NON-VACUOUS GENERALIZATION BOUNDS FOR REINFORCEMENT LEARNING WITH VERIFIABLE REWARDS

Anonymous authors
Paper under double-blind review

ABSTRACT

While reinforcement learning with verifiable rewards (RLVR) is widely used to improve the reasoning capabilities of large language models (LLMs), the generalizability of the resulting models remains poorly understood. In this work, we establish the first non-vacuous generalization bounds for parameter-efficient RLVR fine-tuning at the billion-parameter scale. Our approach adapts PAC-Bayes compression bounds to this setting and addresses the inherent stochasticity of token generation by applying the Gumbel-max reparameterization trick. To obtain non-vacuous bounds, we develop Progressive RLVR, a pipeline combining RLVR with on-policy distillation, TinyLoRA, and quantization, which achieves high empirical accuracy and high compressibility. Progressive RLVR retains 84–97% performance of standard LoRA fine-tuning while producing learned weight updates that are 14,796$\times$ more compressible. We show that this framework yields non-vacuous generalization bounds in four domains: math, programming, general-knowledge reasoning, and text-to-SQL. Our bounds exceed the accuracy of the base model by 9–51% and lie within 6–11% of the accuracy of the fine-tuned models.

1 INTRODUCTION

Reinforcement learning with verifiable rewards (RLVR) has emerged as a dominant paradigm for fine-tuning large language models in reasoning-intensive domains (Guo et al., 2025; Yang et al., 2025; Team et al., 2025; Chen et al., 2025), including mathematical problem-solving (Shao et al., 2024) and programming (Guo et al., 2025; Wei et al., 2025). However, recent work has shown that models fine-tuned via RLVR exhibit limited generalizability even for in-domain tasks (Hu et al., 2025), and are prone to diversity collapse and other pathological behaviors (Yue et al., 2025; Nguyen et al., 2025). These analyses suggest that RLVR specializes models to their training examples rather than enabling generalizable reasoning capabilities (Hu et al., 2025; Nguyen et al., 2025). Consequently, understanding the generalizability of RLVR models is critical for downstream deployment, particularly in high-stakes domains (Hager et al., 2024).

Unfortunately, verifying generalizability for RLVR is theoretically challenging. Existing techniques are either inapplicable or yield vacuous bounds. Classic complexity measures, such as Vapnik-Chervonenkis (VC) dimension (Blumer et al., 1989) and Rademacher complexity (Bartlett & Mendelson, 2002; Wang et al., 2019), scale with parameter counts and often lead to trivial, uninformative bounds for modern large language models (LLMs).

While prior literature has developed the compression-based Probably Approximately Correct Bayesian (PAC-Bayes) framework to compute non-vacuous generalization bounds for LLM pre-training (Lotfi et al., 2023), extending these guarantees to RLVR introduces several challenges. Unlike the pre-training setting of Lotfi et al. (2023), where the negative log-likelihood loss is deterministic for each input sequence given a fixed hypothesis, RLVR rewards are defined over stochastically generated sequences whose support is combinatorial in sequence length and vocabulary size. This makes the empirical reward intractable and introduces a second source of randomness beyond the training data distribution. Together, these make the existing PAC-Bayes bounds (Rodriguez-Galvez et al., 2024; Lotfi et al., 2023; 2022) inapplicable.

In this work, we resolve these theoretical bottlenecks and develop the first framework that yields non-vacuous generalization bounds for LLMs post-trained via parameter-efficient RLVR.

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

To establish generalization bounds for RLVR, we handle the stochasticity of the token generation process by decomposing the generalization gap into an extrinsic component over the input data distribution and an intrinsic component over the model's own generation process. To tackle the combinatorially large support of generated sequences, we apply a Gumbel-max reparameterization trick (Wang et al., 2019; Jang et al., 2016; Gumbel, 1954) to represent token generation as a deterministic function of the input sequence, random noise, and hypothesis, enabling the compression-based PAC-Bayes arguments (Lotfi et al., 2022) on stochastic token generation.

Since the compression-based PAC-Bayes bound scales with the description length of the weight updates, obtaining a non-vacuous bound requires aggressive compression of learned weight updates. To address this challenge, we develop *Progressive RLVR*, a post-training pipeline that achieves aggressive compression while preserving high training performance. We first train a strong teacher model using RLVR and then use on-policy distillation (OPD) (Agarwal et al., 2024) to compress the capability of the teacher model into a student model with fewer trainable parameters. During OPD, we apply TinyLoRA (Morris et al., 2026), which reparameterizes the weight updates into as few as one trainable scalar during training. Empirically, Progressive RLVR achieves 10% higher training rewards (Figure 2) and produces 15% tighter bounds than direct RLVR with TinyLoRA (§ 5.2), due to the dense token-level supervision that OPD provides.

In our experiments, we achieve *the first non-vacuous generalization bounds for RLVR* across four real-world domains, including mathematical problem-solving, programming, general-knowledge reasoning, and text-to-SQL translation. Across four domains, our generalization bounds exceed the accuracy of the base model by 9–51%, and are within 6–11% of the empirical accuracy of the fine-tuned models. Furthermore, Progressive RLVR achieves high accuracy with high compression rates, recovering 84–97% of the teacher models' accuracy, while being 14,796$\times$ more compressible than the corresponding teacher model on average.

We summarize our contributions as follows:

1. **Generalization bounds for RLVR**. We establish the first compression-based PAC-Bayesian framework for bounding generalization errors of RLVR, addressing the stochastic reward structure of RLVR via Gumbel-max reparameterization.
2. **Progressive RLVR**. We develop a parameter-efficient post-training pipeline that leverages TinyLoRA and OPD to obtain non-vacuous bounds at the billion-parameter scale.
3. **Non-vacuous bounds for billion-parameter models on real-world RLVR datasets**. We provide the first non-vacuous generalization bounds for billion-parameter LLMs post-trained via RLVR, providing formal verification of RLVR performance on real-world tasks including mathematical problem-solving, programming, general-knowledge reasoning, and text-to-SQL translation.

2 Preliminaries

We present the foundational concepts and formulations that motivate our framework, focusing on RLVR (§2.1), PAC-Bayes generalization bounds (§2.2), and LLM fine-tuning with LoRA (§2.3).

2.1 Reinforcement Learning with Verifiable Rewards

RLVR has emerged as a dominant paradigm of LLM post-training to enhance reasoning (Guo et al., 2025; Chen et al., 2025; Team et al., 2025). Relying on objective verifiers (e.g., mathematical correctness (Shao et al., 2024) or unit tests (Wei et al., 2025)) rather than human preference models (Bai et al., 2022), RLVR uses deterministic reward signals. While the practical instantiations of RLVR (Shao et al., 2024; Yu et al., 2025; Gao et al., 2025; Chen et al., 2025) rely on various customized clipping mechanisms and KL-divergence penalties to stabilize training, all RLVR variants share the core optimization objective of classic reinforcement learning: maximize the expected reward.

In the context of outcome-based RLVR (Shao et al., 2024), the return simplifies to a deterministic scalar reward evaluated on the final generated response. We write the objective as

$$\mathcal{J}_{\text{simplified}}(\theta) = \mathbb{E}_{x\sim\mathcal{P}_0, y\sim\pi_\theta(\cdot|x)}\left[r(x,y)\right],$$

where x represents the prompt drawn from training distribution $\mathcal{P}_0$, and y represents the generation sampled from the model's conditional policy π_θ. We use this objective as the primary expected risk measure to establish our generalization bounds in Section 3.

2.2 PAC-Bayesian Generalization and Compression

The PAC-Bayes framework (Alquier, 2024) is an important tool for establishing generalization bounds for deep neural networks. It bounds the generalization gap via the divergence between a prior and a posterior distribution defined over a hypothesis space $\mathcal{H}$. A central feature of this framework is that the bound penalizes the chosen hypothesis only via its likelihood under a pre-defined prior and its empirical performance. The procedure used to select the hypothesis from $\mathcal{H}$ is unconstrained. For reinforcement learning, Wang et al. (2019) derive generalization bounds through reparameterization for small networks, while achieving non-vacuous bounds for reparameterizable RL on large-scale models remains an open challenge. We pursue this direction for autoregressive RLVR by combining reparameterization with compression-based bounds.

We follow prior work (Lotfi et al., 2022; 2023) to apply the universal Solomonoff prior (Solomonoff, 1964), where the prior probability of a hypothesis scales inversely with its description length (compressibility). In the post-training setting, the parameter update (the delta in parameter space) contains all the information needed to reconstruct the trained model from the pre-trained checkpoint. Therefore, obtaining a tight bound for massive LLMs requires reducing the parameter updates to a minimal number of parameters. A second challenge specific to RLVR is the intractability of the reward structure. In RLVR, the reward is calculated over the stochastically generated tokens $y \sim \pi_\theta(\cdot \mid x)$, whose support is combinatorial in length and vocabulary size. We address this challenge in Section 3.

2.3 Low-Rank Adaptation (LoRA)

LoRA (Hu et al., 2022) is one of the most popular parameter-efficient fine-tuning (PEFT) techniques (Han et al., 2024) for LLMs (Han et al., 2024; Wang et al., 2025; Xu et al., 2026). Given a frozen pre-trained weight $W_0 \in \mathbb{R}^{d\times k}$, LoRA introduces two trainable matrices $A \in \mathbb{R}^{r\times k}$ and $B \in \mathbb{R}^{d\times r}$ (where rank $r \ll \min(d, k)$). During training, LoRA modifies the forward pass as: $W_0 x + \Delta W x = W_0 x + \frac{\alpha}{r} BAx$.

