompting Reveals Latent Knowledge in Instruct Models

We evaluate whether structured prompting improves access to parametric knowledge across two settings: flat factual associations, where useful intermediate cues are implicit, and formally organized code systems, where the path to an answer is explicit. These settings allow us to test whether the benefits of navigation generalize across different organizations of knowledge. Across direct, chain-of-thought (CoT), and structured prompts, structured prompting shrinks the instruct–reasoning accuracy gap, showing the effectiveness of hierarchical traversal at re-surfacing the knowledge. Response distribution and reasoning trace analyses corroborate that instruct models possess the underlying knowledge but lack the navigation policy that RL internalizes.

2.1 Experimental Setup

Models. To make our conclusions robust, we evaluate eight instruct–reasoning model pairs spanning five families: Qwen2.5-32B (Yang et al., 2024), Qwen3 (8B/30B-A3B/235B-A22B) (Yang et al., 2025), DeepSeek-V3/V3.1 (DeepSeek-AI et al., 2025; Guo et al., 2025), and Kimi-K2/K2.5 (Team et al., 2026, 2025), covering both dense and mixture-of-experts (MoE) architectures.

Datasets. Our evaluation focuses on factual knowledge recall rather than logical reasoning. The benchmarks span relational (PopQA), taxonomic (International Patent Classification (IPC)-Lookup) and ontological (MedConceptsQA) knowledge, with both multiple-choice and open-ended formats. Specifically, PopQA organize facts as entity-relation pairs, while IPC-Lookup and MedConceptsQA encode facts within explicit hierarchies.

- **PopQA** (Mallen et al., 2023): an open-ended, entity-centric QA dataset for evaluating long-tail factual recall, without explicit hierarchical structure.
- **IPC-Lookup:** a multiple-choice dataset of 94 questions requiring models to identify the correct description of a given patent code, drawn from the most recent IPC taxonomy, Version 2026.01, released by World Intellectual Property Organization (2026). [3] Each question presents a patent code alongside four candidate descriptions. Further details are in Appendix B.4.1.
- **MedConceptsQA** (Shoham and Rappoport, 2024): a multiple-choice benchmark testing recall of code descriptions for clinical ontologies including diagnoses (ICD9/10-CM), procedures (ICD9/10-PROC), and drugs (ATC).

Prompting. We use three prompt templates:

- **Direct QA** asks the model to output only the final answer, without explanation.
- **CoT** asks the model to provide a final answer with a free-form explanation, without imposing any procedural constraints.
- **Structured Prompting** asks the model to traverse the relevant hierarchy of facts or codes before giving the final answer. This tests whether explicit structured recall can reduce the performance gap between instruct and reasoning models (see Appendix B.2 for detail).

[3] We use Version 2026.01 and select codes intended to be stable across recent releases (2024.01–2026.01), minimizing any confound from models trained before the latest revision.

Evaluation. We sample from all models using a temperature of 0.8, a top-p of 0.7, and a maximum of 10,000 tokens across three independent runs. Performance is reported as both majority-voted and mean accuracy (± standard deviation), where majority voting selects the most frequent answer per question. Here we prioritize majority-voted accuracy over pass@k because our goal is to measure reliable knowledge access rather than knowledge boundary. We describe the full protocol in Appendix B.3.

2.2 Structured Prompting Narrows the Gap Between Instruct and Reasoning Models on Knowledge Retrieval

Structured prompting explicitly instructs models to traverse unstructured factual data through relevant entities and structured codes through neighbors and predecessors. This substantially reduces the accuracy gap between instruct and reasoning models on knowledge retrieval tasks (Table 1). On MedConceptsQA, the DeepSeek-V3–DeepSeek-R1 gap shrinks from +23.7 pp under direct QA to +7.5 pp with structured prompting. PopQA shows similar pattern: the gap decreases from +12.7 pp to +3.3 pp for Kimi-K2.5, and from +13.4 pp to +6.5 pp for Qwen2.5-32B. These results show that hierarchical navigation helps instruct models better access the latent parametric knowledge. We further analyze the implications of this finding in Section 2.3.

2.3 Instruct Models Already Have Parametric Knowledge But Needs Better Access

Response Distribution Analysis. Section 2.2 shows that providing instruct models with traversal algorithms can narrow the performance gap between instruct and reasoning models. To further investigate this, we examine changes in correctness distribution. Figure 2 reports majority-voting results for Kimi-K2.5-Instruct and Kimi-K2.5-Think over three independent runs on MedConceptsQA, categorized into four classes: *All Incorrect* (0/3), *Majority Incorrect* (1/3), *Majority Correct* (2/3), and *All Correct* (3/3). Switching from direct QA to structured prompting shows marked redistribution in Kimi-K2.5-Instruct where *All Incorrect* drops from 29 % to 6 % and *All Correct* increases from 60% to 78%. Kimi-K2.5-Think shows no significant shift (e.g., *All Incorrect* stays at 4%), suggesting it already operates near its performance ceiling. This suggests that the accuracy gains come not from acquiring new knowledge, but from effec-

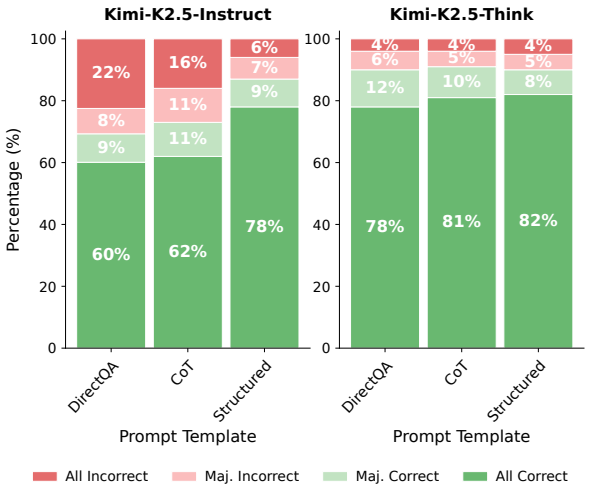


Figure 2: Comparative distribution analysis of Kimi-K2.5-Instruct and Kimi-K2.5-Think on MedConceptsQA.

tively surfacing the parametric knowledge already encoded in instruct models.

Reasoning Trajectories Analysis. We present the responses from instruct and reasoning models in Figures 20–23. The instruct model only retrieves the correct answer under structured prompting by explicitly recalling the code or concept hierarchy, whereas the reasoning model correctly answers the question via spontaneous traversal. This shows that reasoning model internalizes the navigation policy required for accessing parametric knowledge, while structured prompting provides this guidance externally for instruct models.

3 Isolating RL's Effect on Unlocking Inaccessible Knowledge Through Controlled Injection

To isolate the effect of RL on parametric knowledge access from data exposure, we construct a controlled knowledge-injection setting using PopQA. We perform RL exclusively on facts that models have never seen during pre-training and cannot extract through direct retrieval, then evaluate whether this improves access to a disjoint set of facts that are frequently seen in pre-training but similarly inaccessible. If RL improves accuracy on these held-out high-exposure facts, it suggests that RL improves the model's ability to access latent parametric knowledge.

3.1 Experimental Setup

Controlled Knowledge Injection. To estimate how frequently each entity–answer pair co-occurs

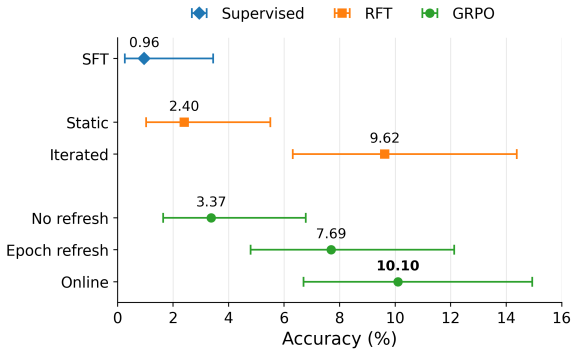


Figure 3: Decomposing RL on the controlled knowledge-injection task using Qwen3-8B-Instruct. Points indicate accuracy, and horizontal error bars indicate 95% Wilson confidence intervals.

in pretraining corpus, we use Infini-gram, an n-gram language model over large text corpora (Liu et al., 2024). The zero-co-occurrence examples that a model cannot answer correctly across three independent runs are then used as a proxy for unseen facts. Here we focus on Qwen3-8B-/30B-A3B-Instruct, evaluate them on the low-frequency examples and retain the subset that the models answer incorrectly as the RL training set. Similarly, examples with high co-occurrence counts that remain non-extractable are also retained as the held-out test set. The two subsets are subsequently filtered to use disjoint relations, eliminating relation-overlap confounds. We then apply RL to the rare, non-extractable subset until convergence and evaluate the trained models on this test set. To further isolate the RL contribution, we also train the models with SFT on the same data similarly. Additional experimental details are reported in Appendix B.5.

3.2 RL Unlocks Latent Knowledge Better Than SFT

RL on the rare, non-extractable data improves access to the high-frequency, inaccessible held-out set from 0% to 10.1% on Qwen3-8B-Instruct and from 0% to 7.05% on Qwen3-30B-A3B-Instruct, while SFT on the same data yields only 0.96% and 0.64%. Because the training and test splits are relation-disjoint and the test facts were never seen during fine-tuning, this gain cannot be explained by data exposure or relation overlap. This suggests that RL improves access to latent parametric knowledge already encoded during pretraining.

3.3 On-Policy Exploration Drives the Gain

The comparison above separates RL from SFT but does not identify which ingredient of RL matters. We therefore decompose the training objective along three axes—whether the training data is self-generated, how often it is refreshed, and whether incorrect rollouts are discarded or instead supply a negative gradient—holding the data and hyperparameters fixed on Qwen3-8B-Instruct so that only the objective changes (Figure 3).

Rejection-sampling fine-tuning (RFT) samples multiple rollouts for each question, retains only those that produce a correct answer, and trains on the retained rollouts with the standard supervised likelihood objective. Static RFT generates this training set once from the initial policy and keeps it fixed throughout training, whereas Iterated RFT regenerates rollouts from the current policy per batch.

Group Relative Policy Optimization (GRPO) (Shao et al., 2024) is an RL algorithm that scores a group of rollouts per question and updates the policy toward those scoring above the group mean and away from those below it. Within GRPO, we change how frequently the self-generated rollouts are refreshed while leaving the rest unchanged. Offline GRPO reuses rollouts sampled once from the initial policy; epoch-refresh GRPO regenerates them from the latest policy once per training epoch; and Online GRPO generates fresh rollouts from the current policy before every update. These variants therefore interpolate from fully stale to fully current training.

Three patterns emerge. First, switching from externally supplied answers to the model's own correct rollouts is necessary but not sufficient: replacing the externally supplied target answer with the model's own correct samples raises recall only from 0.96% to 2.40%. Second, iteration on self-generated data accounts for most of the effect: a single round of rejection-sampling fine-tuning barely moves recall (Static RFT, 2.40%), whereas repeating the sample–filter–finetune loop recovers most of it (Iterated RFT, 9.62%). The same ordering holds within GRPO, where recall rises monotonically as rollouts become fresher ($3.37\% \rightarrow 7.69\% \rightarrow 10.1\%$). Third, GRPO's contrastive advantage—which pushes above-average rollouts per question up while pushing below-average ones down—contributes little once iteration is present: Online GRPO (10.1%) and Iterated RFT (9.62%) are statistically indistinguishable in held-out accuracy. Table 6 reports pairwise confidence intervals supporting these comparisons. Improved knowledge access thus comes from iterated on-policy exploration, which is precisely what distilled models lack (Section 5.2).

4 Reasoning Improves Hierarchical Navigation Under Greater Retrieval Complexity

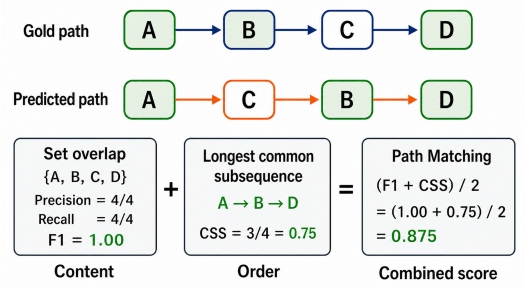


Figure 4: Illustration of the Path Matching score.

Section 2.2 showed that structured prompting recovers most of the instruct–reasoning gap on knowledge retrieval, but the benchmarks used there do not systematically vary the depth of traversal required to reach an answer. To probe this, we introduce IPC-CAR and DBO-CAR—two datasets stratified by retrieval depth—and a path matching metric to score traversal quality. Results show that reasoning models exhibit superior hierarchical traversal as depth increases.

4.1 Experimental Setup

Datasets.