While standard LoRA reduces the parameter count, obtaining non-vacuous PAC-Bayes compression bounds requires more aggressive compression. To achieve this, we adopt TinyLoRA (Morris et al., 2026). Building on the SVD-based parameterization of LoRA-XS (Bałazy et al., 2024), TinyLoRA leverages the truncated singular value decomposition of the frozen pre-trained weight $W_0 \approx U\Sigma V^T$. It replaces the trainable core matrix with a low-dimensional trainable vector $v \in \mathbb{R}^u$ projected through a set of fixed random matrices $P_i \in \mathbb{R}^{r\times r}$. The weight update ΔW is then reparameterized as:

$$\Delta W = U\Sigma \left(\sum_{i=1}^{u} v_i P_i \right) V^T.$$

With TinyLoRA, we can compress the learned policy update to as few as one trainable scalar (Morris et al., 2026). This extreme parameterization forms the architectural basis of our Progressive RLVR framework (Section 4), allowing us to compute tight compression-based bounds.

3 Generalization Bounds for RLVR

We now develop generalization bounds for LLMs post-trained via RLVR. We begin by introducing notation and setup (Section 3.1). Then, we present our main result (Theorem 1), which bounds the generalization gap by decomposing it into an extrinsic component over the prompt distribution and an intrinsic component over the model's generation process. Next, we discuss two extensions of Theorem 1: multi-turn RLVR and out-of-domain (OOD) evaluation (Section 3.2). Finally, we show that the empirical reward can be efficiently estimated from a subsample of training prompts with a mild additive penalty (Section 3.3).

3.1 PAC-Bayes Compression Bounds for RLVR

Notation and setup. Let $\mathcal{P}_0$ denote a distribution over task prompts, and let $\hat{\mathcal{P}}_0$ denote the uniform distribution over a fixed set of m training prompts $\{x_1, \ldots, x_m\}$ drawn i.i.d. from $\mathcal{P}_0$. Let $\pi_{\theta_{\text{ref}}}$ be a fixed pre-trained base model parameterized by θ_{ref}, and let $\mathcal{H}$ be a finite hypothesis space in which each hypothesis $h \in \mathcal{H}$ specifies a parameter update yielding a fine-tuned model $\pi_{\theta(h)}$.

Following Lotfi et al. (2023), we adopt the Solomonoff prior (Solomonoff, 1964), defined as $P(h) = Z^{-1} \cdot 2^{-K(h)}$, where $K(h)$ denotes prefix Kolmogorov complexity (Kolmogorov, 1965) and $Z \leq 1$ is a normalizing constant (Lotfi et al., 2022).

We define the *population reward* of a hypothesis h as the expected value of a deterministic, bounded reward function $r\colon \mathcal{X} \times \mathcal{Y} \to [a,\, a+\Delta]$ where $\mathcal{X}$ is the prompt space and $\mathcal{Y}$ is the generation space:

$$\mathcal{R}(h) \;=\; \mathbb{E}_{x\sim\mathcal{P}_0}\Big[\mathbb{E}_{y\sim\pi_{\theta(h)}(\cdot|x)}[r(x,y)]\Big]\,.$$

Since PAC-Bayes bounds depend on h only through its prior probability $P(h)$ and empirical reward (not on the procedure that produced h), the reward used in the bound need not match the training objective. This enables us to establish bounds on task accuracy throughout, even when training optimizes a richer objective. We leverage this flexibility in Section 5.

Reparameterization of stochastic decoding. While we can bound the outer expectation (over $\mathcal{P}_0$) in $\mathcal{R}(h)$ using standard generalization bounds, the inner expectation is hypothesis-dependent and taken over sequences $y \sim \pi_{\theta(h)}(\cdot \mid x)$, a distribution whose support is combinatorial in the sequence length and vocabulary size. This makes direct evaluation or analytic bounding intractable. To obtain a tractable form, we reparameterize the token decoding process by externalizing the stochasticity into a fixed, hypothesis-independent noise distribution.

Concretely, consider the autoregressive generation of a sequence $y_i = (y_{i,1}, \ldots, y_{i,T})$ with a bounded sequence length T, a temperature $\tau > 0$ and a vocabulary size V. At each step t, the model produces logits $\ell_{i,t} \in \mathbb{R}^V$ and samples a token from the categorical distribution

$$y_{i,t} \sim \mathrm{Categorical}(\mathrm{softmax}(\ell_{i,t}/\tau))\,,$$

whose probabilities depend on the hypothesis h through the logits. The Gumbel-max theorem (Gumbel, 1954; Jang et al., 2016) provides an equivalent construction that externalizes this stochasticity into a fixed noise distribution. Namely, for $\xi_{i,t}^{(v)} \overset{\text{iid}}{\sim} \mathrm{Gumbel}(0,1)$,

$$y_{i,t} \;=\; \arg\max_{v\in[V]}\Big(\tau^{-1}\ell_{i,t}^{(v)} + \xi_{i,t}^{(v)}\Big)$$

is distributed identically to the categorical sample above.

Collecting all noise variables across tokens and generations, we denote $\xi_{i,j} = (\xi_{i,j,1}, \ldots, \xi_{i,j,T}) \in \mathbb{R}^{T\times V}$ with $\xi_{i,j,t}^{(v)} \overset{\text{iid}}{\sim} \mathrm{Gumbel}(0,1)$ and $j \in [n]$ indexing the n independent generation samples per prompt. The token generation process becomes a deterministic function $y_{i,j} = g(x_i, \xi_{i,j}; h)$ of the prompt, noise, and hypothesis. We denote the product distribution over each $\xi_{i,j}$ by $\mathcal{P}_\xi$. Under this reparameterization, the population reward can be equivalently written as

$$\mathcal{R}(h) \;=\; \mathbb{E}_{x\sim\mathcal{P}_0}\left[\mathbb{E}_{\xi\sim\mathcal{P}_\xi}\left[r(x,\, g(x,\xi;\, h))\right]\right].$$

With this reparameterization, both sources of randomness in $\mathcal{R}(h)$, the prompt distribution $\mathcal{P}_0$ and the generation noise $\mathcal{P}_\xi$, are now explicit and hypothesis-independent. The following theorem exploits this structure to bound the gap between the population reward and its empirical estimate. We defer the proof of Theorem 1 to Appendix B.2.

Theorem 1 (PAC-Bayes bound for RLVR)**.** *For any $\delta \in (0,1)$, we have*

$$\mathbb{P}\Bigg[\forall h \in \mathcal{H}, \mathcal{R}(h) \geq \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n} r\left(x_i, g\left(x_i, \xi_{i,j}; h\right)\right) - \Delta\left(\sqrt{\frac{1}{n}}+1\right)\sqrt{\frac{C(h)\log(2) + 2\log(C(h)) + \log(2/\delta)}{2m}}\Bigg] \geq 1-\delta$$

where $\{x_i\}_{i=1}^m \overset{\text{iid}}{\sim} \mathcal{P}_0$, $\{\xi_{i,j}\}_{i\in[m],\, j\in[n]} \overset{\text{iid}}{\sim} \mathcal{P}_\xi$, and $C(h)$ is the compression size in bits of h using a pre-defined encoding scheme.

Remark 1 (Error decomposition)**.** *Theorem 1 directly follows from a decomposition of the generalization gap into two components, each addressing one source of randomness:*

$$\mathcal{R}(h) - \hat{R}_{m,n}(h) \;=\; \underbrace{\mathcal{R}(h) - \frac{1}{m}\sum_{i=1}^{m}\bar{r}(x_i;h)}_{\varepsilon_{\text{ext}}(h)\ \textit{(extrinsic error)}} \;+\; \underbrace{\frac{1}{m}\sum_{i=1}^{m}\bar{r}(x_i;h) - \hat{R}_{m,n}(h)}_{\varepsilon_{\text{int}}(h)\ \textit{(intrinsic error)}},$$

where $\bar{r}(x;h) = \mathbb{E}_\xi[r(x, g(x,\xi;h))]$ *and* $\hat{R}_{m,n}(h) = \frac{1}{mn}\sum_{i,j} r(x_i, g(x_i,\xi_{i,j};h))$. *The extrinsic error* $\varepsilon_{\text{ext}}(h)$ *measures the generalization over the prompt distribution, while the intrinsic error* $\varepsilon_{\text{int}}(h)$ *measures estimation accuracy of the empirical reward given* n *sampled generations per prompt. Bounding each yields the* $\mathcal{O}(1/\sqrt{m})$ *and* $\mathcal{O}(1/\sqrt{mn})$ *rates in Theorem 1. Because* m *is fixed by the training set, while* n *is freely chosen at evaluation time, taking* $n = \Omega(m)$ *shrinks the intrinsic rate to* $\mathcal{O}(1/m)$*, lower-order than the extrinsic* $\mathcal{O}(1/\sqrt{m})$*. Hence, the bound is effectively governed by the extrinsic error.*

3.2 Extensions of Theorem 1

We extend Theorem 1 to two practically important settings: (1) multi-turn RLVR, where the model interacts with an environment over multiple steps, and (2) out-of-distribution (OOD) evaluation. Multi-turn RLVR introduces a potential new source of randomness from the environment's responses on top of the per-token Gumbel noise treated in Section 3.1. We handle multi-turn RLVR by considering two cases: environments whose transitions are deterministic, and environments whose stochastic transitions admit a reparameterization.

Multi-turn RLVR with deterministic environments. Theorem 1 extends without modification to multi-turn RLVR with deterministic environments, which cover most practical RLVR settings, such as mathematical problem-solving with symbolic verifiers and code execution on deterministic tests. In such settings, the environment introduces no additional randomness due to deterministic transition and termination functions. Given a prompt and interaction history, the next observation and termination signal are uniquely determined. The full trajectory $\tau = (o_0, a_1, o_1, \ldots, a_H, o_H)$ over a bounded horizon H is therefore a deterministic function of the prompt, the model's responses, and the environment's responses. Because the model's responses are themselves token sequences generated via Gumbel-max reparameterization, the trajectory reduces to a deterministic function $G(x_i, \xi_{i,j}; h)$, where $\xi_{i,j} \in \mathbb{R}^{H\times T\times V}$ collects noise across all H turns. The proof of Theorem 1 applies verbatim under the substitution $g \mapsto G$.

Multi-turn RLVR with stochastic environments. When the environment itself is *stochastic*, its transitions contribute a second source of randomness, independent of the model's token decoding. If stochastic transitions of the environment admit a reparameterization (e.g., tool responses expressed as deterministic functions of random noise), we introduce a hypothesis-independent noise variable $\eta_{i,j} \sim \mathcal{P}_\eta$ alongside the Gumbel noise. The trajectory becomes $G(x_i, \xi_{i,j}, \eta_{i,j}; h)$, and Theorem 1 continues to hold with (ξ, η) jointly playing the role of the noise variable. Non-stationary environments, such as live tool APIs (Wang et al., 2026) or environments whose transitions shift between training and deployment, fall outside the scope of this work and are left for future work.

OOD evaluation. When a labeled corpus is available from a different target distribution $\mathcal{Q}_0$, a similar bound applies to the expected reward $\mathcal{R}_\mathcal{Q}(h)$ under $\mathcal{Q}_0$. As long as the hypothesis is chosen independently of the OOD corpus, the compression penalty $C(h)\log 2 + 2\log C(h)$ drops out and only a Hoeffding term of size $\mathcal{O}(\Delta/\sqrt{m'})$ in the OOD sample size m' remains. We defer the formal statement (Theorem 2) and proof to Appendix B.4, and an empirical study to Appendix F.

3.3 Subsampling for Efficient Empirical Reward Computation

The generalization error bound in Theorem 1 involves calculating the empirical reward $\frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n} r(x_i, g(x_i,\xi_{i,j};h))$, which requires evaluating the RLVR model on all m data points and n noise vectors per data point. When m is large, this full-dataset evaluation is computationally expensive. With the following corollary, we can estimate the empirical reward using a subsample of training prompts, incurring a small additive penalty. We defer the proof to Appendix B.3.

Corollary 1. *Under the setting of Theorem 1, with training prompts* $x_1, \ldots, x_m \overset{\text{iid}}{\sim} \mathcal{P}_0$ *and independent Gumbel noise vectors* $\xi_{i,j} \overset{\text{iid}}{\sim} \mathcal{P}_\xi$ *for* $i \in [m]$, $j \in [n]$, *let* $\{i_1, \ldots, i_{m_0}\} \subset [m]$ *be a subsample drawn uniformly at random, and* $h \in \mathcal{H}$ *be any hypothesis chosen independently of the subsample.*

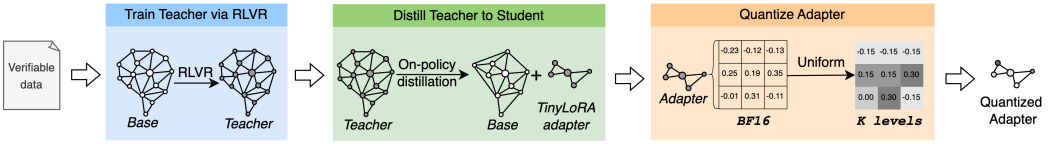

Figure 1: Our proposed Progressive RLVR pipeline, which integrates RLVR with parameter-efficient fine-tuning, OPD, and quantization to achieve high training rewards while preserving small compression sizes.

Then for any $\delta \in (0, 1)$,

$$\mathbb{P}\Bigg[\mathcal{R}(h) \geq \frac{1}{m_0 n}\sum_{k=1}^{m_0}\sum_{j=1}^{n} r\big(x_{i_k},\, g(x_{i_k}, \xi_{i_k,j};\, h)\big) \;-\; \Delta\sqrt{\frac{\log(3/\delta)}{2m_0}} \\ -\; \Delta\left(\sqrt{\tfrac{1}{n}}+1\right)\sqrt{\frac{C(h)\log 2 + 2\log C(h) + \log(3/\delta)}{2m}}\Bigg] \geq 1-\delta.$$

4 Progressive RLVR via On-Policy Distillation

To establish non-vacuous bounds via Theorem 1, we require simultaneously large empirical reward $r(x_i, g(x_i, \xi_{i,j}; h))$ and small compression size $C(h)$. While this is challenging, the PAC-Bayes bounds in Theorem 1 do not constrain the procedure that produces h. This decouples *capacity used during search* from *capacity charged to the bound*. Namely, we are free to use a high-capacity teacher to explore the reasoning landscape, as long as the final compression size is small. To exploit this asymmetry, we introduce the Progressive RLVR pipeline (Figure 1).

RLVR and distillation. As shown in Figure 1, we first train a high-capacity teacher model using a standard RLVR objective. This allows the teacher to freely explore and master complex reasoning trajectories via RLVR without restrictive parameter bottlenecks. Enabling this exploration is a key feature of our approach, since the bound depends only on the final model, not on the procedure. Next, we apply OPD (Agarwal et al., 2024) to compress the capabilities learned in RLVR into a student model with TinyLoRA (Morris et al., 2026) parameterization. OPD improves the sample efficiency and boosts the performance of ultra-compact models by providing token-level optimization signals. In Figure 2, we show that on the math domain, OPD achieves 10% higher rewards than direct RLVR on average, given the same number of training steps.

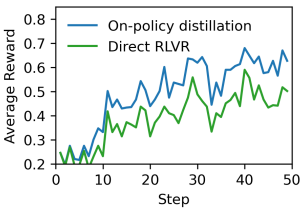

Figure 2: Training rewards of the distillation phase of Progressive RLVR, achieving 10% higher rewards than direct RLVR on average.

Quantization. The final stage of our Progressive RLVR pipeline applies quantizes the trained TinyLoRA adapter of the student model to further reduce its description length. Specifically, we quantize the parameters of the TinyLoRA adapter into uniformly distributed discrete levels (Lotfi et al., 2023; Han et al., 2015). Empirically, this quantization step vastly reduces the bit-width of the adapter while inducing only minor degradation in the acquired empirical reward.

5 Non-Vacuous RLVR Generalization Bounds for Billion-Parameter LLMs

In this section, we empirically demonstrate that our framework yields non-vacuous and tight generalization bounds for billion-parameter LLMs post-trained via TinyLoRA-based RLVR. We first detail our experimental setup, including environments, models, and optimization hyperparameters in Section 5.1. We then evaluate our bounds and report the main empirical findings in Section 5.2.

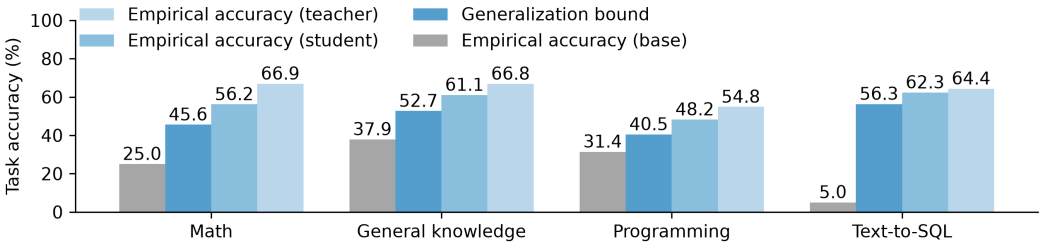

Figure 3: Our framework yields non-vacuous generalization bounds on Qwen3.5-4B post-trained via Progressive RLVR on four domains. We report the generalization bound ($\delta = 0.05$) on the quantized TinyLoRA student, alongside the empirical accuracy of the teacher, student, and base model. The bound exceeds base model's accuracy in all domains and lies within 6–11% of the student's accuracy.

5.1 Experimental Settings

Datasets and environments. To demonstrate the applicability of our framework on diverse application domains, we consider four datasets and environments:

1. **Mathematical problem-solving**: We use the mathematics subset of the Eurus-2-RL dataset (Cui et al., 2025), sourced from NuminaMath-CoT (LI et al., 2024). This contains 456,285 questions ranging from high-school mathematics to International Mathematical Olympiad challenges. We standardize multiple-choice questions to require only the choice index and normalize string representations (e.g., Unicode) to LaTeX. We verify the correctness of an answer using `sympy` and `math_verify` (Kydlíček, 2026) , yielding a binary reward $r \in \{0, 1\}$. Hence, the average reward is equivalent to task accuracy.
2. **Programming**: We construct a programming dataset by merging the programming subset of Eurus-2-RL (Cui et al., 2025) with the KodCode-V1 dataset (Xu et al., 2025), yielding 504,885 programming tasks. We evaluate the correctness of a generated program against the unit tests provided with each task, yielding a binary reward $r \in \{0, 1\}$.
3. **General knowledge**: We combine large-scale datasets from knowledge-intensive domains, including Med-QA (Jin et al., 2020), LogiQA (Liu et al., 2020), C-Eval (Huang et al., 2023), Arc (Clark et al., 2018), LegalBench (Guha et al., 2023; Koreeda & Manning, 2021; Hendrycks et al., 2021c; Wang et al., 2023; Wilson et al., 2016; Zheng et al., 2021; Zimmeck et al., 2019; Ravichander et al., 2019; Holzenberger & Van Durme, 2021; Lippi et al., 2019), MMLU (Hendrycks et al., 2021b;a), and Webinstruct-verified (Ma et al., 2025). We exclude the "Mathematics" subset of the Webinstruct-verified dataset to avoid overlap with the math domain, yielding 438,908 data points. We consider an answer correct if the final answer is enclosed in `\boxed{}` and matches the ground truth exactly, leading to a binary reward $r \in \{0, 1\}$.
4. **Text-to-SQL**: We use the SynSQL-2.5M dataset (Li et al., 2025a), which contains 2,544,390 synthetic natural language questions mapped to SQL queries. Following SkyRL-SQL (Liu et al.), we enforce a strict output formatting constraint (`<solution></solution>`) to reliably extract the generated SQL. Our environment allows the model to interactively query the database for up to five turns. We assign a trajectory a reward of 1 if the final query returns the same result as the ground-truth SQL query, 0 if it leads to a different result, and -1 if it is malformed or unparseable. We report the bound on binary task accuracy to be consistent with the other domains. We defer the non-vacuous generalization bound on the tri-level reward to Appendix G.