- **IPC Common Ancestor Retrieval (IPC-CAR)**: a dataset used to evaluate model's fine-grained recall ability under controlled retrieval complexity. Each question presents two leaf IPC codes and asks the model to identify their nearest common ancestor (NCA). We define retrieval depth as the total number of edges that must be recalled along the two upward paths. Since each leaf is at least one edge away from the NCA, the minimum depth is two.
- **DBpedia Ontology Common Ancestor Retrieval (DBO-CAR)**: Derived from Wikipedia infoboxes, the community-curated DBpedia Ontology covers 768 classes across domains like people, organizations, and species (Mendes et al., 2012). We use it to

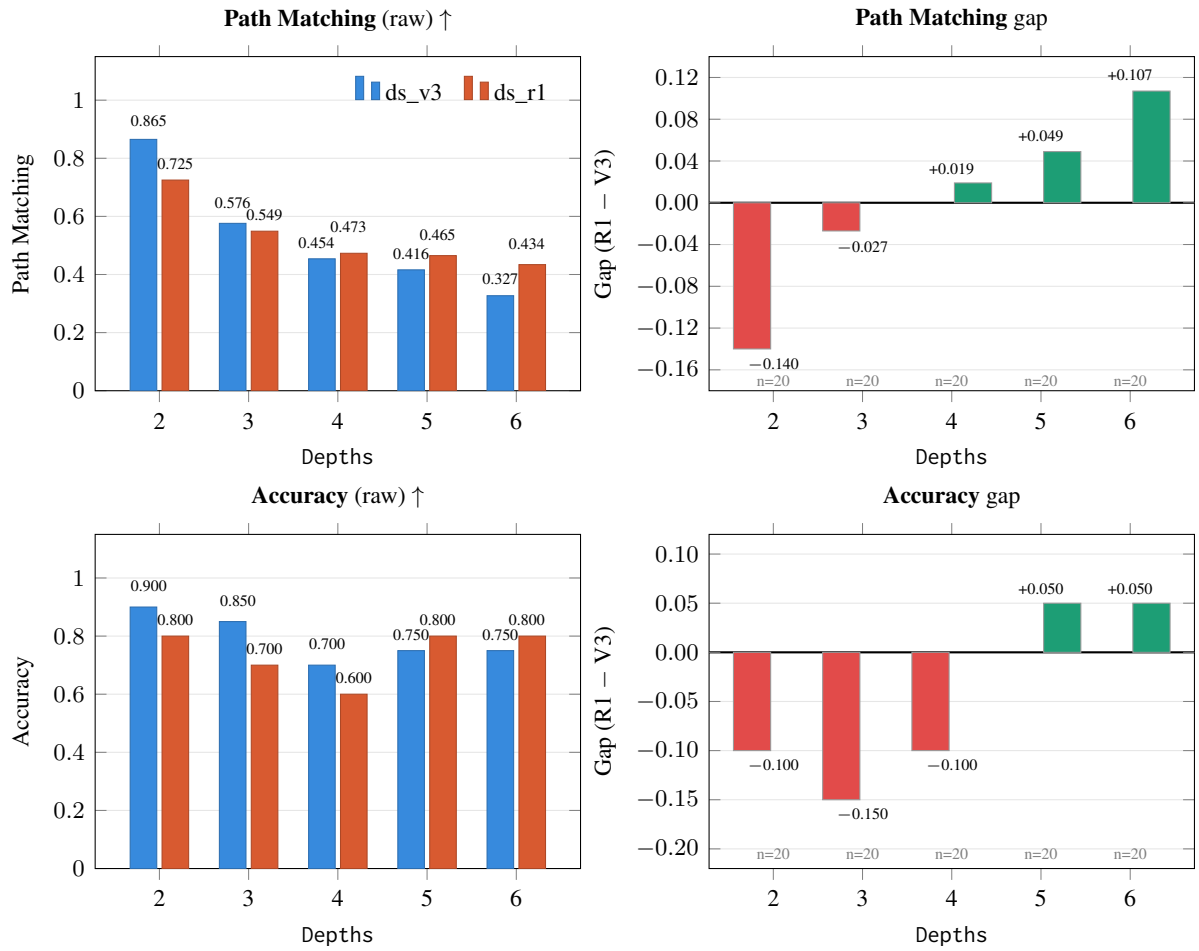


Figure 5: Path matching score and final-answer accuracy of DeepSeek-V3 (Instruct) versus DeepSeek-R1 (Reasoning) on IPC-CAR, stratified by depth. **Left column:** raw scores; arrows (↑) indicate higher is better. **Right column:** gap (R1 − V3) plotted on a tight y-axis centered on zero, with positive gaps (R1 better) in green and negative gaps (R1 worse) in red. Each level has n=20 questions.

construct DBO-CAR, an ontology-grounded counterpart to IPC-CAR that tests recall over CamelCase concept names rather than alphanumeric patent codes. Full construction details for IPC/DBO-CAR are in Section B.4.2 and B.4.3.

Path Matching Score. To evaluate the quality of predicted hierarchical paths, we propose the path matching score, which combines two metrics:

- **F1 Score:** The harmonic mean of precision P and recall R over the set of predicted vs. ground-truth ancestors, $F_1 = \frac{2 \times P \times R}{P+R}$ (Buckland and Gey, 1994).
- **Common Subsequence Score (CSS):** Evaluates path order fidelity as the length of the Longest Common Subsequence (LCS) (Paterson and Dančík, 1994) between the predicted and ground-truth paths, normalized by ground-truth length: $\text{CSS} = \frac{|\text{LCS}(\text{predicted}, \text{ground truth})|}{|\text{ground truth ancestors}|}$.

The Path Matching score is the mean of the two components, $\text{PathMatch} = \frac{F_1 + \text{CSS}}{2}$, capturing both set-level accuracy and order fidelity in hierarchical recall, as illustrated in Figure 4.

4.2 Reasoning Models Show Superior Path Recall When Traversal Depth Increases

We evaluate DeepSeek-V3 (Instruct) and DeepSeek-R1 (Reasoning) on IPC-CAR, stratified by retrieval depth (Figure 5).

A clear crossover emerges: V3 outperforms R1 at shallow depths (PathMatch 0.865 vs. 0.725 at depth 2), but R1 overtakes V3 from depth 4 onward, with the gap widening to +0.107 at depth 6. While final-answer accuracy shows the same directional crossover, only PathMatch captures the depth-dependent widening. R1's advantage under greater retrieval complexity shows procedural navigation beyond simple memorization. DBO-CAR

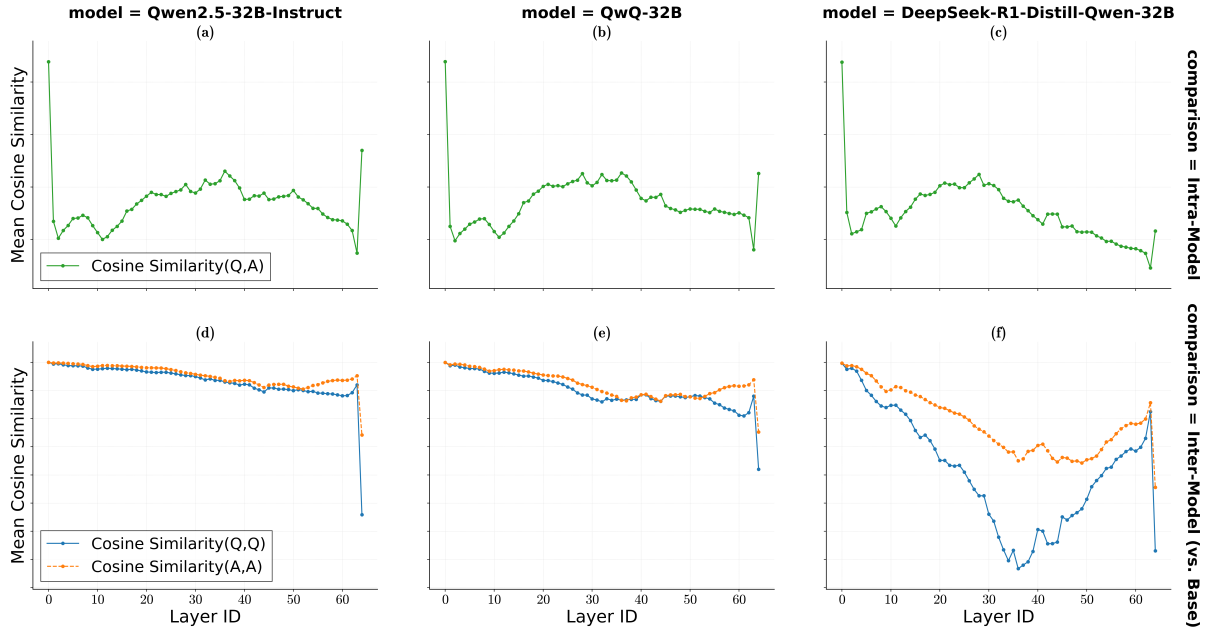


Figure 6: Layerwise cosine similarity of question (Q) and answer (A) representations on ICD9PROC. **Top row (a–c)**: intra-model Q–A similarity within the instruct, reasoning, and distilled variants of Qwen2.5. **Bottom row (d–f)**: inter-model Q–Q and A–A similarity between each variant and the base model.

shows similar pattern (Appendix C.3).

5 Mechanistic and Failure-Mode Analysis

5.1 Layerwise Representation Analysis

To localize how post-training reshapes internal knowledge processing, we analyze layerwise activations on MedConceptsQA using contrastive question–answer probes, comparing the base model against its instruct, reasoning, and distilled variants.

5.1.1 Experimental Setup

Probe Construction. We sample 100 question–answer pairs from the `ICD9PROC` vocabulary in MedConceptsQA. Each pair consists of a factual question and its ground-truth description, formatted as a declarative statement. For example, code `87.15` yields:

> *Question:* What is the description of the medical code `87.15` in `ICD9PROC`?
>
> *Answer:* The description of the medical code `87.15` in `ICD9PROC` is contrast radiogram of sinus.

For a model with L layers and hidden dimension d, we process each probe's question and answer independently. At every layer ℓ, we extract the final-token hidden state $\mathbf{h}_\ell \in \mathbb{R}^d$, which summarizes the full preceding context at that layer.

Analyses. We compute cosine similarity between representations under two regimes:

- **Inter-model**: For each layer, we compare $\mathbf{Q}^{\text{base}}$ to $\mathbf{Q}^{\text{post-trained}}$ and $\mathbf{A}^{\text{base}}$ to $\mathbf{A}^{\text{post-trained}}$, measuring how much post-training moves question vs. answer representations.
- **Intra-model**: For each model in isolation, we compare $\mathbf{Q}$ to $\mathbf{A}$ layer-by-layer, tracing how internal activations evolve from encoding a question to producing an answer.

Similarity at layer ℓ, cosine similarity for source pair (a, b) is $\cos^{(a,b)}(\ell) = \frac{1}{N}\sum_{i=1}^{N} \frac{\mathbf{h}_\ell^{(a)}(i)^\top \mathbf{h}_\ell^{(b)}(i)}{\|\mathbf{h}_\ell^{(a)}(i)\|_2 \, \|\mathbf{h}_\ell^{(b)}(i)\|_2}$, averaged over $N = 100$ probes (formal notation in Appendix B.6).

Models. We compare Qwen2.5-32B (base) against its instruct, reasoning, and distilled variants. Mistral-Small-24B results are in Appendix C.4.

5.1.2 Instruct and Reasoning Models Reshape Question Processing, Not Stored Knowledge

Figure 6 shows layerwise similarity trends; Tables 9 and 10 report inter- and intra-model values aggregated into early (1–21), middle (22–43), and late (44–64) bands. We highlight two findings:

- Within each model, question and answer representations start highly similar but diverge in later layers (e.g., the reasoning variant's

Q–A similarity drops from 0.878 at layer 0 to 0.161 at layer 63; Figure 6), indicating that representations accumulate increasingly distinct features.

- Instruct and reasoning models stay closely aligned with the base (e.g., mean Q/A similarity: 0.966/0.976 for instruct; Table 9), whereas the distilled model diverges substantially (mean Q/A: 0.808/0.892), mirroring its degraded knowledge retrieval (Section 5.2). Notably, question representations diverge more than answers across all post-trained models, suggesting that performance gains arise primarily from refining question understanding rather than reorganizing factual knowledge.

5.2 Distillation Captures Surface Patterns Without Navigation

Table 2: Qwen2.5-32B model series majority-vote accuracy on MedConceptsQA.

Model	QA	CoT	Structured
Base	0.221	0.332	0.404
Distilled	0.375	0.380	0.447
Instruct	0.379	0.475	0.469
Reasoning	**0.482**	**0.513**	**0.505**

In addition to SFT and RL, distillation — training on CoT traces from a capable teacher model — is another widely used approach to improving model performance (Guo et al., 2025; Yue et al., 2025). To analyze the distillation effectivess, we examine the Qwen2.5-32B model series, comparing four variants: base, instruct, reasoning, and distilled, which is trained on responses from DeepSeek-R1. As shown in Table 2, on MedConceptsQA, the distilled model outperforms its base yet remains below both the instruct and reasoning versions, even under structured prompting. Additional results are reported in Table 11.