Models. We use the pre-trained Qwen3.5-4B (Team, 2026b) as the base model ($\pi_{\theta_{\text{ref}}}$) for both the teacher and student. We additionally use Qwen3.5-2B for ablation studies.

Hyperparameters. To train teachers, we apply standard LoRA with a rank of 64. We use a standard importance-sampling objective (Shao et al., 2024; Liu et al., 2025) without a KL-divergence penalty. For OPD, we use TinyLoRA with an inner dimension of $u = 16$ and a projection rank of $r = 16$. In both phases, we use a batch size of 64 and 8 rollouts per prompt. We sweep over the learning rates $\{10^{-5}, 5 \times 10^{-5}, 10^{-4}\}$, selecting the configuration that maximizes the training reward. Using Corollary 1, we estimate the empirical reward on the training corpus using 4,096 uniformly sampled

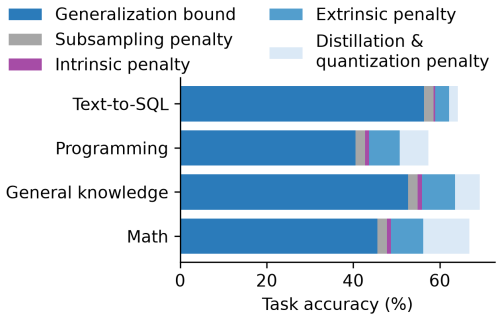

Figure 4: Decomposition of the teacher's accuracy (the total bar length) into our generalization bounds and the penalties separating them.

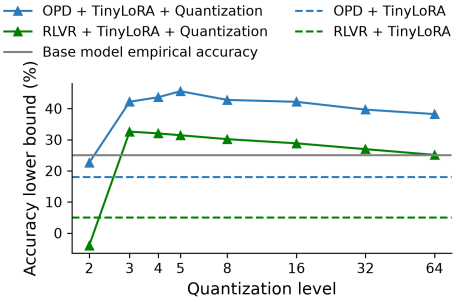

Figure 5: Ablation of the Progressive RLVR pipeline. We skip standard LoRA in the figure since it yields vacuous bounds with either OPD or RLVR, with or without quantization.

prompts and 64 generations per prompt. For quantization, we use five levels (2.3 bits). We defer details about the hyperparameters to Appendix C and the encoding scheme to Appendix E.

Implementation and hardware. We implement the teacher training procedure using the Tinker API[1] and execute it on a local cluster of 8×3GHz vCPUs with 80 GB RAM. We implement OPD with TinyLoRA via SkyRL (Cao et al., 2025) and execute it with two A100s.

5.2 Non-vacuous Bounds on Real-World RLVR Tasks

Non-vacuous bounds across four domains. In Figure 3, we present our primary results, the generalization bounds from Corollary 1. As shown, the generalization bounds are non-vacuous (outperforming both random guess and the base model) in all domains, indicating that the model fine-tuned via Progressive RLVR endows the base model with new capabilities that generalize beyond the training data. Specifically, our bounds exceed the accuracy of the base model by 9–51%, providing meaningful verification for the generalizability of billion-parameter LLMs post-trained via Progressive RLVR. No prior framework produces non-vacuous generalization bounds for billion-parameter RLVR models on real-world datasets.

In addition to being non-vacuous, our bounds are tight. As shown, the gap between the student's accuracy and the generalization bound ranges over 6–11% across domains. This consistency across different tasks suggests that the tightness of our bounds is governed by properties of the training setup (sample sizes, compression) rather than domain-specific factors.

Decomposition of the bounds. In Figure 4, we break down the accuracy of the teacher into our generalization bound and the penalties separating the bound from the teacher's accuracy. As shown, the dominant penalty differs across domains. For text-to-SQL, all penalties are small (total 8%), reflecting both the student's close match to the teacher and the tight extrinsic penalty due to a large training set of 2.5 million examples. For other domains, we identify similar contributions from the extrinsic penalty (7–8%) and the distillation and quantization penalty (6–11%), suggesting that additional training prompts and a better optimized TinyLoRA configuration would tighten these bounds. In addition, we find that the intrinsic generalization gaps are negligible in all domains, confirming that 64 generations per prompt are sufficient for reward estimation.

Ablation study of Progressive RLVR. In Figure 5, we isolate the effects of OPD and quantization on Math with Qwen3.5-4B and TinyLoRA. Removing OPD reduces the bound by 15% on average, while removing quantization leads to bounds below the empirical accuracy of the base model. Therefore, both OPD and quantization are critical to obtain tight bounds. Furthermore, overly aggressive quantization ($k = 2$) yields vacuous bounds, illustrating the tradeoff between description length and empirical accuracy. In Appendix D, we extend this analysis to standard LoRA, which yields vacuous bounds under both OPD and RLVR, with or without quantization.

[1] https://tinker-docs.thinkingmachines.ai/tinker

6 DISCUSSION

Our results support *verified specialization*: training a domain-specific RLVR model and obtaining a high-probability lower bound on its expected accuracy on unseen data drawn from the same distribution as the training set. This vision is increasingly relevant as practitioners specialize open-weight models on proprietary or internal data and find them outperforming larger general-purpose models on target tasks Team (2025). Without theoretical verification, it remains unclear whether the RLVR model overfits or generalizes. Our bounds for Qwen3.5-4B exceed the base model's accuracy by 9–51%, showing that a meaningful verification of this kind is achievable at billion-parameter scale. Beyond establishing this in theory, our proposed Progressive RLVR offers a concrete recipe for achieving it in practice: train a high-capacity teacher, distill into a compact student, and quantize. The effectiveness of this recipe further suggests that distillation plays a role beyond cost reduction, enabling verifiable generalization that pipelines relying on compression alone struggle to achieve.

7 RELATED WORK

RLVR and its generalizability. Building on GRPO (Shao et al., 2024), methods such as DAPO, CISPO, SAPO, Dr. GRPO, and TIS address optimization stability, bias, and sampling–training mismatch (Yu et al., 2025; Chen et al., 2025; Gao et al., 2025; Liu et al., 2025; Yao et al., 2025). Despite these algorithmic successes, a tension exists in the empirical literature regarding RLVR's generalizability. Some studies show that RLVR's generalization to unseen data is fragile (Hu et al., 2025). RLVR often induces narrowed exploration that forces models to overfit to highly rewarded training trajectories (Yue et al., 2025; Nguyen et al., 2025). Conversely, other studies argue that RL post-training fosters broader generalization than standard supervised fine-tuning (Chu et al., 2025). This empirical debate underscores the critical need for rigorous theoretical generalization bounds.

Generalization Bounds. Classical complexity and norm-based analyses can yield vacuous bounds for modern neural networks (Zhang et al., 2016; Nagarajan & Kolter, 2019; Arora et al., 2018). PAC-Bayes (Lotfi et al., 2022; Grünwald & Mehta, 2019) has emerged as an important theory to explain the generalization of deep neural networks. Prior work has established non-vacuous compression-based PAC-Bayes bounds for LLM pre-training (Lotfi et al., 2023). However, extending these bounds to RLVR is non-trivial. Reinforcement learning optimizes over a non-stationary state-action distribution dependent on the active policy, violating the *i.i.d.* assumptions of standard supervised bounds. While prior work (Wang et al., 2019) established bounds for specific reparameterizable RL, they do not prove compression-based PAC-Bayes bounds and do not establish non-vacuous bounds for LLMs.

Parameter-Efficient Fine-Tuning. To reduce the massive computational costs of fine-tuning LLMs, a wide range of PEFT techniques has emerged (Wang et al., 2025; Xu et al., 2026). LoRA and subsequent methods reduce trainable parameters through adapter placement, shared or structured parameterizations, and quantization (Hayou et al., 2025; Li et al., 2025b; Zhang et al., 2025; Ye et al., 2026; Hwang et al., 2025; Li et al., 2023; Xu et al., 2024; Hu et al., 2022). TinyLoRA further reduces updates to a small set of trainable coefficients (Morris et al., 2026). Progressive RLVR combines these established techniques with adapter quantization to preserve empirical accuracy under the description-length budget required for a non-vacuous bound.

8 CONCLUSION

We present the first non-vacuous generalization bounds for LLMs post-trained via RLVR. Our approach leverages PAC-Bayes compression bounds and uses a Gumbel-max reparameterization to account for the stochasticity of token generation and the combinatorially large support of the language model's output distribution. To meet the compression size and accuracy requirements of the theoretical bounds, we introduce the Progressive RLVR pipeline, which integrates RLVR with TinyLoRA, OPD, and quantization. Empirically, we demonstrate non-vacuous bounds for Qwen3.5-4B across four real-world domains, with Progressive RLVR achieving tighter bounds than direct RLVR with TinyLoRA. We hope this work provides a foundation for verifying the generalization of RLVR models, particularly in high-stakes applications.

AI USE STATEMENT

In this work, we used generative AI tools for check grammar errors and citation format errors. We have not used generative AI tools for generating synthetic data sets, helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, proposing or refine hypotheses, designing or provide feedback on research methodology or experiments, implementing methods, assisting with translation, cleaning and reformating dataset, supporting qualitative and thematic data analysis, interpreting results, or writing complete sentences, paragraphs, or sections. We have reviewed all AI-assisted work. We checked LLM-corrected grammars and citation formats a manual verification. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *The twelfth international conference on learning representations*, 2024.