To investigate this, we analyze 40 responses from DeepSeek-R1-Distill-Qwen2.5-32B that lead to wrong answers. We find that models fail to perform proper hierarchical traversal: whereas DeepSeek-R1 reconstructs multi-level paths (e.g., ICD9PROC $\rightarrow$ 81 $\rightarrow$ 81.3 $\rightarrow$ 81.36), the distilled model frequently stops at shallow prefixes or bypasses code traversal entirely. Examining these traces, we identify two failure modes. The most prevalent (80%) happens when distilled models appear to simulate self-correction linguistically using keywords such as "wait", yet never truly uncommit from their initial hypothesis. Here the model forms a strong initial prior and filters alternatives through shallow pattern matching against its initial choice. For the remaining 20%, model backtracks across multiple answers before selecting the wrong one. This suggests that, unlike reasoning models, which learn a navigation policy through trial-and-error exploration, SFT-based distillation captures only surface-level self-correction patterns without learning to uncommit from mistakes. As a result, distilled model lacks the navigation skills to unlock the parametric knowledge. Additional reasoning trajectories for each failure mode are provided in Table 12.

6 Conclusion

We showed that RL improves how LLMs *access* knowledge rather than *what* they store. Structured prompting recovers most of the instruct–reasoning gap on knowledge retrieval, showing that instruct models already possess the underlying knowledge but lack the navigation policy that RL internalizes. It therefore provides a low-cost, training-free alternative for recovering much of the reasoning advantage by explicitly guiding instruct models to traverse knowledge already stored in their parameters. A controlled knowledge-injection experiment confirms that RL on unseen data can re-surface high-exposure facts that are previously inaccessible. Across RFT and GRPO variants, performance improves as self-generated rollouts are refreshed more frequently, while Online GRPO and Iterated RFT perform similarly. This points to iterated on-policy exploration as the main driver of the gain. On depth-stratified IPC-CAR and DBO-CAR benchmarks, the instruct–reasoning gap widens with retrieval depth, indicating that reasoning models internalize a scalable traversal capacity. Layer-wise analysis shows that post-training primarily reshapes query representations while leaving factual representations stable, and failure-mode analysis explains why distilled models, which imitate self-correction without acquiring exploratory behavior, fail to inherit this navigation ability. Together, these findings reframe the instruct–reasoning gap on knowledge-intensive tasks as an access bottleneck rather than a knowledge gap, and suggest that future studies should optimize retrieval procedures alongside knowledge content.

Limitations

First, we use co-occurrence counts as a proxy for whether data was encountered during pretraining, but it is not a definitive measure of training exposure. Second, while our controlled knowledge-injection experiment attempts to isolate the effect of RL on knowledge access, it cannot fully account for all the differences of a given instruct/reasoning model pair due to their closed training recipes.

Ethical considerations

In this paper, we only use open source datasets or databases and do not collect any sensitive data. This work studies how post-training affects access to parametric knowledge. The main potential risk is that improving parametric knowledge access may also make models more likely to retrieve incorrect or outdated facts. Our evaluations are carried out on verifiable benchmarks, and improved benchmark performance should not be interpreted as reliable factuality in real-world deployment.

Acknowledgments

We gratefully acknowledge the Foresight Institute for supporting this work.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, and 257 others. 2023. Gpt-4 technical report.

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. 2024. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Lakshya A Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alex Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. 2026. GEPA: Reflective prompt evolution can outperform reinforcement learning. In *The Fourteenth International Conference on Learning Representations, ICLR 2026, Rio de Janeiro, Brazil, April 23-27, 2026*. OpenReview.net.

Zeyuan Allen-Zhu and Yuanzhi Li. 2024. Physics of language models: Part 3.1, knowledge storage and extraction. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Amanda Askell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Ben Mann, Nova DasSarma, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Jackson Kernion, Kamal Ndousse, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, and 3 others. 2021. A general language assistant as a laboratory for alignment.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, and 32 others. 2022. Constitutional AI: Harmlessness from AI feedback.

Lukas Berglund, Meg Tong, Maximilian Kaufmann, Mikita Balesni, Asa Cooper Stickland, Tomasz Korbak, and Owain Evans. 2024. The reversal curse: Llms trained on "a is b" fail to learn "b is a". In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, and 9 others. 2022. Improving language models by retrieving from trillions of tokens. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 2206–2240. PMLR.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Michael Buckland and Fredric Gey. 1994. The relationship between recall and precision. *Journal of the American society for information science*, 45(1):12–19.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V Le, Sergey Levine, and Yi Ma. 2025. SFT memorizes, RL generalizes: A comparative study of foundation model post-training. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, Canada, July 13-19, 2025*. OpenReview.net.

Xingyu Dang, Christina Baek, J Zico Kolter, and Aditi Raghunathan. 2025. Assessing diversity collapse in reasoning. In *Scaling Self-Improving Foundation Models without Human Supervision*.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, and 181 others. 2025. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

Zorik Gekhman, Roee Aharoni, Eran Ofek, Mor Geva, Roi Reichart, and Jonathan Herzig. 2026. Thinking to recall: How reasoning unlocks parametric knowledge in llms.

Zorik Gekhman, Gal Yona, Roee Aharoni, Matan Eyal, Amir Feder, Roi Reichart, and Jonathan Herzig. 2024. Does fine-tuning LLMs on new knowledge encourage hallucinations? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7765–7784, Miami, Florida, USA. Association for Computational Linguistics.

Sreyan Ghosh, Chandra Kiran Reddy Evuru, Sonal Kumar, Ramaneswaran S., Deepali Aneja, Zeyu Jin, Ramani Duraiswami, and Dinesh Manocha. 2024. A closer look at the limitations of instruction tuning. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nat.*, 645(8081):633–638.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Retrieval augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, Proceedings of Machine Learning Research, pages 3929–3938. PMLR.

Maggie Huan, Yuetai Li, Tuney Zheng, Xiaoyu Xu, Seungone Kim, Minxin Du, Radha Poovendran, Graham Neubig, and Xiang Yue. 2025. Does math reasoning improve general llm capabilities? understanding transferability of llm reasoning. *ArXiv preprint*, abs/2507.00432.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan A, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. 2024. DSPy: Compiling declarative language model calls into state-of-the-art pipelines. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Robert Kirk, Ishita Mediratta, Christoforos Nalmpantis, Jelena Luketina, Eric Hambro, Edward Grefenstette, and Roberta Raileanu. 2024. Understanding the effects of RLHF on LLM generalisation and diversity. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James Validad Miranda, Alisa Liu, Nouha Dziri, Xinxi Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Christopher Wilhelm, Luca Soldaini, and 4 others. 2025. Tulu 3: Pushing frontiers in open language model post-training. In *Second Conference on Language Modeling*.

Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Junliang Li, Yucheng Wang, Yan Chen, Yu Ran, Ruiqing Zhang, Jing Liu, Hua Wu, and Haifeng Wang. 2025. Knowledge-level consistency reinforcement learning: Dual-fact alignment for long-form factuality. *ArXiv preprint*, abs/2509.23765.

Jintao Liang, Sugang, Huifeng Lin, You Wu, Rui Zhao, and Ziyue Li. 2025. Reasoning RAG via system 1 or system 2: A survey on reasoning agentic retrieval-augmented generation for industry challenges. In *Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pages 1954–1966, Mumbai, India. The Asian Federation of Natural Language Processing and The Association for Computational Linguistics.

Yong Lin, Hangyu Lin, Wei Xiong, Shizhe Diao, Jianmeng Liu, Jipeng Zhang, Rui Pan, Haoxiang Wang,

Wenbin Hu, Hanning Zhang, Hanze Dong, Renjie Pi, Han Zhao, Nan Jiang, Heng Ji, Yuan Yao, and Tong Zhang. 2024. Mitigating the alignment tax of RLHF. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Jiacheng Liu, Sewon Min, Luke Zettlemoyer, Yejin Choi, and Hannaneh Hajishirzi. 2024. Infini-gram: Scaling unbounded n-gram language models to a trillion tokens. In *First Conference on Language Modeling*.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. 2025. Understanding r1-zero-like training: A critical perspective. In *Second Conference on Language Modeling*.

Melody Ma and John Hewitt. 2026. Improving parametric knowledge access in reasoning language models.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9802–9822, Toronto, Canada. Association for Computational Linguistics.

Pablo Mendes, Max Jakob, and Christian Bizer. 2012. DBpedia: A multilingual cross-domain knowledge base. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 1813–1817, Istanbul, Turkey. European Language Resources Association (ELRA).

Sagnik Mukherjee, Lifan Yuan, Dilek Hakkani-Tür, and Hao Peng. 2025. Reinforcement learning finetunes small subnetworks in large language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. 2021. Webgpt: Browser-assisted question-answering with human feedback.

Mike Paterson and Vlado Dančík. 1994. Longest common subsequences. In *International symposium on mathematical foundations of computer science*, pages 127–142. Springer.

Hoang Phan, Xianjun Yang, Kevin Yao, Jingyu Zhang, Shengjie Bi, Xiaocheng Tang, Madian Khabsa, Lijuan Liu, and Deren Lei. 2025. Beyond reasoning gains: Mitigating general capabilities forgetting in large reasoning models. *ArXiv preprint*, abs/2510.21978.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2024. Toolllm: Facilitating large language models to master 16000+ real-world apis. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal V. Nayak, Debajyoti Datta, and 21 others. 2022. Multitask prompted training enables zero-shot task generalization. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models.

Ofir Ben Shoham and Nadav Rappoport. 2024. Medconceptsqa: Open source medical concepts qa benchmark. *Computers in Biology and Medicine*, 182:109089.

Aditi Singh, Abul Ehtesham, Saket Kumar, Tala Talaei Khoei, and Athanasios V. Vasilakos. 2025. Agentic retrieval-augmented generation: A survey on agentic rag.

Oscar Skean, Md Rifat Arefin, Dan Zhao, Niket Patel, Jalal Naghiyev, Yann LeCun, and Ravid Shwartz-Ziv. 2025. Layer by layer: Uncovering hidden representations in language models. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, Proceedings of Machine Learning Research. PMLR / OpenReview.net.

Taylor Sorensen, Benjamin Newman, Jared Moore, Chan Young Park, Jillian Fisher, Niloofar Mireshghallah, Liwei Jiang, and Yejin Choi. 2026. Spectrum tuning: Post-training for distributional coverage and in-context steerability. In *The Fourteenth International Conference on Learning Representations, ICLR 2026, Rio de Janeiro, Brazil, April 23-27, 2026*. OpenReview.net.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, S. H. Cai, Yuan Cao, Y. Charles, H. S. Che, Cheng Chen,

Guanduo Chen, Huarong Chen, Jia Chen, Jiahao Chen, Jianlong Chen, Jun Chen, Kefan Chen, Liang Chen, Ruijue Chen, Xinhao Chen, and 307 others. 2026. Kimi k2.5: Visual agentic intelligence.

Kimi Team, Yifan Bai, Yiping Bao, Y. Charles, Cheng Chen, Guanduo Chen, Haiting Chen, Huarong Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, Zhuofu Chen, Jialei Cui, Hao Ding, Mengnan Dong, Angang Du, and 181 others. 2025. Kimi k2: Open agentic intelligence.

Yiping Wang, Qing Yang, Zhiyuan Zeng, Liliang Ren, Liyuan Liu, Baolin Peng, Hao Cheng, Xuehai He, Kuan Wang, Jianfeng Gao, Weizhu Chen, Shuohang Wang, Simon Shaolei Du, and yelong shen. 2025a. Reinforcement learning for reasoning in large language models with one training example. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2–7, 2025*. OpenReview.net.

Zengzhi Wang, Fan Zhou, Xuefeng Li, and Pengfei Liu. 2025b. Octothinker: Mid-training incentivizes reinforcement learning scaling. *ArXiv preprint*, abs/2506.20512.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

World Intellectual Property Organization. 2026. International patent classification (IPC), version 2026.01. `https://www.wipo.int/en/web/classification-ipc/`. Accessed: 2026.

Fang Wu, Weihao Xuan, Ximing Lu, Mingjie Liu, Yi Dong, Zaid Harchaoui, and Yejin Choi. 2025. The invisible leash: Why rlvr may or may not escape its origin. *ArXiv preprint*, abs/2507.14843.

Tian Xie, Zitian Gao, Qingnan Ren, Haoming Luo, Yuqian Hong, Bryan Dai, Joey Zhou, Kai Qiu, Zhirong Wu, and Chong Luo. 2025. Logic-rl: Unleashing llm reasoning with rule-based reinforcement learning.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, and 43 others. 2024. Qwen2 technical report. *Preprint*, arXiv:2407.10671.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Jiayuan Ye, Vitaly Feldman, and Kunal Talwar. 2026. Cram less to fit more: Training data pruning improves memorization of facts.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, YuYue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, LingJun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, and 17 others. 2026. DAPO: An open-source LLM reinforcement learning system at scale. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025*. OpenReview.net.

Jiaqing Yuan, Lin Pan, Chung-Wei Hang, Jiang Guo, Jiarong Jiang, Bonan Min, Patrick Ng, and Zhiguo Wang. 2024. Towards a holistic evaluation of llms on factual knowledge recall. *ArXiv preprint*, abs/2404.16164.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. 2025. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond the base model? In *Advances in Neural Information Processing Systems*.

Rosie Zhao, Alexandru Meterez, Sham M. Kakade, Cengiz Pehlevan, Samy Jelassi, and Eran Malach. 2025. Echo chamber: RL post-training amplifies behaviors learned in pretraining. In *Second Conference on Language Modeling*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V. Le, and Ed H. Chi. 2023. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Noah Ziems, Dilara Soylu, Lakshya A Agrawal, Isaac Miller, Liheng Lai, Chen Qian, Kaiqiang Song, Meng Jiang, Dan Klein, Matei Zaharia, and 1 others. 2025. Multi-module grpo: Composing policy gradients and prompt optimization for language model programs. *ArXiv preprint*, abs/2508.04660.

Appendix

A Related Work

A.1 The Alignment Tax and Factual Degradation

The trade-off between alignment and factual accuracy has been extensively explored. Lin et al. (2024) introduced the concept of "alignment tax", showing systematic degradation on factual benchmarks as RLHF reward strength increases. Achiam et al. (2023) similarly reported that RLHF "does not improve exam performance (without active effort, it actually degrades it)" and can reduce calibration. Mechanistic analyses in Ghosh et al. (2024) reveal that instruction tuning primarily adjusts style rather than new knowledge. In particular, responses generated from pre-trained knowledge consistently outperform those from models learning new knowledge through instruction tuning. Both Li et al. (2025) and Kirk et al. (2024) show that base models' parametric knowledge originates from pretraining while aligned models learn how to express it. In addition, Li et al. (2025) shows that training directly from base models mitigates knowledge forgetting and alignment tax incurred by SFT-based distillation. Recent work by Phan et al. (2025) shows a trade-off in Reinforcement Learning from Verifiable Rewards (RLVR): while optimizing for verifiable rewards improves reasoning on targeted tasks, it can degrade general capabilities acquired from pretraining and increase hallucinations.

While these studies document factual degradation from alignment, our work reveals a contrasting phenomenon: reasoning models *outperform* their base counterparts on structured knowledge recall. This suggests that alignment tax may not uniformly affect all forms of parametric knowledge retrieval, especially when retrieval needs systematic navigation rather than direct factual recall.

A.2 Retrieval-augmented generation (RAG)

RAG grounds model outputs by retrieving relevant documents at test time and conditioning generation on the retrieved context (Guu et al., 2020; Lewis et al., 2020; Borgeaud et al., 2022). A related line of work extends RAG to tool-calling and agentic settings (Nakano et al., 2021; Asai et al., 2024; Qin et al., 2024; Schick et al., 2023; Yao et al., 2023; Liang et al., 2025; Singh et al., 2025; Ye et al., 2026), equipping models with external tools, APIs, and web search alongside planning, reflection, and multi-agent collaboration. These systems can dynamically decide when to retrieve, what evidence to gather, and how to integrate it across reasoning steps. Our work is orthogonal: instead of augmenting LLMs through external retrieval, we ask whether models can reliably access factual knowledge already stored in their parameters. This isolates parametric knowledge access from retrieval quality, tool selection, context construction, and the reliability of external pipelines.

A.3 RL and Prompting for Reasoning

RL and prompting have emerged as the two central approaches for eliciting long-form reasoning behaviors in LLMs. Systems such as OpenAI o1 and DeepSeek-R1 show that RL can encourage models to produce extended chains of thought, self-reflect, backtrack, and verify intermediate steps (Achiam et al., 2023; Shao et al., 2024; Guo et al., 2025; Lambert et al., 2025; Xie et al., 2025; Wang et al., 2025a; Yu et al., 2026). Consequently, this leads to strong gains on mathematics, coding, and other verifiable reasoning tasks. Optimized prompting can match or exceed fine-tuned and reasoning models on reasoning tasks (Brown et al., 2020; Sanh et al., 2022; Wei et al., 2022; Kojima et al., 2022; Zhou et al., 2023; Agrawal et al., 2026).

Our work differs from both lines of research. While prior work studies RL and prompting as ways to improve reasoning, we investigate whether RL also improves a model's ability to retrieve parametric knowledge. We do not draw a hard boundary between reasoning and navigation; rather, we aim to isolate what RL improves in our setting—the traversal procedure used to access information, or the underlying knowledge itself. To this end, we use prompting strictly as a diagnostic tool. Because the prompt only tells a model how to navigate a knowledge hierarchy, success relies on accessing existing parametric memory, allowing us to attribute performance gaps to inefficient knowledge access rather than missing information.

A.4 Reasoning for Knowledge Retrieval

Reasoning over parametric knowledge recall has gained attention. Ma and Hewitt (2026) show that RL on QA tasks improves factual retrieval on various out-of-distribution benchmarks, but do not isolate whether the gains come from improved knowledge access or from data exposure. Through a frequency-stratified, relation-disjoint experiment, we show that RL on rare facts surfaces common,

previously non-extractable knowledge, thus decoupling access from exposure. Gekhman et al. (2026) study single-hop factual questions and identify computational buffering and factual priming as the drivers behind reasoning-aided recall. While their work focuses on expanding the knowledge boundary via pass@k and test-time recall improvements, we focus on reliable knowledge access via majority-voted accuracy and systematically compare how SFT, RL, and distillation affect this knowledge access. Finally, while these works study unstructured factual recall, our work investigates explicit and implicit hierarchical navigation within structured and unstructured knowledge. We also isolate SFT/RL/distillation's internal mechanisms through a layer-wise representational analysis.

A.5 Knowledge Storage versus Knowledge Access

Recent evidence suggests that improving how a model accesses existing knowledge may be equally important as expanding what it knows. Allen-Zhu and Li (2024) and Berglund et al. (2024) show that models trained on "A is B" often fail to answer the reverse query "B is A". This indicates that factual knowledge can be directionally inaccessible even when it was seen during training. An analogous pattern has been observed for reasoning: RL improves the sampling efficiency of correct solution paths rather than expanding the base model's reasoning boundary (Dang et al., 2025; Liu et al., 2025; Wu et al., 2025; Yue et al., 2025; Zhao et al., 2025). Whether a similar access-versus-storage distinction holds for parametric factual knowledge under RL remains unstudied. To address this gap, we systematically compare the effect of RL against other post-training methods including distillation and SFT, using prompting interventions and layer-wise representation analysis. We further validate our findings through majority-vote response distributions, knowledge-injection experiments, and distillation failure-mode analysis.

B Technical Appendices and Supplementary Material

B.1 Terminology

Table 3 defines the two recurring notions—*knowledge access* and *hierarchical traversal*—used throughout the paper.

Term	Definition
Knowledge access (retrieval)	A model's *ability* to surface knowledge already encoded in its parameters; a fact can be stored yet inaccessible.
Hierarchical traversal (navigation)	The *procedure* of navigating explicit taxonomies or implicit entity relations step by step (e.g., chapter → category → code) to reach the queried fact.

Table 3: Terminology used throughout the paper.

B.2 Zero-Shot Prompt Templates

We present the three zero-shot prompt templates used in IPC and MedConceptsQA in Figures 15, 16, and 17:

- **Direct QA**: the model outputs only a single answer letter with no explanation.
- **CoT**: the model provides a final answer together with a free-form explanation.
- **Structured**: the model is instructed to recall the hierarchical breakdown of relevant facts or codes before reaching the final answer.

B.3 Evaluation Protocol for Structured Prompting Experiments

For PopQA, we use exact-match over open-ended answers; for IPC and MedConceptsQA, we use multiple-choice accuracy. To avoid any confound from in-context demonstrations on knowledge access, we evaluate all model pairs in a zero-shot setting. Due to compute constraints, we only evaluate on 5,000 randomly selected questions from PopQA.

B.4 Dataset Construction Details

B.4.1 IPC-Lookup

Dataset Construction. Two representative questions from the IPC dataset are shown in Figures 7 and 8. Each question presents a patent code and four candidate descriptions; the correct answer requires recalling the precise code description within the IPC taxonomy. Distractors are drawn from structurally adjacent nodes (sharing the same parent or grandparent), avoiding coarse domain elimination. Questions containing code strings in the description are excluded to prevent leakage. All correct answers are randomly assigned to A/B/C/D to avoid position bias. All questions in the dataset were constructed manually and cross-checked by two annotators. Two example questions are provided below.

Example 1 — IPC-Lookup E01B 11/50

Question: What does the patent code E01B 11/50 in the International Patent Classification describe?

A. Joint constructions for relatively movable rails, e.g. rails on turntables, traversers, or swing bridges
B. Non-dismountable rail joints; Welded joints; e.g., joints made by electric welding
C. Special arrangements for supporting rail ends
D. Electrically-insulating rail joints

Correct Answer: A *(Subgroup of E01B 11/00)*

Figure 7: IPC example question for code `E01B 11/00`.

Example 2 — IPC-Lookup H01B 7/20

Question: What does the patent code `H01B 7/20` in the International Patent Classification describe?

A. Extensible conductors or cables, e.g. self-coiling cords
B. Protection against damage caused by external factors, by wear, mechanical force or pressure; metal tubes, e.g. lead sheaths
C. Submarine cables
D. Rigid-tube cables

Correct Answer: B *(Subgroup of H01B 7/00)*

Figure 8: IPC example question for code `H01B 7/20`.

B.4.2 IPC-CAR

We use the NCA task to probe the fine-grained recall capabilities of instruct and reasoning models because it forces the model to traverse upward paths in the hierarchy rather than recognize a single code. The dataset was generated by prompting Claude 4.6 Sonnet with the parsing rules, task specification, and constraints shown in Figure 17. Distractors are sampled from neighbors of the correct answer. Two annotators then cross-check every question for hierarchical paths and answer correctness. We construct questions with depths ranging from 2 to 6, with 20 questions per depth and 100 questions in total. Answer options are randomly permuted across A/B/C/D to avoid position bias. Two example questions at retrieval depths 3 and 6 are shown in Figures 9 and 10.

Example 1 — IPC-CAR Depth 3

Question: Nearest common ancestor of A47L 5/24 and A47L 5/30 is:
A) A47L 5/22 B) A47L 5/16 C) A47L 5/12 D) A47L 5/14

Hierarchical Paths:

- A47L 5/24 → A47L 5/22
- A47L 5/30 → A47L 5/28 → A47L 5/22

Answer: A

Figure 9: IPC common-ancestor example at depth 3.

Example 2 — IPC-CAR Depth 6

Question: Nearest common ancestor of A61K 31/713 and A61K 31/7064 is:
A) A61K 31/095 B) A61K 31/70 C) A61K 35/22 D) A61K 8/73

Hierarchical Paths:

- A61K 31/713 → A61K 31/7088 → A61K 31/70
- A61K 31/7064 → A61K 31/706 → A61K 31/7052 → A61K 31/7042 → A61K 31/70

Answer: B

Figure 10: IPC common-ancestor example at depth 6.

B.4.3 DBO-CAR

We construct 500 questions across depths 2–6 (100 per depth), following the same procedure as IPC-CAR. Figures 11 and 12 are two representative examples from DBO-CAR.

Example 1 — DBO-CAR `Depth`= 2

Question: Nearest common ancestor of `SoftballLeague` and `MixedMartialArtsLeague` is:

A. `SambaSchool`
B. `SportsLeague`
C. `InternationalOrganisation`
D. `Broadcaster`

Hierarchical Paths:

- `SoftballLeague` → `SportsLeague`
- `MixedMartialArtsLeague` → `SportsLeague`

Answer: B

Figure 11: DBO-CAR common-ancestor example at `depth` 2.

Example 2 — DBO-CAR Depth 5

Question: Nearest common ancestor of `CultivatedVariety` and `Psychologist` is:

A. `Person`
B. `Animal`
C. `Eukaryote`
D. `Bacteria`

Hierarchical Paths:

- `CultivatedVariety` → `Plant` → `Eukaryote`
- `Psychologist` → `Person` → `Animal` → `Eukaryote`

Answer: C

Figure 12: DBO-CAR common-ancestor example at `depth 5`.

B.5 Controlled Knowledge Injection through RL

Entity-Answer Co-occurrence Rates. First we randomly sample 5,000 PopQA questions, then we extract the subject entity and relation using DeepSeek-V3.1-NonThink with the prompt shown in Figure 14. We use Infini-gram to count entity-answer co-occurrences in the Dolma pre-training corpus within a window of `max_diff_tokens=100` (the default). Questions are binned by co-occurrence count into four groups: $a = 0$, $0 < a \leq 10$, $10 < a \leq 100$, $100 < a \leq 1{,}000$, $1{,}000 < a \leq 10{,}000$ and $a > 10{,}000$. The $a = 0$ bin isolates unseen facts and serves as the source for the low-frequency, inaccessible training set; the $a > 100$ bin is merged from the original $(100, 1{,}000]$, $(1{,}000, 10{,}000]$ and $(10{,}000, \infty)$ ranges to ensure a sufficiently large high-frequency, inaccessible evaluation set. Per-bin counts are shown in Figure 13.

Inaccessible Subgroups. We evaluate Qwen3-8B (Instruct) on the high-frequency and low-frequency bins, retaining only questions that the model answers incorrectly across three independent runs. They are subsequently filtered to contain disjoint sets of relation as shown in Table 4. This yields 731 questions as the low-frequency, inaccessible training set and 208 questions as the high-frequency, inaccessible held-out evaluation set. The identical pipeline is repeated for Qwen3-30B-A3B.