Pierre Alquier. User-friendly introduction to pac-bayes bounds. *Foundations and Trends in Machine Learning*, 17(2):174–303, 2024.

Sanjeev Arora, Rong Ge, Behnam Neyshabur, and Yi Zhang. Stronger generalization bounds for deep nets via a compression approach. In *International conference on machine learning*, pp. 254–263. PMLR, 2018.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

Klaudia Bałazy, Mohammadreza Banaei, Karl Aberer, and Jacek Tabor. Lora-xs: Low-rank adaptation with extremely small number of parameters. *arXiv preprint arXiv:2405.17604*, 2024.

Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of machine learning research*, 3(Nov):463–482, 2002.

Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K Warmuth. Learnability and the vapnik-chervonenkis dimension. *Journal of the ACM (JACM)*, 36(4):929–965, 1989.

Shiyi Cao, Sumanth Hegde, Dacheng Li, Tyler Griggs, Shu Liu, Eric Tang, Jiayi Pan, Xingyao Wang, Akshay Malik, Graham Neubig, Kourosh Hakhamaneshi, Richard Liaw, Philipp Moritz, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. Skyrl-v0: Train real-world long-horizon agents via reinforcement learning, 2025.

Aili Chen, Aonian Li, Bangwei Gong, Binyang Jiang, Bo Fei, Bo Yang, Boji Shan, Changqing Yu, Chao Wang, Cheng Zhu, et al. Minimax-m1: Scaling test-time compute efficiently with lightning attention. *arXiv preprint arXiv:2506.13585*, 2025.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. *arXiv preprint arXiv:2501.17161*, 2025.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv:1803.05457v1*, 2018.

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Yuchen Zhang, Jiacheng Chen, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, et al. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*, 2025.

Chang Gao, Chujie Zheng, Xiong-Hui Chen, Kai Dang, Shixuan Liu, Bowen Yu, An Yang, Shuai Bai, Jingren Zhou, and Junyang Lin. Soft adaptive policy optimization. *arXiv preprint arXiv:2511.20347*, 2025.

Peter D Grünwald and Nishant A Mehta. A tight excess risk bound via a unified pac-bayesian–rademacher–shtarkov–mdl complexity. In *Algorithmic learning theory*, pp. 433–465. PMLR, 2019.

Neel Guha, Julian Nyarko, Daniel E. Ho, Christopher Ré, Adam Chilton, Aditya Narayana, Alex Chohlas-Wood, Austin Peters, Brandon Waldon, Daniel N. Rockmore, Diego Zambrano, Dmitry Talisman, Enam Hoque, Faiz Surani, Frank Fagan, Galit Sarfaty, Gregory M. Dickinson, Haggai Porat, Jason Hegland, Jessica Wu, Joe Nudell, Joel Niklaus, John Nay, Jonathan H. Choi, Kevin Tobia, Margaret Hagan, Megan Ma, Michael Livermore, Nikon Rasumov-Rahe, Nils Holzenberger, Noam Kolt, Peter Henderson, Sean Rehaag, Sharad Goel, Shang Gao, Spencer Williams, Sunny Gandhi, Tom Zur, Varun Iyer, and Zehua Li. Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models, 2023.

Emil Julius Gumbel. *Statistical theory of extreme values and some practical applications: a series of lectures*, volume 33. US Government Printing Office, 1954.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Paul Hager, Friederike Jungmann, Robbie Holland, Kunal Bhagat, Inga Hubrecht, Manuel Knauer, Jakob Vielhauer, Marcus Makowski, Rickmer Braren, Georgios Kaissis, et al. Evaluation and mitigation of the limitations of large language models in clinical decision-making. *Nature medicine*, 30(9):2613–2622, 2024.

Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. Parameter-efficient fine-tuning for large models: A comprehensive survey. *arXiv preprint arXiv:2403.14608*, 2024.

Soufiane Hayou, Nikhil Ghosh, and Bin Yu. Plop: Precise lora placement for efficient finetuning of large models. *arXiv preprint arXiv:2506.20629*, 2025.

Dan Hendrycks, Collin Burns, Steven Basart, Andrew Critch, Jerry Li, Dawn Song, and Jacob Steinhardt. Aligning ai with shared human values. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021a.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021b.

Dan Hendrycks, Collin Burns, Anya Chen, and Spencer Ball. Cuad: An expert-annotated nlp dataset for legal contract review. *arXiv preprint arXiv:2103.06268*, 2021c.

Nils Holzenberger and Benjamin Van Durme. Factoring statutory reasoning as language understanding challenges. *arXiv preprint arXiv:2105.07903*, 2021.

Chuxuan Hu, Yuxuan Zhu, Antony Kellermann, Caleb Biddulph, Suppakit Waiwitlikhit, Jason Benn, and Daniel Kang. Breaking barriers: Do reinforcement post training gains transfer to unseen domains? *arXiv preprint arXiv:2506.19733*, 2025.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *ICLR*, 2022.

Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Jiayi Lei, Yao Fu, Maosong Sun, and Junxian He. C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models. In *Advances in Neural Information Processing Systems*, 2023.

Junseo Hwang, Wonguk Cho, and Taesup Kim. Pica: Parameter-efficient fine-tuning with column space projection. *arXiv preprint arXiv:2505.20211*, 2025.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *arXiv preprint arXiv:2009.13081*, 2020.

Andrei N Kolmogorov. Three approaches to the quantitative definition of information. *Problems of information transmission*, 1(1):1–7, 1965.

Yuta Koreeda and Christopher D Manning. Contractnli: A dataset for document-level natural language inference for contracts. *arXiv preprint arXiv:2110.01799*, 2021.

Raphail Krichevsky and Victor Trofimov. The performance of universal encoding. *IEEE Transactions on Information Theory*, 27(2):199–207, 1981.

Hynek Kydlíček. Math-verify, 2026. URL `https://github.com/huggingface/Math-Verify`. Accessed on 09-20-2026.

Haoyang Li, Shang Wu, Xiaokang Zhang, Xinmei Huang, Jing Zhang, Fuxin Jiang, Shuai Wang, Tieying Zhang, Jianjun Chen, Rui Shi, et al. Omnisql: Synthesizing high-quality text-to-sql data at scale. *arXiv preprint arXiv:2503.02240*, 2025a.

Jia LI, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. Numinamath, 2024.

Kaiyang Li, Shaobo Han, Qing Su, Wei Li, Zhipeng Cai, and Shihao Ji. Uni-lora: One vector is all you need. *arXiv preprint arXiv:2506.00799*, 2025b.

Yixiao Li, Yifan Yu, Chen Liang, Pengcheng He, Nikos Karampatziakis, Weizhu Chen, and Tuo Zhao. Loftq: Lora-fine-tuning-aware quantization for large language models. *arXiv preprint arXiv:2310.08659*, 2023.

Marco Lippi, Przemysław Pałka, Giuseppe Contissa, Francesca Lagioia, Hans-Wolfgang Micklitz, Giovanni Sartor, and Paolo Torroni. Claudette: an automated detector of potentially unfair clauses in online terms of service. *Artificial Intelligence and Law*, 27:117–139, 2019.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. *arXiv preprint arXiv:2007.08124*, 2020.

Shu Liu, Alan Zhu, Sumanth Hegde, Shiyi Cao, Shuo Yuan, Samion Suwito, Tyler Griggs, Matei Zaharia, Joseph E Gonzalez, and Ion Stoica. Skyrl-sql: Multi-turn sql data agents via rl. In *First Workshop on Multi-Turn Interactions in Large Language Models*.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025.

Sanae Lotfi, Marc Finzi, Sanyam Kapoor, Andres Potapczynski, Micah Goldblum, and Andrew G Wilson. Pac-bayes compression bounds so tight that they can explain generalization. *Advances in Neural Information Processing Systems*, 35:31459–31473, 2022.

Sanae Lotfi, Marc Finzi, Yilun Kuang, Tim GJ Rudner, Micah Goldblum, and Andrew Gordon Wilson. Non-vacuous generalization bounds for large language models. *arXiv preprint arXiv:2312.17173*, 2023.

Xueguang Ma, Qian Liu, Dongfu Jiang, Ge Zhang, Zejun MA, and Wenhu Chen. General-Reasoner: Advancing LLM reasoning across all domains. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL `https://openreview.net/forum?id=pBFVoll8Xa`.

John X Morris, Niloofar Mireshghallah, Mark Ibrahim, and Saeed Mahloujifar. Learning to reason in 13 parameters. *arXiv preprint arXiv:2602.04118*, 2026.

Vaishnavh Nagarajan and J Zico Kolter. Uniform convergence may be unable to explain generalization in deep learning. *Advances in neural information processing systems*, 32, 2019.

Phuc Minh Nguyen, Chinh D La, Duy MH Nguyen, Nitesh V Chawla, Binh T Nguyen, and Khoa D Doan. The reasoning boundary paradox: How reinforcement learning constrains language models. *arXiv preprint arXiv:2510.02230*, 2025.

Abhilasha Ravichander, Alan W Black, Shomir Wilson, Thomas Norton, and Norman Sadeh. Question answering for privacy policies: Combining computational and legal perspectives. *arXiv preprint arXiv:1911.00841*, 2019.

Borja Rodriguez-Galvez, Ragnar Thobaben, and Mikael Skoglund. More pac-bayes bounds: From bounded losses, to losses with general tail behaviors, to anytime validity. *Journal of Machine Learning Research*, 25(110):1–43, 2024.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Ray J Solomonoff. A formal theory of inductive inference. part i. *Information and control*, 7(1):1–22, 1964.