SFT/RL training details. The learning rate for each model is selected by a grid search over 1×10^{-4}, 5×10^{-4}, 1×10^{-5}, 4×10^{-5}, 1×10^{-6}, 5×10^{-6}, 8×10^{-6}. Each model is evaluated on a randomly chosen subset of 150 questions from low-frequency, inaccessible training set for 30 RL steps. For each learning rate, we compute the mean over the last third of steps as the selection metric instead of the overall reward, avoiding early reward spikes. The reward is a graded contains-check against the list of acceptable answers: 1.0 if any synonym appears in the response and 0.0 for an incorrect response. Advantages are computed using Group Relative Policy Optimization (GRPO). The RL hyperparameters are reported in Table 5 (same for both the learning rate search and the main RL training). SFT uses the same setup but instead trains the model to predict a randomly sampled candidate answer given the question. For the main training, the selected rate is 4×10^{-5} for Qwen3-8B-Instruct and 5×10^{-4} for Qwen3-30B-A3B-Instruct; all runs converge within 140 steps and we report results at this checkpoint.

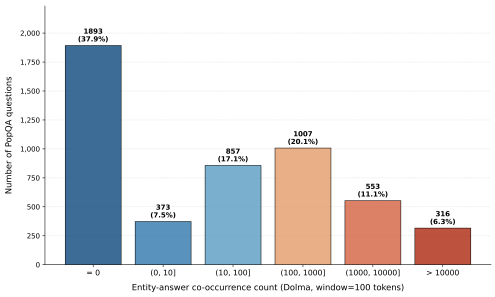


Figure 13: Distribution of PopQA entity-answer co-occurrence counts in the Dolma pre-training corpus.

Wilson Confidence Intervals. For each method, we treat the correctness of held-out questions as a binary vector $y_i \in \{0, 1\}$. For the accuracies in Figure 3, we compute $\hat{p} = x/n$, where $x = \sum_{i=1}^{n} y_i$ and $n = 208$, and report a 95% Wilson score interval. With $z = 1.96$, let

$$c = \frac{\hat{p} + z^2/(2n)}{1 + z^2/n}, \qquad h = \frac{z\sqrt{\hat{p}(1-\hat{p})/n + z^2/(4n^2)}}{1 + z^2/n}. \tag{1}$$

The reported interval is $[c - h, c + h]$.

Paired Bootstrap Confidence Intervals. For comparison between methods A and B, we retain their binary correctness vectors on the same $n = 208$ held-out questions and compute the accuracy difference

$$\widehat{\Delta} = \frac{1}{n}\sum_{i=1}^{n}\left(y_i^{(A)} - y_i^{(B)}\right). \tag{2}$$

Then we draw question indices with replacement and recompute the difference for that bootstrap

Comparison	Δ Acc. (pp)	95% paired-bootstrap CI (pp)
Static RFT − SFT	+1.44	[−0.96, 3.85]
Iterated RFT − Static RFT	+7.21	[3.37, 11.54]
Epoch-refresh GRPO − no-refresh GRPO	+4.33	[0.48, 8.65]
Online GRPO − epoch-refresh GRPO	+2.40	[−2.40, 7.21]
Online GRPO − no-refresh GRPO	+6.73	[2.88, 11.06]
Online GRPO − Iterated RFT	+0.48	[−3.85, 4.81]

Table 6: Held-out accuracy with bootstrap confidence intervals.

Table 4: Per-relation counts for the held-out test set and the RL training set after enforcing disjoint relations.

Relation	test	train
director	0	180
screenwriter	0	163
genre	0	161
producer	0	114
author	0	113
composer	100	0
father	27	0
capital	19	0
capital of	18	0
place of birth	11	0
mother	8	0
occupation	8	0
religion	7	0
sport	5	0
country	5	0
Total	**208**	**731**

Table 5: RL training hyperparameters for the controlled knowledge-injection experiment.

Hyperparameter	Value
LoRA configuration	
LoRA rank	32
RL training	
Algorithm	GRPO
Batch size	16
Group size	16
Number of epochs	3
Sampling temperature	0.7
Evaluation temperature	0.0
LR scheduler	constant
Optimizer	Adam (β_1=0.9, β_2=0.95, ϵ=10^{-8})
Random seed	42
Max token	10,000 tokens

sample. We repeat this procedure for 100,000 samples using seed 20260825 and report the 2.5th and 97.5th percentiles of the resulting differences as the 95% paired-bootstrap interval.

B.6 Layerwise Similarity Metric: Formal Definition

This section gives the formal definition of the cosine similarity metric used in Section 5.1, including the set of source pairs over which inter-model and intra-model comparisons are computed.

Comparison Metric. For each layer $\ell \in \{1, \ldots, L\}$, we use cosine similarity, a measure of directional alignment, to define representation similarity:

$$\cos^{(a,b)}(\ell) = \frac{1}{N} \sum_{i=1}^{N} \frac{\mathbf{h}_\ell^{(a)}(i)^\top \mathbf{h}_\ell^{(b)}(i)}{\|\mathbf{h}_\ell^{(a)}(i)\|_2 \, \|\mathbf{h}_\ell^{(b)}(i)\|_2}. \quad (3)$$

Here, $\mathbf{h}_\ell^{(s)}(i)$ denotes the layer-ℓ hidden representation for probe i from a source s. The set of sources $\mathcal{S} = \{\mathrm{Q}^{\text{base}}, \mathrm{A}^{\text{base}}, \mathrm{Q}^{\text{post-trained}}, \mathrm{A}^{\text{post-trained}}\}$ includes representations for both the question (Q) and answer (A) components from the base and post-trained (instruct/reasoning/distilled) models. Pair (a, b) represents either inter-model (e.g., Q^{base} vs $\mathrm{Q}^{\text{post-trained}}$, A^{base} vs $\mathrm{A}^{\text{post-trained}}$) or intra-model comparisons (e.g., Q^{base} vs A^{base}, $\mathrm{Q}^{\text{post-trained}}$ vs $\mathrm{A}^{\text{post-trained}}$).

C Extended Experimental Results

C.1 Extended Model Performance Across Prompt Types

Tables 6–8 report majority vote and mean accuracy for all model pairs across the three prompt types on PopQA, IPC, and MedConceptsQA. Across all datasets, the instruct–reasoning gap consistently narrows as prompt complexity increases.

C.2 Reasoning Trajectories Analysis

Figures 20–23 show reasoning traces across prompts for DeepSeek-V3, Qwen3-30B-A3B-Instruct, Qwen3-235B-A22B-Instruct, Kimi-K2.5-Instruct and their respective reasoning models. On MedConceptsQA, DeepSeek-V3 and Qwen3-235B-A22B-Instruct both fail under direct QA and CoT but succeed under structured prompting via explicit hierarchical traversal. DeepSeek-R1 and Qwen3-235B-A22B-Think arrive at the correct answer spontaneously without any structural guid-

ance, demonstrating that reasoning models internalize the traversal policy that structured prompting supplies externally. Similar patterns hold on PopQA (Kimi-K2.5 pair) and IPC (Qwen3-30B-A3B pair), where the instruct model recovers only under structured prompting while its reasoning counterpart succeeds directly. Note that under direct QA, reasoning models output a short final answer, so we report their internal reasoning instead.

C.3 Additional Depth-Stratified Results on DBO-CAR

Figure 24 reports DeepSeek-V3 vs DeepSeek-R1 on DBO-CAR. We observe a directionally consistent crossover: V3 leads at shallow levels (Path-Match gap -0.083 at `depth`$= 2$) while R1 overtakes at the deepest level ($+0.054$ at `depth`$= 6$). Final-answer accuracy saturates across both models, only path matching score surfaces the depth-dependent divergence.

C.4 Additional Layerwise Cosine Similarity Results

Qwen2.5 Series. Tables 9 and 10 quantifies the layerwise cosine similarity findings from Section 5.1 on the `ICD9PROC` vocabulary, comparing Qwen2.5-32B (base) against its instruct, reasoning, and distilled variants, as well as similarity between question (Q) and answer (A) representations within each variant. We partition the 64 transformer layers into early (1–21), middle (22–43), and late (44–64) thirds, and report the mean similarity within each band for questions (Q) and answers (A) separately, along with the per-component overall mean and its minimum across layers. Across both tables, question representations diverge more than answer representations and the distilled variant diverges most, supporting our claim that post-training reshapes question processing rather than stored knowledge.
Mistral Series. Figure 19 shows the corresponding analysis for the Mistral-Small model family, comparing the base (-Base-2503), instruction-tuned (-Instruct-2503), and reasoning (Magistral-Small-2507) variants across MedConceptsQA vocabulary `ICD9PROC`.

C.5 Failure Modes of Distilled Models Under Structured Prompting

Table 11 reports additional results for the Qwen2.5-32B model series on MedConceptsQA, IPC, and PopQA, comparing the base, instruct, reasoning, and distilled variants across all three prompt types. Across datasets, the distilled model consistently underperforms the reasoning model and, in several cases, fails to match even the instruct or base variants, even under structured prompting. This shows the ineffectiveness of distillation on knowledge retrieval tasks.

Table 12 presents representative reasoning trajectories from DeepSeek-R1-Distill-Qwen2.5-32B for each identified failure mode under structured prompting. The first category is surface-level self-correction, where the model imitates backtracking (e.g., "Wait...") while remaining anchored to its initial hypothesis. The second category shows unsuccessful backtracking, where the model reconsiders multiple options but converges on the wrong one. This further supports our conclusion that distillation, lacking the trial-and-error exploration inherent to RL, fails to acquire the navigation skills necessary to retrieve the parametric knowledge.

D Experimental Artifacts and Responsible Use

D.1 Artifacts License and Intended Use

We use several public datasets, models, and analytic tools. PopQA is released under an MIT license. MedConceptsQA is distributed on Hugging Face under an Apache license 2.0. The DBpedia Ontology is a community-curated ontology distributed under Creative Commons Attribution-ShareAlike 3.0 License (CC BY-SA 3.0). The International Patent Classification (IPC) is provided by WIPO as a public classification system for organizing and searching patent documents. We use IPC Version 2026.01 only for research evaluation, we only find its python library under GNU GPL v3 License. Infini-gram is used to estimate corpus co-occurrence frequency for entity–answer pairs, it is under Apache license 2.0. Qwen open-weight models are under Apache-2.0-style licenses, DeepSeek-R1/V3 and Kimi models are released under the MIT License; we use them only for research evaluation and do not redistribute any model weights. We cite all original artifacts in the main paper. Our use of artifacts is intended for research on language model evaluation and analysis only.

The artifacts used and created cover several domains of factual knowledge. PopQA evaluates open-ended real world entity-relation recall; MedConceptsQA evaluates clinical code-description recall over medical ontologies; IPC-Lookup and IPC-CAR evaluate patent-code and patent-hierarchy re-

call; DBO-CAR evaluates hierarchy traversal over DBpedia ontology classes, including people, organizations, places, and species. All tasks are evaluated in English. The artifacts are used for studying factual recall and are not designed to represent demographic groups.

D.2 Model Size and Computational Budget

We evaluate models ranging from 8B to large-scale mixture-of-experts models, including Qwen2.5-32B, Qwen3-8B, Qwen3-30B-A3B, Qwen3-235B-A22B, DeepSeek-V3/V3.1/R1 with 671B parameters in total and 37B activated, and Kimi-K2/K2.5 with 1T parameters in total and 32B activated. We use around 100 GPU hours in total for running inference experiments, dataset construction, doing RL/SFT experiments. Experiments are done on serverless distributed GPU training, hence we do not have exact GPU type or hours used.

D.3 Use of AI assistants.

We used AI assistants for literature search and for improving grammar and clarity. Everything reviewed and verified by the authors.

Prompt — Entity and Relation Extraction from PopQA Questions

```
You are an information extraction expert. Given a factual question, identify:
1. The subject ENTITY the question is about (a named entity: person, place, work, etc.).
2. The RELATION (attribute or property) being asked about the entity.

Respond ONLY with a JSON object of the form:
{"entity": "<entity string>", "relation": "<relation string>"}

Use a short, lowercase, hyphen-or-space-separated noun phrase for the relation
(e.g. "composer", "genre", "place of birth", "occupation"). Use the entity
exactly as it appears in the question (preserve casing and spelling).

###
Example 1:
Question: Who was the composer of America?
Output: {"entity": "America", "relation": "composer"}
###

###
Example 2:
Question: What genre is Holiday?
Output: {"entity": "Holiday", "relation": "genre"}
###

###
Example 3:
Question: In what city was Albert Einstein born?
Output: {"entity": "Albert Einstein", "relation": "place of birth"}
###

###
Example 4:
Question: What is the capital of France?
Output: {"entity": "France", "relation": "capital"}
###

###
Example 5:
Question: What is George Orwell's occupation?
Output: {"entity": "George Orwell", "relation": "occupation"}
###

...

Now extract the entity and relation from the following question.
Output ONLY the JSON object, no additional text.

Question: {question}
Output:
```

Figure 14: Few-shot prompt used to extract the subject entity and relation from each PopQA question before Infini-gram co-occurrence probing.

Prompt Template 1: Final Answer Only

Answer the question. Output only the final answer, with no explanation.

[... Insert Question Text Here ...]

Answer (only the final answer):

Figure 15: Direct QA prompt for PopQA, IPC, and MedConceptsQA.

Prompt Template 2: CoT with Explanation

You are a *[medical / patent / knowledge]* research assistant. Read the following question carefully. Your task is to:

1. *MCQ only:* Answer with one of A/B/C/D, which corresponds to the four options. *PopQA:* Provide the correct answer.
2. For convenience, give the response in the format `Answer: ...`, with an additional explanation in the format `Explanation: ....`

[... Insert Question Text Here ...]

Respond in the following format:
Answer:
Explanation:

Figure 16: CoT prompt. The role label and answer placeholder adapt to the dataset: *medical research assistant* (MedConceptsQA), *patent research assistant* (IPC), or *research assistant* (PopQA). Here MCQ datasets refer to IPC and MedConceptsQA. The structural format is otherwise identical.

Prompt Template 3: Structured Recall

MCQ (IPC / MedConceptsQA): You are a *[medical / patent]* classification expert. For each option, first **recall the general category and structural breakdown of the *[medical / patent]* code**, then explain **why it might be wrong**. Finally pick the correct one.

[. . . Insert Question Text Here . . .]

Steps to follow:

1. Recall the general category and structural breakdown of the code.
2. Evaluate each option (A–D) briefly.
3. Choose the best option and justify.

Answer format:

```
Step 1:  . . .
Step 2A: . . .
Step 2B: . . .
Step 2C: . . .
Step 2D: . . .
Final Answer: [A/B/C/D] because . . .
```

Open QA (PopQA):

You are a knowledge assistant. Your task is to answer a factual question by first retrieving the relevant information from memory in a structured way.

[. . . Insert Question Text Here . . .]

Steps to follow:

1. Identify the type of fact being asked about (e.g., date, person, place, attribute, relation, event).
2. Recall the relevant fact or facts needed to answer the question.
3. Provide the final answer and justify it.

Answer format:

```
Step 1: . . .
Step 2: . . .
Step 3: . . .
Final answer: <short factual answer> because
. . .
```

Figure 17: Structured navigation prompt. For MCQ datasets (IPC, MedConceptsQA), the model performs hierarchical code recall followed by per-option elimination. For PopQA, the model identifies the fact type, recalls the relevant knowledge, and then states the final answer. Italicised labels in square brackets denote dataset-specific substitutions.

IPC-CAR Generation Prompt

Task: Generate IPC Common-Ancestor Dataset
Background: IPC Code Hierarchy. The International Patent Classification (IPC) is a tree-structured taxonomy. In the source PDF, each entry looks like:

```
1/00  Hand tools (edge trimmers for lawns A01G 3/06) [1, 2006.01]
1/02  • Spades; Shovels [1, 2006.01]
1/04  • • with teeth [1, 2006.01]
1/06  • Hoes; Hand cultivators [1, 2006.01]
1/08  • • with a single blade [1, 2006.01]
1/10  • • with two or more blades [1, 2006.01]
1/12  • • with blades provided with teeth [1, 2006.01]
1/14  • • with teeth only [1, 2006.01]
```

Parsing rules:

- The number of dots (•) before the description indicates the depth in the tree.
- A code's parent is the nearest preceding code with exactly one fewer dot.
- A code with zero dots (e.g., `1/00`) is a section header; its parent is the subclass (e.g., `A01B`, which is a more general sectional header immediately above it).
- Example: `1/04` (2 dots) → parent is `1/02` (1 dot, immediately above). `1/02` (1 dot) → parent is `1/00` (0 dots).

Goal. Build a multiple-choice dataset that tests an LLM's ability to recall the nearest common ancestor (NCA) of two IPC codes by traversing this hierarchy from parametric memory alone.
Question Format. Each question has 4 options (A–D), one correct answer, and a reasoning trace showing the two upward paths to the NCA. Example:

```
{
  "C07C_313_32_C07C_313_22_P2g": {
    "QUESTION": "Nearest common ancestor of C07C 313/32 and
                 C07C 313/22 is: A) C07C 313/08 B) C07C 313/18
                 C) C07C 313/00 D) C07C 313/12",
    "REASONING": {
      "H_PATHS": [
        "C07C 313/32 -> C07C 313/30 -> C07C 313/26 -> C07C 313/18",
        "C07C 313/22 -> C07C 313/18"
      ]
    },
    "ANSWER": "B",
    "QUESTION_TYPE": "P2g_Common_ancestor",
    "PATENT_CODE": "C07C 313/32, C07C 313/22",
    "TASK_COMPLEXITY": "MM",
    "max_levels": 3,
    "add_levels": 4
  }
}
```

Dataset Specification.

- **Levels:** Generate questions for traversal depths of 2, 3, 4, 5, and 6 levels, where the level equals the total number of edges traversed across both upward paths, excluding the two leaf nodes shown in the question.
- **Per-level count:** 20 questions per level.

Deliverable. Write Python code that:

1. Parses the IPC PDF (or its text extraction) into a tree using the dot-depth rules above.
2. For each (category, level) pair, samples leaf-code pairs whose NCA satisfies Constraint 1.
3. Constructs 3 distractors per question.
4. Emits the final dataset as JSON in the format shown in the example.

Figure 18: The LLM prompt for generating IPC-CAR dataset.

Table 6: Performance comparison of Instruct vs. Reasoning models on PopQA. The first column indicates the dataset. Models are evaluated across three prompt templates (QA, CoT, Structured). Metrics shown are majority voting accuracy (Maj. Vote Acc.) and mean accuracy (Mean Acc.). Mean accuracy is reported as Mean Acc. $_{\text{(Std.)}}$, with the standard deviation in subscripted parentheses. For each model pair, a Δ row shows the gap from the reasoning model for both Maj. Acc. (orange) and Mean Acc. (yellow). Bold values indicate the best performance within each model pair. Δ values are highlighted, with darker shades indicating larger gaps.

Dataset	Model	Model Type	Maj. Vote Acc.			Mean Acc.$_{\text{(Std.)}}$		
			QA	CoT	Structured	QA	CoT	Structured
PopQA	Qwen2.5-32B	Instruct	0.252	0.288	0.310	0.251$_{(.0009)}$	0.291$_{(.0018)}$	0.310$_{(.0003)}$
		Reasoning	**0.386**	**0.380**	**0.375**	**0.386$_{(.0010)}$**	**0.379$_{(.0014)}$**	**0.378$_{(.0030)}$**
		Δ	**+0.134**	**+0.092**	**+0.065**	**+0.135**	**+0.088**	**+0.068**
	Qwen3-8B	Instruct	0.205	0.252	**0.262**	0.204$_{(.0004)}$	0.251$_{(.0013)}$	0.263$_{(.0016)}$
		Reasoning	**0.237**	**0.267**	**0.262**	**0.234$_{(.0042)}$**	**0.267$_{(.0016)}$**	**0.265$_{(.0015)}$**
		Δ	**+0.032**	**+0.015**	**+0.000**	**+0.029**	**+0.016**	**+0.002**
	Qwen3-30B-A3B	Instruct	0.231	0.279	0.302	0.226$_{(.0010)}$	0.277$_{(.0026)}$	0.304$_{(.0014)}$
		Reasoning	**0.288**	**0.317**	**0.312**	**0.283$_{(.0033)}$**	**0.318$_{(.0029)}$**	**0.313$_{(.0011)}$**
		Δ	**+0.058**	**+0.037**	**+0.010**	**+0.057**	**+0.041**	**+0.010**
	Qwen3-235B-A22B	Instruct	0.429	0.491	0.541	0.426$_{(.0005)}$	0.490$_{(.0010)}$	0.542$_{(.0007)}$
		Reasoning	**0.494**	**0.562**	**0.567**	**0.490$_{(.0020)}$**	**0.566$_{(.0027)}$**	**0.566$_{(.0021)}$**
		Δ	**+0.065**	**+0.071**	**+0.026**	**+0.064**	**+0.076**	**+0.024**
	DeepSeek-V3	Instruct	0.430	0.505	**0.569**	0.422$_{(.0030)}$	0.504$_{(.0023)}$	0.569$_{(.0022)}$
		Reasoning	**0.513**	**0.557**	**0.569**	**0.502$_{(.0023)}$**	**0.559$_{(.0027)}$**	**0.570$_{(.0020)}$**
		Δ	**+0.083**	**+0.052**	**+0.000**	**+0.080**	**+0.055**	**+0.001**
	DeepSeek-V3.1	Instruct	**0.429**	0.506	0.565	**0.424$_{(.0013)}$**	0.506$_{(.0016)}$	**0.566$_{(.0017)}$**
		Reasoning	**0.429**	**0.508**	**0.567**	**0.424$_{(.0015)}$**	**0.511$_{(.0031)}$**	0.565$_{(.0019)}$
		Δ	**+0.003**	**+0.002**	**+0.002**	**+0.001**	**+0.004**	**-0.001**
	Kimi-K2	Instruct	0.507	0.533	0.566	0.503$_{(.0033)}$	0.536$_{(.0023)}$	0.570$_{(.0040)}$
		Reasoning	**0.580**	**0.662**	**0.652**	**0.574$_{(.0018)}$**	**0.663$_{(.0007)}$**	**0.653$_{(.0021)}$**
		Δ	**+0.073**	**+0.129**	**+0.086**	**+0.071**	**+0.127**	**+0.083**
	Kimi-K2.5	Instruct	0.507	0.573	0.649	0.503$_{(.0011)}$	0.572$_{(.0009)}$	0.647$_{(.0012)}$
		Reasoning	**0.597**	**0.700**	**0.682**	**0.595$_{(.0021)}$**	**0.697$_{(.0026)}$**	**0.685$_{(.0020)}$**
		Δ	**+0.100**	**+0.127**	**+0.033**	**+0.107**	**+0.125**	**+0.038**

Table 7: Performance comparison of Instruct vs. Reasoning models on IPC Codes.

Dataset	Model	Model Type	Maj. Vote Acc.			Mean Acc.$_{\text{(Std.)}}$		
			QA	CoT	Structured	QA	CoT	Structured
IPC Codes	Qwen2.5-32B	Instruct	0.300	**0.370**	**0.400**	0.313$_{(.017)}$	**0.380$_{(.014)}$**	**0.383$_{(.048)}$**
		Reasoning	**0.350**	0.360	0.360	**0.370$_{(.014)}$**	0.370$_{(.016)}$	0.353$_{(.017)}$
		Δ	**+0.050**	**-0.010**	**-0.040**	**+0.057**	**-0.010**	**-0.030**
	Qwen3-8B	Instruct	0.266	0.277	0.372	0.280$_{(.010)}$	0.284$_{(.005)}$	0.340$_{(.048)}$
		Reasoning	**0.298**	**0.394**	**0.394**	**0.316$_{(.010)}$**	**0.344$_{(.010)}$**	**0.372$_{(.023)}$**
		Δ	**+0.032**	**+0.117**	**+0.022**	**+0.036**	**+0.060**	**+0.032**
	Qwen3-30B-A3B	Instruct	0.309	0.319	0.436	0.312$_{(.005)}$	0.312$_{(.013)}$	**0.411$_{(.027)}$**
		Reasoning	**0.426**	**0.372**	**0.457**	**0.394$_{(.009)}$**	**0.383$_{(.031)}$**	**0.411$_{(.020)}$**
		Δ	**+0.117**	**+0.053**	**+0.021**	**+0.082**	**+0.071**	**+0.000**
	Qwen3-235B-A22B	Instruct	0.372	0.383	**0.596**	0.351$_{(.015)}$	0.379$_{(.005)}$	**0.550$_{(.027)}$**
		Reasoning	**0.457**	**0.404**	0.574	**0.454$_{(.018)}$**	**0.422$_{(.028)}$**	0.518$_{(.022)}$
		Δ	**+0.085**	**+0.021**	**-0.022**	**+0.103**	**+0.043**	**-0.032**
	DeepSeek-V3	Instruct	0.415	0.596	0.553	0.415$_{(.009)}$	0.564$_{(.057)}$	0.518$_{(.020)}$
		Reasoning	**0.638**	**0.670**	**0.574**	**0.592$_{(.039)}$**	**0.628$_{(.015)}$**	**0.564$_{(.035)}$**
		Δ	**+0.223**	**+0.074**	**+0.021**	**+0.177**	**+0.064**	**+0.046**
	DeepSeek-V3.1	Instruct	0.468	0.500	**0.606**	0.447$_{(.040)}$	0.482$_{(.027)}$	**0.592$_{(.027)}$**
		Reasoning	**0.479**	**0.511**	0.585	**0.454$_{(.013)}$**	**0.518$_{(.013)}$**	0.553$_{(.017)}$
		Δ	**+0.011**	**+0.011**	**-0.021**	**+0.007**	**+0.036**	**-0.039**
	Kimi-K2	Instruct	0.511	0.521	0.543	0.511$_{(.000)}$	0.514$_{(.010)}$	**0.560$_{(.028)}$**
		Reasoning	**0.575**	**0.585**	**0.564**	**0.543$_{(.009)}$**	**0.564$_{(.009)}$**	0.525$_{(.013)}$
		Δ	**+0.064**	**+0.064**	**+0.021**	**+0.032**	**+0.050**	**-0.035**
	Kimi-K2.5	Instruct	0.521	0.500	**0.553**	0.482$_{(.010)}$	0.514$_{(.033)}$	0.543$_{(.031)}$
		Reasoning	**0.574**	**0.574**	**0.585**	**0.557$_{(.018)}$**	**0.571$_{(.018)}$**	**0.553$_{(.035)}$**
		Δ	**+0.053**	**+0.074**	**+0.032**	**+0.075**	**+0.057**	**+0.010**

Table 8: Performance comparison of Instruct vs. Reasoning models on MedConceptsQA.