Kimi Team. Kimi k2.6: From code to creation, from one to many, April 2026a. URL https://www.kimi.com/ai-models/kimi-k2-6.

Kimi Team, Yifan Bai, Yiping Bao, Y Charles, Cheng Chen, Guanduo Chen, Haiting Chen, Huarong Chen, Jiahao Chen, Ningxin Chen, et al. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*, 2025.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026b. URL https://qwen.ai/blog?id=qwen3.5.

The Databricks AI Research Team. The power of rlvr: Training a leading sql reasoning model on databricks, July 2025. URL https://www.databricks.com/blog/power-rlvr-training-leading-sql-reasoning-model-databricks.

Huan Wang, Stephan Zheng, Caiming Xiong, and Richard Socher. On the generalization gap in reparameterizable reinforcement learning. In *International Conference on Machine Learning*, pp. 6648–6658. PMLR, 2019.

Luping Wang, Sheng Chen, Linnan Jiang, Shu Pan, Runze Cai, Sen Yang, and Fei Yang. Parameter-efficient fine-tuning in large language models: a survey of methodologies. *Artificial Intelligence Review*, 58(8):227, 2025.

Steven H Wang, Antoine Scardigli, Leonard Tang, Wei Chen, Dimitry Levkin, Anya Chen, Spencer Ball, Thomas Woodside, Oliver Zhang, and Dan Hendrycks. Maud: An expert-annotated legal nlp dataset for merger agreement understanding. *arXiv preprint arXiv:2301.00876*, 2023.

Yinjie Wang, Xuyang Chen, Xiaolong Jin, Mengdi Wang, and Ling Yang. Openclaw-rl: Train any agent simply by talking. *arXiv preprint arXiv:2603.10165*, 2026.

Yuxiang Wei, Olivier Duchenne, Jade Copet, Quentin Carbonneaux, Lingming Zhang, Daniel Fried, Gabriel Synnaeve, Rishabh Singh, and Sida I Wang. Swe-rl: Advancing llm reasoning via reinforcement learning on open software evolution. *arXiv preprint arXiv:2502.18449*, 2025.

Shomir Wilson, Florian Schaub, Aswarth Abhilash Dara, Frederick Liu, Sushain Cherivirala, Pedro Giovanni Leon, Mads Schaarup Andersen, Sebastian Zimmeck, Kanthashree Mysore Sathyendra, N Cameron Russell, et al. The creation and analysis of a website privacy policy corpus. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1330–1340, 2016.

Lingling Xu, Haoran Xie, S Joe Qin, Xiaohui Tao, and Fu Lee Wang. Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026.

Yuhui Xu, Lingxi Xie, Xiaotao Gu, Xin Chen, Heng Chang, Hengheng Zhang, Zhengsu Chen, Xiaopeng Zhang, and Qi Tian. Qa-lora: Quantization-aware low-rank adaptation of large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

Zhangchen Xu, Yang Liu, Yueqin Yin, Mingyuan Zhou, and Radha Poovendran. Kodcode: A diverse, challenging, and verifiable synthetic dataset for coding. 2025. URL https://arxiv.org/abs/2503.02951.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Feng Yao, Liyuan Liu, Dinghuai Zhang, Chengyu Dong, Jingbo Shang, and Jianfeng Gao. Your efficient rl framework secretly brings you off-policy rl training, August 2025. URL https://fengyao.notion.site/off-policy-rl.

Bangguo Ye, Yuanwei Zhang, and Xiaoqun Zhang. Trac: Tensor-train based across-layer compression for parameter-efficient fine-tuning. In *The Fourteenth International Conference on Learning Representations*, 2026.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *arXiv preprint arXiv:2504.13837*, 2025.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

Yifei Zhang, Hao Zhu, Junhao Dong, Haoran Shi, Ziqiao Meng, Piotr Koniusz, and Han Yu. Crossspectra: Exploiting cross-layer smoothness for parameter-efficient fine-tuning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Lucia Zheng, Neel Guha, Brandon R Anderson, Peter Henderson, and Daniel E Ho. When does pretraining help? assessing self-supervised learning for law and the casehold dataset of 53,000+ legal holdings. In *Proceedings of the eighteenth international conference on artificial intelligence and law*, pp. 159–168, 2021.

Sebastian Zimmeck, Peter Story, Daniel Smullen, Abhilasha Ravichander, Ziqi Wang, Joel R Reidenberg, N Cameron Russell, and Norman Sadeh. Maps: Scaling privacy compliance analysis to a million apps. *Proc. Priv. Enhancing Tech.*, 2019.

A Limitations and Future Directions

Our work has two limitations that could motivate directions for future research. First, our bounds apply only when the deployment distribution matches the training distribution. They fail to hold under domain shift without labeled out-of-domain data. Second, our experiments are at the 4B-parameter scale, while real-world deployed models (e.g., Kimi-K2.6 Team (2026a) and Qwen3.5-397B-A17B Team (2026b)) are often orders of magnitude larger. The compression budget our bounds require has not been shown achievable at that scale without sacrificing empirical reward. Therefore, we call for future study on the generalizability of larger-scale models in practical use.

Table 1: Summary of notation.

Symbol	Description
Spaces and distributions	
$\mathcal{X}$	Prompt space
$\mathcal{Y}$	Generation (sequence) space
$\mathcal{P}_0$	Population distribution over task prompts
$\hat{\mathcal{P}}_0$	Empirical distribution over m training prompts
$\mathcal{P}_\xi$	Product distribution over Gumbel noise vectors
$\mathcal{H}$	Finite hypothesis space
Models and hypotheses	
θ_{ref}	Parameters of the frozen pre-trained base model
$\pi_{\theta_{\text{ref}}}$	Base model policy
h	Hypothesis (parameter update) in $\mathcal{H}$
$\pi_{\theta(h)}$	Fine-tuned model policy under hypothesis h
Generation and reward	
V	Vocabulary size
T	Maximum sequence length
τ	Sampling temperature
$\ell_{i,t} \in \mathbb{R}^V$	Logits at step t for prompt x_i
$\xi_{i,j} \in \mathbb{R}^{T\times V}$	Gumbel noise for prompt i, generation j
$g(x_i, \xi_{i,j};\, h)$	Deterministic generation function (reparameterized)
$r\colon \mathcal{X}\times\mathcal{Y} \to [a,\, a{+}\Delta]$	Deterministic, bounded reward function
Δ	Range of the reward function
$\bar{r}(x; h)$	Expected reward: $\mathbb{E}_\xi[r(x, g(x, \xi; h))]$
$\mathcal{R}(h)$	Population reward of hypothesis h
Sampling and evaluation	
m	Number of training prompts
n	Number of generations (noise draws) per prompt
m_0	Number of subsampled prompts ($m_0 \le m$)
Complexity and prior	
$K(h)$	Prefix Kolmogorov complexity of h
$C(h)$	Compression size of h in bits (upper bound on $K(h)$)
$P(h)$	Solomonoff prior: $Z^{-1}\cdot 2^{-K(h)}$
Z	Normalizing constant ($Z \le 1$)
Error decomposition	
$\varepsilon_{\text{ext}}(h)$	Extrinsic error (prompt generalization gap)
$\varepsilon_{\text{int}}(h)$	Intrinsic error (generation Monte Carlo gap)
$\varepsilon_{\text{sub}}(h)$	Subsampling error (Corollary 1)
δ	Confidence parameter

B Proof of Generalization Bounds

B.1 Notation

We summarize the notation in Table 1.

B.2 Proof of Theorem 1

Proof. We decompose the generalization gap into two components:

$$\begin{aligned}
&\mathcal{R}(h) - \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n} r\big(x_i, g(x_i, \xi_j; h)\big) \\
&= \underbrace{\mathbb{E}_{x\sim\mathcal{P}_0}\big[\mathbb{E}_{\xi\sim\mathcal{P}_\xi}[r(x, g(x, \xi; h))]\big] - \frac{1}{m}\sum_{i=1}^{m}\mathbb{E}_{\xi\sim\mathcal{P}_\xi}[r(x_i, g(x_i, \xi; h))]}_{\varepsilon_{\text{ext}}(h)\ \text{(extrinsic error)}} \\
&\quad + \underbrace{\frac{1}{m}\sum_{i=1}^{m}\mathbb{E}_{\xi\sim\mathcal{P}_\xi}[r(x, g(x, \xi; h))] - \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n} r\big(x_i, g(x_i, \xi_j; h)\big)}_{\varepsilon_{\text{int}}(h)\ \text{(intrinsic error)}}
\end{aligned}$$

We bound each term separately.

Bounding the extrinsic error. Recall that $\hat{\mathcal{P}}_0$ is the uniform distribution over m training prompts drawn i.i.d. from $\mathcal{P}_0$. Define $\bar{r}(x;h) \coloneqq \mathbb{E}_{\xi\sim\mathcal{P}_\xi}[r(x,g(x,\xi;h))]$. Since $r \in [a, a+\Delta]$, the function $\bar{r}$ is bounded in the same interval. The extrinsic error is then

$$\varepsilon_{\text{ext}}(h) = \mathbb{E}_{x\sim\mathcal{P}_0}[\bar{r}(x;h)] - \frac{1}{m}\sum_{i=1}^{m}\bar{r}(x_i;h),$$

which is the deviation of the sample mean of m i.i.d. bounded random variables from their population mean. By Hoeffding's inequality, for any $t_1 > 0$,

$$\mathbb{P}[\varepsilon_{\text{ext}}(h) \geq -t_1] \geq 1 - \exp\left(-\frac{2m\,t_1^2}{\Delta^2}\right).$$

To ensure this bound holds uniformly over all $h \in \mathcal{H}$, we distribute half of the total confidence budget, $\delta/2$, across hypotheses using the Solomonoff prior, allocating a failure probability of $\frac{1}{2}\delta \cdot P(h)$ to each h. Setting the Hoeffding tail equal to this budget and applying the union bound yields

$$\exp\left(-\frac{2m\,t_{1,h}^2}{\Delta^2}\right) = \tfrac{1}{2}\delta \cdot P(h).$$

Solving for $t_{1,h}$:

$$t_{1,h} = \Delta\sqrt{\frac{\log(1/P(h)) + \log(2/\delta)}{2m}}.$$

Since $\sum_{h\in\mathcal{H}} P(h) \leq 1$, the union bound gives

$$\mathbb{P}[\forall h \in \mathcal{H},\ \varepsilon_{\text{ext}}(h) \geq -t_{1,h}] \geq 1 - \tfrac{1}{2}\delta.$$

Bounding the intrinsic error. Conditional on the training prompts $x_1, \ldots, x_m$, the intrinsic error measures the deviation of a Monte Carlo estimate from its expectation. Because the noise vectors $\xi_{i,j}$ are drawn independently for each prompt, the mn random variables $r(x_i, g(x_i, \xi_{i,j}; h))$ are mutually independent and each bounded in $[a, a+\Delta]$. Note that Hoeffding's inequality requires only independence and uniform boundedness, not identical distributions — the reward distributions may differ across prompts. Applying Hoeffding's inequality over mn independent samples and distributing the remaining budget $\delta/2$ via the same prior-weighted union bound, we obtain

$$t_{2,h} = \Delta\sqrt{\frac{\log(1/P(h)) + \log(2/\delta)}{2mn}},$$

with

$$\mathbb{P}[\forall h \in \mathcal{H},\ \varepsilon_{\text{int}}(h) \geq -t_{2,h}] \geq 1 - \tfrac{1}{2}\delta.$$

Combining the bounds. Applying the union bound over the two events, with probability at least $1-\delta$, for all $h \in \mathcal{H}$ simultaneously:

$$\mathcal{R}(h) - \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n} r\big(x_i, g(x_i, \xi_j; h)\big) = \varepsilon_{\text{ext}}(h) + \varepsilon_{\text{int}}(h) \geq -t_{1,h} - t_{2,h}.$$

Let $B(h) \coloneqq \log(1/P(h)) + \log(2/\delta)$. Then

$$t_{1,h} + t_{2,h} = \Delta\left(\sqrt{\frac{B(h)}{2m}} + \sqrt{\frac{B(h)}{2mn}}\right) = \Delta\left(1 + \sqrt{\frac{1}{n}}\right)\sqrt{\frac{B(h)}{2m}}.$$

Upper-bounding the Kolmogorov complexity. Since the true Kolmogorov complexity $K(h)$ is uncomputable, we upper-bound $\log(1/P(h))$ using a practical compression size $C(h)$ in bits. Encoding h with a self-delimiting prefix code (Lotfi et al., 2023) gives

$$\log\frac{1}{P(h)} = \log Z + K(h)\log 2 \ \leq\ C(h)\log 2 + 2\log C(h).$$

Substituting into $B(h)$ and rearranging yields the final bound and completes the proof. □

B.3 Proof of Corollary 1

Proof. We decompose the gap between the population reward and the subsampled empirical estimate by inserting the *full-data* estimator $\hat{R}_{m,n}(h) \coloneqq \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n} r(x_i, g(x_i, \xi_{i,j}; h))$ as an intermediate quantity:

$$\mathcal{R}(h) - \hat{R}_{m_0,n}(h) \;=\; \underbrace{\big[\mathcal{R}(h) - \hat{R}_{m,n}(h)\big]}_{\text{full-sample gap}} \;+\; \underbrace{\big[\hat{R}_{m,n}(h) - \hat{R}_{m_0,n}(h)\big]}_{\varepsilon_{\mathrm{sub}}(h)\ \text{(subsampling gap)}}.$$

This decomposition allows us to invoke Theorem 1 directly as a black box for the first term, while the second term reduces to a clean Hoeffding bound on subsampling from a fixed finite population. We split the confidence budget to $2\delta/3$ for the first term and $\delta/3$ for the second term.

Bounding the full-sample gap. Applying Theorem 1 with confidence parameter $2\delta/3$, with probability at least $1 - 2\delta/3$, for all $h \in \mathcal{H}$ simultaneously,

$$\mathcal{R}(h) - \hat{R}_{m,n}(h) \;\geq\; -\Delta\left(\sqrt{\tfrac{1}{n}} + 1\right)\sqrt{\tfrac{B_3(h)}{2m}},$$

where $B_3(h) \coloneqq C(h)\log 2 + 2\log C(h) + \log(3/\delta)$.

Bounding the subsampling gap. Condition on the training prompts $\{x_i\}_{i=1}^{m}$ and the noise vectors $\{\xi_{i,j}\}_{i\in[m],\, j\in[n]}$. Define the per-prompt empirical averages

$$s_i(h) \;\coloneqq\; \tfrac{1}{n}\sum_{j=1}^{n} r\big(x_i, g(x_i, \xi_{i,j};\, h)\big) \;\in\; [a,\, a+\Delta], \qquad i \in [m].$$

Once we condition on the prompts and noise, the values $s_1(h), \ldots, s_m(h)$ are *fixed* bounded scalars. The full-data estimator is their mean, $\hat{R}_{m,n}(h) = \frac{1}{m}\sum_{i=1}^{m} s_i(h)$, and the subsampled estimator is the corresponding subsample mean, $\hat{R}_{m_0,n}(h) = \frac{1}{m_0}\sum_{k=1}^{m_0} s_{i_k}(h)$.

Because the indices $\{i_k\}_{k=1}^{m_0}$ are drawn uniformly at random from $[m]$ and are independent of h, the random variables $s_{i_1}(h), \ldots, s_{i_{m_0}}(h)$ are i.i.d. draws (with replacement) from the empirical distribution over $\{s_1(h), \ldots, s_m(h)\}$, with common mean $\hat{R}_{m,n}(h)$ and values in $[a, a+\Delta]$. Applying Hoeffding's inequality with budget $\delta/3$,

$$\mathbb{P}\left[\varepsilon_{\mathrm{sub}}(h) \;\geq\; -\Delta\sqrt{\tfrac{\log(3/\delta)}{2m_0}}\right] \;\geq\; 1 - \tfrac{1}{3}\delta.$$

Crucially, no union bound over $\mathcal{H}$ is required for this term: h is fixed before the subsample is drawn, so the bound holds for any *single* h and the penalty is independent of $C(h)$. Marginalizing over the conditioning preserves the unconditional probability statement.

Combining the bounds. By a union bound over the two events, with probability at least $1 - \delta$, any hypothesis h chosen before subsampling satisfies

$$\mathcal{R}(h) - \hat{R}_{m_0,n}(h) \;\geq\; -\Delta\sqrt{\tfrac{\log(3/\delta)}{2m_0}} \;-\; \Delta\left(\sqrt{\tfrac{1}{n}} + 1\right)\sqrt{\tfrac{B_3(h)}{2m}},$$

which is the stated bound. This completes the proof. □

B.4 Out-of-Distribution Extension

Setup. Let $\mathcal{Q}_0$ denote an arbitrary target prompt distribution, possibly different from $\mathcal{P}_0$. Let $\{x'_k\}_{k=1}^{m'} \overset{\text{iid}}{\sim} \mathcal{Q}_0$ denote an OOD evaluation corpus of size m', with independent Gumbel noise vectors $\xi'_{k,j} \overset{\text{iid}}{\sim} \mathcal{P}_\xi$ for $k \in [m']$, $j \in [n']$. The *target population reward* of a hypothesis h is

$$\mathcal{R}_{\mathcal{Q}}(h) \;=\; \mathbb{E}_{x\sim\mathcal{Q}_0}\big[\mathbb{E}_{\xi\sim\mathcal{P}_\xi}[r(x,\, g(x, \xi;\, h))]\big]\,,$$

and the corresponding empirical OOD reward is

$$\hat{R}^{\mathcal{Q}}_{m',n'}(h) \;=\; \frac{1}{m'n'}\sum_{k=1}^{m'}\sum_{j=1}^{n'} r(x'_k,\, g(x'_k, \xi'_{k,j};\, h)).$$

We assume a *labeled OOD oracle*: the RLVR verifier can evaluate $r(x', y)$ for any OOD prompt x' and generation y. This holds for verifiable-reward tasks whose verifiers extend across domains (e.g., symbolic math verifiers on new problems, unit tests on new programs, SQL execution on new databases).

Theorem 2 (PAC-Bayes bound for RLVR under distribution shift)**.** *Fix any $\delta \in (0,1)$ and let $h^\star \in \mathcal{H}$ be any hypothesis chosen independently of the OOD evaluation set $\{(x'_k, \xi'_{k,1}, \ldots, \xi'_{k,n'})\}_{k=1}^{m'}$. Then*

$$\mathbb{P}\left[\mathcal{R}_{\mathcal{Q}}(h^\star) \;\geq\; \hat{R}^{\mathcal{Q}}_{m',n'}(h^\star) - \Delta\left(\sqrt{\tfrac{1}{n'}} + 1\right)\sqrt{\tfrac{\log(2/\delta)}{2m'}}\right] \;\geq\; 1-\delta,$$

where the probability is taken over the OOD sample.