Dataset	Model	Model Type	Maj. Vote Acc.			Mean Acc.$_{(Std.)}$		
			QA	CoT	Structured	QA	CoT	Structured
MedConceptsQA	**Qwen2.5-32B**	Instruct	0.379	0.475	0.469	$0.371_{(.012)}$	$0.449_{(.010)}$	$0.454_{(.007)}$
		Reasoning	**0.482**	**0.513**	**0.505**	$\mathbf{0.470_{(.012)}}$	$\mathbf{0.487_{(.009)}}$	$\mathbf{0.481_{(.005)}}$
		Δ	**+0.103**	**+0.038**	**+0.036**	**+0.099**	**+0.038**	**+0.027**
	Qwen3-8B	Instruct	0.338	0.337	0.406	$0.339_{(.007)}$	$0.335_{(.005)}$	$0.386_{(.011)}$
		Reasoning	**0.400**	**0.424**	**0.421**	$\mathbf{0.386_{(.012)}}$	$\mathbf{0.410_{(.003)}}$	$\mathbf{0.407_{(.009)}}$
		Δ	**+0.062**	**+0.087**	**+0.015**	**+0.047**	**+0.075**	**+0.021**
	Qwen3-30B-A3B	Instruct	0.398	0.404	**0.489**	$0.399_{(.005)}$	$0.399_{(.005)}$	$\mathbf{0.485_{(.006)}}$
		Reasoning	**0.498**	**0.506**	0.476	$\mathbf{0.494_{(.007)}}$	$\mathbf{0.501_{(.014)}}$	$0.477_{(.008)}$
		Δ	**+0.100**	**+0.102**	**-0.013**	**+0.095**	**+0.102**	**-0.008**
	Qwen3-235B-A22B	Instruct	0.542	0.548	**0.631**	$0.503_{(.004)}$	$0.528_{(.005)}$	$\mathbf{0.589_{(.007)}}$
		Reasoning	**0.641**	**0.656**	0.580	$\mathbf{0.599_{(.003)}}$	$\mathbf{0.617_{(.003)}}$	$0.554_{(.008)}$
		Δ	**+0.099**	**+0.108**	**-0.051**	**+0.096**	**+0.089**	**-0.035**
	DeepSeek-V3	Instruct	0.541	0.632	0.717	$0.551_{(.014)}$	$0.636_{(.049)}$	$0.701_{(.026)}$
		Reasoning	**0.778**	**0.790**	**0.792**	$\mathbf{0.830_{(.006)}}$	$\mathbf{0.774_{(.013)}}$	$\mathbf{0.775_{(.026)}}$
		Δ	**+0.237**	**+0.158**	**+0.075**	**+0.279**	**+0.138**	**+0.074**
	DeepSeek-V3.1	Instruct	0.526	0.595	**0.785**	$0.520_{(.008)}$	$0.593_{(.009)}$	$0.763_{(.004)}$
		Reasoning	**0.794**	**0.805**	0.774	$\mathbf{0.785}_{(.011)}$	$\mathbf{0.772}_{(.008)}$	$\mathbf{0.772}_{(.008)}$
		Δ	**+0.268**	**+0.210**	**-0.011**	**+0.265**	**+0.179**	**+0.009**
	Kimi-K2	Instruct	0.595	0.634	0.750	$0.593_{(.003)}$	$0.625_{(.003)}$	$0.735_{(.011)}$
		Reasoning	**0.834**	**0.829**	**0.830**	$\mathbf{0.809_{(.001)}}$	$\mathbf{0.810_{(.006)}}$	$\mathbf{0.799_{(.007)}}$
		Δ	**+0.239**	**+0.195**	**+0.080**	**+0.216**	**+0.185**	**+0.064**
	Kimi-K2.5	Instruct	0.699	0.734	0.875	$0.690_{(.006)}$	$0.727_{(.008)}$	$0.864_{(.008)}$
		Reasoning	**0.905**	**0.915**	**0.911**	$\mathbf{0.881_{(.008)}}$	$\mathbf{0.895_{(.008)}}$	$\mathbf{0.895_{(.008)}}$
		Δ	**+0.206**	**+0.181**	**+0.036**	**+0.191**	**+0.168**	**+0.031**

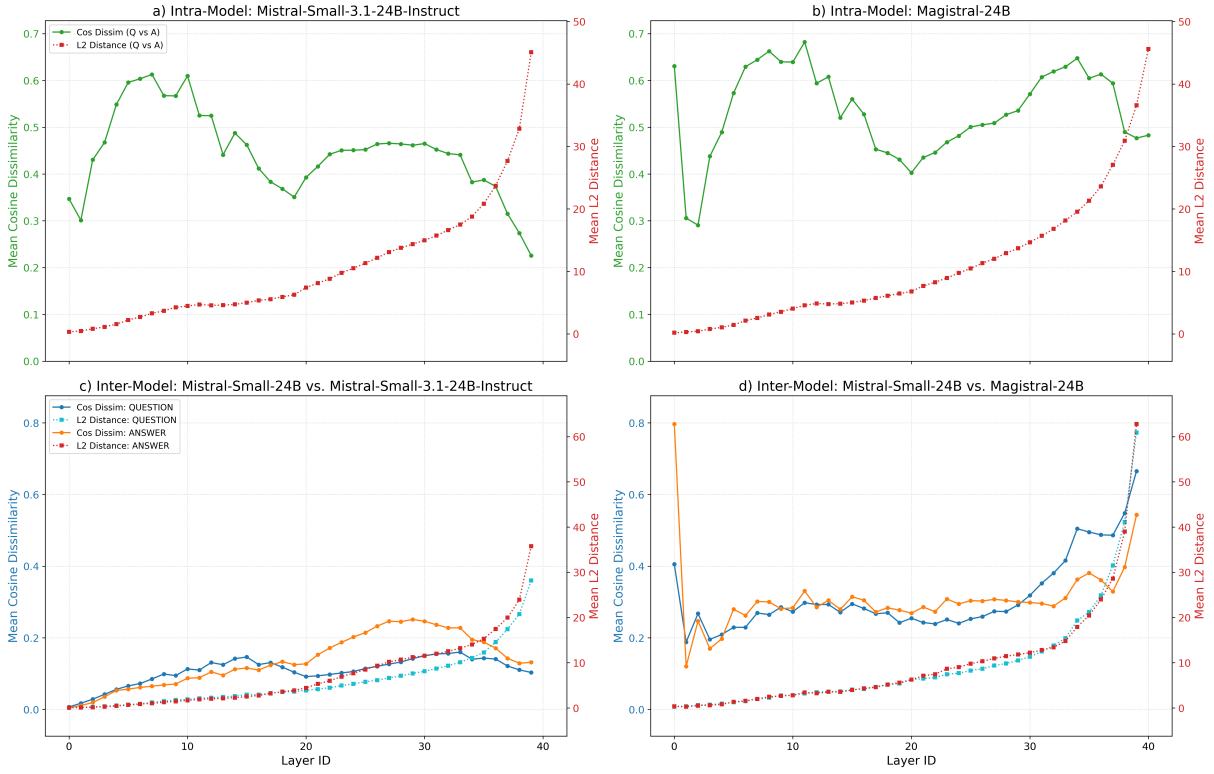


Figure 19: Layerwise cosine similarity of question (Q) and answer (A) representations on ICD9PROC for Mistral-Small-24B family.

Figure 20: Reasoning trajectories for Kimi-K2.5 (non-thinking) and its reasoning variant across prompt templates on PopQA.

Question

`What is the capital of Poni Province?`

`Ground Truth: Gaoua` ✓

Kimi-K2.5-NonThink — Direct QA

Boulounga ×

Kimi-K2.5-NonThink — CoT

Answer: Kedougou
Explanation: The capital city of Poni Province is Kedougou (also spelled Kédougou). The province shares its name with the Poni River and is known for its agricultural activities and natural landscapes. ×

Kimi-K2.5-NonThink — Structured Prompting

Step 1: This is a geography question asking about the capital of a place (Poni Province).

Step 2: Poni Province is a province in Burkina Faso, a country in West Africa. The capital of Poni Province is Gaoua.

Step 3: Gaoua serves as the administrative center of Poni Province in southwestern Burkina Faso.

Final Answer: Gaoua ✓

Kimi-K2.5-Think — Direct QA

Reasoning: Poni Province is one of the 45 provinces of Burkina Faso, located in the Sud-Ouest Region. The capital is indeed Gaoua.

Final Answer: Gaoua ✓

Figure 21: Reasoning trajectories for Qwen3-30B-A3B (non-thinking) and its reasoning variant across prompt templates on IPC.

Question

What does the patent code B01F 27/1145 in the International Patent Classification describe?

A. Centrifugal stirrers, i.e. stirrers with radial outlets; Stirrers of the turbine type, e.g. with means to guide the flow
B. Helically shaped stirrers, ribbon shaped with an open space between the helical ribbon flight and the rotating axis
C. Propeller-shaped stirrers for producing an axial flow, e.g. shaped like a ship or aircraft propeller
D. Stirrers shaped as cylinders, balls or rollers

Ground Truth: B ✓

Qwen3-30B-A3B-NonThink — Direct QA

A ×

Qwen3-30B-A3B-NonThink — CoT

Answer: A

Explanation: The patent code B01F 27/1145 refers to centrifugal stirrers with radial outlets and stirrers of the turbine type, which may include means to guide the flow. This classification covers stirrers that operate by creating a radial flow, commonly seen in industrial mixing applications. ×

Qwen3-30B-A3B-NonThink — Structured Prompting

Step 1: B01 covers physical or chemical processes or apparatus; B01F covers mixing; B01F 27/1145 denotes helically shaped or ribbon shaped stirrers with an open space between the helical ribbon flight and the rotating axis.

Step 2A: Option A (centrifugal/turbine-type stirrers with radial outlets) falls under different subclasses such as B01F 27/111. **Incorrect.**

Step 2B: Option B (helically shaped, ribbon-type stirrers with open space between ribbon flight and axis) is a precise match for B01F 27/1145. **Correct.**

Step 2C: Option C (propeller-shaped stirrers producing axial flow) is typically classified under axial-flow impeller subclasses, not B01F 27/1145. **Incorrect.**

Step 2D: Option D (stirrers shaped as cylinders, balls, or rollers) does not correspond to helical ribbon stirrers and is covered by code such as B01F 27/116. **Incorrect.**

Final Answer: B ✓

Qwen3-30B-A3B-Think — Direct QA

Reasoning: B01F relates to mixing apparatus. B01F 27 refers to mixers with rotary devices. The subgroup 27/1145 specifically covers helical ribbon stirrers—ribbon-shaped with an open space between the ribbon flight and the rotating axis. Options A (centrifugal/turbine), C (propeller/axial), and D (cylinders/balls/rollers) each describe distinct stirrer types classified elsewhere. Option B is therefore the correct match.

Final Answer: B ✓

Figure 22: Reasoning trajectories for Qwen3-235B-A22B-2507-Instruct and its reasoning model across prompt templates on MedConceptsQA.

Question

What is the description of the medical code 66.97 in ICD9PROC?

A. Bilateral partial salpingectomy, not otherwise specified
B. Salpingotomy and salpingostomy
C. Other operations on female genital organs
D. Burying of fimbriae in uterine wall ✓

Qwen3-235B-A22B-2507-Instruct — Direct QA

The code **66.97** in **ICD-9-PROC** stands for "Other operations on female genital organs." This code is used when a procedure on the female genital organs is performed that is not specifically listed elsewhere in the coding system. Therefore, the correct answer is

$$\boxed{C}$$

Qwen3-235B-A22B-2507-Instruct — CoT

Answer: C
Explanation: ICD-9-CM procedure code **66.97** falls under the category "Other operations on female genital organs." It's a residual category for procedures not otherwise specified in the coding system. This makes option C the accurate description.

Option Analysis:

The other options are incorrect: option A mentions bilateral partial salpingectomy (e.g., 66.6x) which would be a different code range. B refers to salpingotomy and salpingostomy procedures (66.01, 66.02), which have their own specific codes. Option D describes a very specific technique of burying fimbriae in the uterine wall, which is not a standard procedure or code listed under 66.97.

Qwen3-235B-A22B-2507-Instruct — Structured Prompting

The medical code 66.97 belongs to the ICD-9-CM (International Classification of Diseases, 9th Revision, Clinical Modification) procedural coding system. It falls under the broad category of "Operations on the female genital organs" (ICD-9-CM codes 65–69). Chapter 66 refers to "Other operations on the fallopian tubes and broad ligament.", procedure code 66.97 is "Burying of fimbriae in uterine wall".

Option Analysis:

A. Bilateral partial salpingectomy, not otherwise specified — incorrect, typically coded under 66.6x (e.g., 66.62 for salpingectomy related procedures)

B. Salpingotomy and salpingostomy — incorrect, specific procedures like salpingotomy (an incision into the fallopian tube, often for ectopic pregnancy) usually coded as 66.01 and 66.02.

C. Other operations on female genital organs — incorrect, very broad category and refers to codes in the 69.x range.

D. Burying of fimbriae in uterine wall — correct, classified under ICD-9-CM code 66.97.

Final Answer: D

Qwen3-235B-A22B-2507-Think — Direct QA

Code Structure: In ICD-9-CM, codes in the 65-71 range are for operations on the female genital organs. Within this chapter, code 66 is for "Operations on fallopian tubes and broad ligaments." and 66.9 is "Other operations on fallopian tubes." 66.97 is "Burying of fimbriae in uterine wall."

Option Analysis:

A. Bilateral partial salpingectomy, not otherwise specified — incorrect, might fall under 66.62.

B. Salpingotomy and salpingostomy — incorrect. Salpingotomy is often 66.02 or included in 66.0, but not specifically 66.97.

C. Other operations on fallopian tubes — incorrect, C is a distractor because "other operations" might be misinterpreted, but 66.97 is specific. Code 66.9 is "Other operations on fallopian tubes," and 66.97 is a specific type under that.

Final Answer: D — code **66.97** is "Burying of fimbriae in uterine wall", where the fimbriated end of the fallopian tube is buried into the uterine wall to prevent fluid accumulation or for sterilization.

Figure 23: Reasoning trajectories for DeepSeek-V3 and DeepSeek-R1 across prompt templates on MedConceptsQA.

Question

What is the description of the medical code 57.95 in ICD9PROC?

A. Other procedures for relief of elevated intraocular pressure
B. Replacement of indwelling urinary catheter ✓
C. Arthrodesis of other joint
D. Biopsy of heart

DeepSeek-V3 — Direct QA

A

DeepSeek-V3 — CoT

Answer: A **Explanation:** The medical code **57.95** in **IPC-9-PROC** corresponds to "other procedures for relief of elevated intraocular pressure." This code is used to classify surgical or procedural interventions aimed at reducing high pressure within the eye, such as in glaucoma treatment. The other options (b, c, d) are unrelated to this specific procedure.

DeepSeek-V3 — Structured Prompting

Here is the breakdown for **IPC-9-PROC procedure code 57.95**:

Code Structure: Category 57 refers to "Other operations on the bladder" in ICD-9-PROC Volume 3. Subcode .95 indicates "Replacement of indwelling urinary catheter."

Option Analysis:

A. Other procedures for relief of elevated intraocular pressure — incorrect. This falls under ophthalmology (codes in the **12–14** range), not bladder procedures.

B. Replacement of indwelling urinary catheter — correct. Matches the **57.95** code description for bladder-related catheter replacement.

C. Arthrodesis of other joint — incorrect. Arthrodesis is a musculoskeletal procedure (codes in the **81** range).

D. Biopsy of heart — incorrect. Heart biopsies are cardiovascular procedures (codes in the **37** range).

Final Answer: B — code **57.95** specifically refers to the replacement of an indwelling urinary catheter, a bladder-related procedure.

DeepSeek-R1 — Direct QA

Code Structure: Code 57.95 is an ICD-9-PROC procedure code. In ICD-9-PROC Volume 3, Chapter 11 covers procedures on the urinary system (codes 55–59). Within this chapter, category 57 and subcode 57.9 refer to "Other operations on bladder," and subcode 57.95 is "Replacement of indwelling urinary catheter."

Option Analysis:

A. Other procedures for relief of elevated intraocular pressure — incorrect. This is an ophthalmologic procedure falling under a different system entirely.

B. Replacement of indwelling urinary catheter — correct. Directly matches **57.95** under the bladder operations category.

C. Arthrodesis of other joint — incorrect. Arthrodesis is a musculoskeletal procedure (codes in the **80s** range).

D. Biopsy of heart — incorrect. Heart biopsies are cardiovascular procedures (codes in the **35–39** range).

Final Answer: B — code **57.95** specifically refers to the replacement of an indwelling urinary catheter, a bladder-related procedure.

Table 9: Layerwise mean cosine similarity between Qwen2.5-32B (base) and its post-trained variants on ICD9PROC question (Q) and answer (A) representations, averaged over $N=100$ probes. Mean cosine similarity is reported for early (1–21), middle (22–43), and late (44–64) thirds. Lower similarity indicates greater divergence from the base model. Across all three variants, question representations diverge more than answer representations, and the distilled model diverges most.

Variant	Comp.	Early	Middle	Late	Overall	Min	Min Layer
Instruct	Q	0.990	0.970	0.938	0.966	0.729	64
	A	0.994	0.976	0.958	0.976	0.871	64
Reasoning	Q	0.982	0.942	0.923	0.950	0.810	64
	A	0.987	0.951	0.944	0.961	0.876	64
Distilled	Q	0.911	0.718	0.789	0.808	0.634	36
	A	0.955	0.864	0.854	0.892	0.778	64

Table 10: Within-model layerwise cosine similarity between question (Q) and answer (A) representations on ICD9PROC, averaged over $N=100$ probes. Higher similarity indicates that questions and answers share representational structure at that layer. Across all models, Q and A representations diverge substantially after the embedding layer and the distilled model's late layers show the largest divergence.

Variant	Early (1–21)	Middle (22–43)	Late (44–64)	Overall	Min	Min Layer
Instruct	0.276	0.393	0.325	0.333	0.148	63
Reasoning	0.285	0.410	0.310	0.336	0.161	63
Distilled	0.317	0.363	0.203	0.295	0.091	63

Table 11: Qwen2.5-32B model series accuracy on MedConceptsQA, IPC and PopQA. Mean Acc. shown as $\text{mean}_{(\text{std})}$.

		Maj. Vote Acc.			Mean Acc.(Std.)		
Dataset	**Model**	**QA**	**CoT**	**Structured**	**QA**	**CoT**	**Structured**
MedConceptsQA	Base	0.221	0.332	0.404	$0.219_{(.012)}$	$0.260_{(.071)}$	$0.372_{(.007)}$
	Distilled	0.375	0.380	0.447	$0.351_{(.009)}$	$0.369_{(.005)}$	$0.420_{(.002)}$
	Instruct	0.379	0.475	0.469	$0.371_{(.012)}$	$0.449_{(.010)}$	$0.454_{(.007)}$
	Reasoning	**0.482**	**0.513**	**0.505**	$\mathbf{0.470}_{(.012)}$	$\mathbf{0.487}_{(.009)}$	$\mathbf{0.481}_{(.005)}$
IPC Codes	Base	**0.350**	0.320	0.390	$0.310_{(.065)}$	$0.270_{(.008)}$	$0.370_{(.071)}$
	Distilled	0.340	0.340	0.330	$0.369_{(.050)}$	$0.350_{(.045)}$	$0.330_{(.014)}$
	Instruct	0.300	**0.370**	**0.400**	$0.313_{(.017)}$	$\mathbf{0.380}_{(.014)}$	$\mathbf{0.383}_{(.048)}$
	Reasoning	**0.350**	0.360	0.360	$\mathbf{0.370}_{(.014)}$	$0.370_{(.016)}$	$0.353_{(.017)}$
PopQA	Base	0.281	0.277	0.299	$0.280_{(.007)}$	$0.279_{(.002)}$	$0.303_{(.003)}$
	Distilled	0.332	0.334	0.328	$0.334_{(.021)}$	$0.334_{(.017)}$	$0.325_{(.022)}$
	Instruct	0.302	0.289	0.310	$0.297_{(.003)}$	$0.291_{(.002)}$	$0.310_{(.000)}$
	Reasoning	**0.386**	**0.380**	**0.375**	$\mathbf{0.386}_{(.010)}$	$\mathbf{0.379}_{(.014)}$	$\mathbf{0.378}_{(.030)}$

Table 12: Failure mode categorization from 40 incorrect responses of `DeepSeek-R1-Distill-Qwen2.5-32B` under structured prompting.

#	Failure Mode Description	Redacted Reasoning Trajectory	Rate (%)
1	Surface-level self-correction: mimics self-correction patterns using phrases such as "Wait…" or "Alternatively…", but remains trapped in the initial wrong hypothesis. It performs pattern-matching against its first committed option to eliminate other options rather than genuinely revisiting alternative hypotheses.	The **ICD-9-PROC** code **00.09** falls under the "Therapeutic Ultrasound", a part of the "Radiology" section. This code refers to other types of therapeutic ultrasound not classified elsewhere. Option A relates to gastrointestinal procedures, specifically endoscopic treatments. **It does not involve ultrasound, which is the focus of code 00.09, so it is incorrect.** [...] **Wait, but just to double-check, what's the exact description of 00.09?** I think it's "Other therapeutic ultrasound procedures." **Option C concerns prosthetics or orthotics, which are unrelated to therapeutic ultrasound, so it is incorrect.** [...] **Final Answer**: B, because code 00.09 specifically refers to other therapeutic ultrasound procedures.	80
2	Backtracks, jumps back and forth between multiple uncertain options before ultimately landing on the wrong one.	[...] I know that 77.24 is a code for "Removal of vaginal packing" or "Removal of intravenous catheter". **Wait, no—77.24 is actually for "Removal of vaginal packing."** But I need to make sure. **Alternatively, perhaps it's a biopsy**. In ICD-9PROC, codes for biopsies are often in the 28–29 series, but for specific areas, they might be under the body system. **Wait, maybe 77.2x is for procedures on the Bartholin's gland**. [...] B is the correct answer and 77.24 is a removal of a vaginal packing.	20

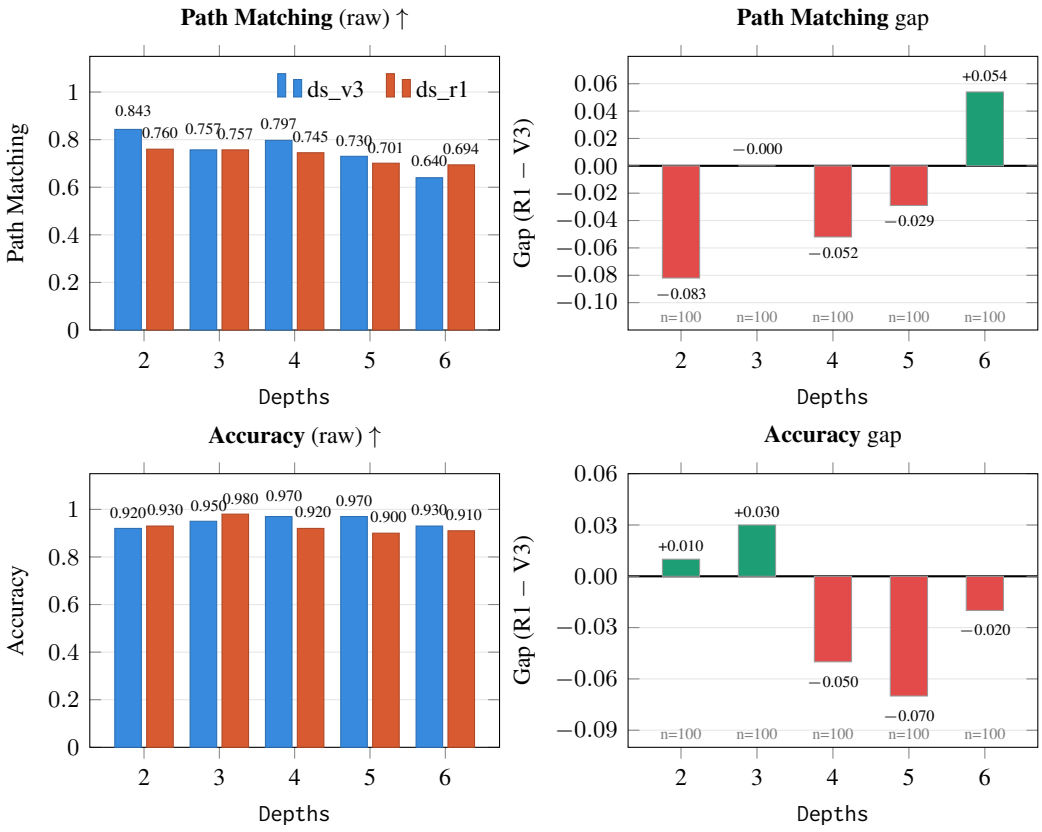


Figure 24: Path matching score and final-answer accuracy of DeepSeek-V3 (Instruct) versus DeepSeek-R1 (Reasoning) on DBO-CAR, stratified by `depths`.