Proof of Theorem 2. Following the decomposition of Remark 1, we split the OOD generalization gap into extrinsic and intrinsic components:

$$\begin{aligned}&\mathcal{R}_{\mathcal{Q}}(h^\star) - \hat{R}^{\mathcal{Q}}_{m',n'}(h^\star)\\ &\quad= \underbrace{\mathbb{E}_{x\sim\mathcal{Q}_0}[\bar{r}(x;h^\star)] - \tfrac{1}{m'}\textstyle\sum_{k=1}^{m'}\bar{r}(x'_k;h^\star)}_{\varepsilon^{\mathcal{Q}}_{\mathrm{ext}}(h^\star)\text{ (extrinsic)}} + \underbrace{\tfrac{1}{m'}\textstyle\sum_{k=1}^{m'}\bar{r}(x'_k;h^\star) - \hat{R}^{\mathcal{Q}}_{m',n'}(h^\star)}_{\varepsilon^{\mathcal{Q}}_{\mathrm{int}}(h^\star)\text{ (intrinsic)}},\end{aligned}$$

where $\bar{r}(x;h) := \mathbb{E}_{\xi\sim\mathcal{P}_\xi}[r(x, g(x,\xi;h))]$. We bound each term separately.

Bounding the extrinsic error. Since $r \in [a, a+\Delta]$, the conditional expectation $\bar{r}(\cdot;h^\star)$ is likewise valued in $[a, a+\Delta]$. By hypothesis, $h^\star$ is a deterministic function of the in-distribution training data only, and is therefore independent of $\{x'_k\}_{k=1}^{m'}$. The sequence $\bar{r}(x'_1;h^\star), \ldots, \bar{r}(x'_{m'};h^\star)$ consists of m' i.i.d. bounded random variables with common mean $\mathbb{E}_{x\sim\mathcal{Q}_0}[\bar{r}(x;h^\star)]$. By Hoeffding's inequality, for any $t_1 > 0$,

$$\mathbb{P}\big[\varepsilon^{\mathcal{Q}}_{\mathrm{ext}}(h^\star) \geq -t_1\big] \;\geq\; 1 - \exp\!\left(-\tfrac{2m't_1^2}{\Delta^2}\right).$$

Allocating half of the confidence budget to this event and solving $\exp(-2m't_1^2/\Delta^2) = \delta/2$ yields

$$t_1 \;=\; \Delta\sqrt{\tfrac{\log(2/\delta)}{2m'}}.$$

No union bound over $\mathcal{H}$ is required: the bound is stated for the specific (pre-selected) $h^\star$. This is the key structural difference from the proof of Theorem 1 in Appendix B.2 and is what eliminates the $C(h)\log 2 + 2\log C(h)$ term.

Bounding the intrinsic error. Condition on the OOD prompts $x'_1, \ldots, x'_{m'}$. The OOD noise vectors $\{\xi'_{k,j}\}_{k\in[m'],j\in[n']}$ are drawn independently across all (k,j) and are independent of both $\{x'_k\}$ and $h^\star$. Therefore the $m'n'$ random variables $\{r(x'_k, g(x'_k, \xi'_{k,j}; h^\star))\}_{k,j}$ are mutually independent conditional on $\{x'_k\}$, with each bounded in $[a, a+\Delta]$; reward distributions may differ across prompts, but Hoeffding's inequality requires only independence and uniform boundedness. For any $t_2 > 0$,

$$\mathbb{P}\big[\varepsilon^{\mathcal{Q}}_{\mathrm{int}}(h^\star) \geq -t_2 \,\big|\, x'_1, \ldots, x'_{m'}\big] \;\geq\; 1 - \exp\!\left(-\tfrac{2m'n't_2^2}{\Delta^2}\right).$$

Marginalizing over $\{x'_k\}$ preserves the unconditional probability statement. Allocating the remaining $\delta/2$ and solving yields

$$t_2 \;=\; \Delta\sqrt{\tfrac{\log(2/\delta)}{2m'n'}}.$$

Again, no union bound over $\mathcal{H}$ is required.

Combining the bounds. By a union bound over the two events, with probability at least $1-\delta$,

$$\mathcal{R}_{\mathcal{Q}}(h^\star) - \hat{R}^{\mathcal{Q}}_{m',n'}(h^\star) \;\geq\; -t_1 - t_2.$$

Factoring $\sqrt{\log(2/\delta)/(2m')}$ out of both terms gives

$$t_1 + t_2 \;=\; \Delta\left(1 + \sqrt{\tfrac{1}{n'}}\right)\sqrt{\tfrac{\log(2/\delta)}{2m'}} \;=\; \Delta\left(\sqrt{\tfrac{1}{n'}} + 1\right)\sqrt{\tfrac{\log(2/\delta)}{2m'}},$$

which yields the stated inequality and completes the proof. □

Table 2: Training configuration for the teacher RLVR and on-policy distillation stages. Dashes (—) indicate the setting is not applicable or follows the framework default.

Setting	Teacher Training	On-Policy Distillation
Adapter		
Type	LoRA	TinyLoRA
Rank	64	16
Trainable coefficients $\lvert v \rvert$	—	16
Target modules	—	all linear
Optimization		
Optimizer	Adam ($\beta_1 = 0.9$, $\beta_2 = 0.95$, $\epsilon = 10^{-8}$)	AdamW
Final learning rate	1×10^{-5} (math/code/general); 1×10^{-4} (sql)	1×10^{-4}
Warmup steps	0	0
Weight decay	0	0
Training steps	200	200
Batch size (groups per step)	64	64
Loss		
Reward	verifier-defined (per-domain)	per-token KL to teacher
Policy loss	importance sampling	importance sampling
Rollouts		
Multi-turn	sql only (`max_turns` = 5)	inherits domain setting
Group size (rollouts per prompt)	8	8
Sampling temperature	1.0	1.0
`top_p`, `top_k`	1.0, -1	1.0, -1
Max generation length	4096 tokens	4096 tokens
Inference engine	Tinker default	vLLM
Seed	0	0

C Experimental Setup and Hyperparameters

We detail the training and evaluation settings used in our experiments. Our main results use Qwen3.5-4B as the backbone for both teachers and students; an additional ablation uses Qwen3.5-2B. Compression applies to the learned updates, while the full backbone is retained at inference. Whether comparable accuracy–description-length tradeoffs hold for larger models or other model families remains open. The training and evaluation pipelines are provided in the supplementary material.

C.1 Teacher Training

Each of the four per-domain teachers is a rank-64 LoRA adapter on top of Qwen3.5-4B, trained via the Tinker[2] reinforcement-learning cookbook against verifier-defined rewards on the same dataset that will later be used for student rollouts. We keep the recipe structure identical across domains. We use GRPO-style rollouts Shao et al. (2024) at temperature 1.0, group size 8, 64 groups per training batch, and a maximum generation length of 4096 tokens. We conduct a hyperparameter sweep over learning rates $\{10^{-5}, 5 \times 10^{-5}, 10^{-4}\}$, and select the checkpoint with the highest training reward. We train for 200 optimizer steps for all runs, which is enough to achieve convergence.

We apply multi-turn RL on the text-to-SQL domain. In each turn, the model issues `<tool_call>`-wrapped SQL probes, observes execution results, and ultimately commits an answer inside a `<solution>` block, with up to 5 turns per trajectory. The reward is $+1$ if the committed query's result-set matches ground truth, 0 if it executes but is wrong, and -1 for format violations (no `<solution>` tag, leaked nested markup, etc.). Math and general domains use `math_verify` and `sympy` parsing of `\boxed{}`-style answers (binary $\{0, 1\}$, with a hard 30 s timeout per response). Code domain executes the generated code against unit tests via pytest in a sandboxed subprocess (binary $\{0, 1\}$, 30 s timeout). The full configuration is summarized in Table 2.

C.2 On-Policy Distillation Training

All runs share the hyperparameters in Table 2 and were trained for 200 update steps each, which is sufficient to achieve convergence.

[2] `https://github.com/thinking-machines-lab/tinker-cookbook`

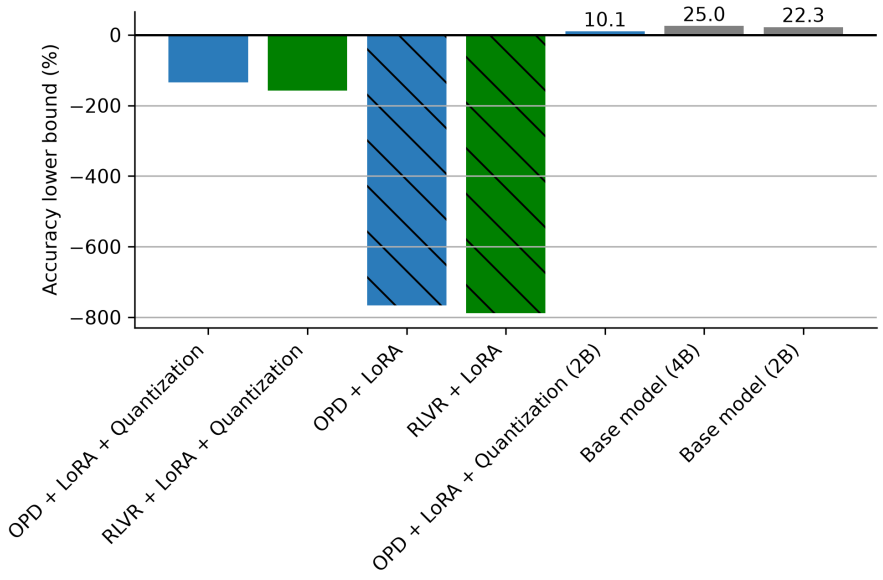

Figure 6: Extended ablation on Math using standard rank-1 LoRA ($\delta = 0.05$). Quantized configurations use $k = 5$; unquantized configurations use FP32. Models use Qwen3.5-4B unless marked 2B. Colored bars report accuracy lower bounds; gray bars report empirical base-model accuracies. Negative bounds are vacuous.

D Extended Ablation Study

We extend Figure 5 to isolate the contribution of the adapter parameterization. On Math with Qwen3.5-4B, we replace TinyLoRA with standard rank-1 LoRA and evaluate both training objectives (OPD and direct RLVR) under five-level quantization and FP32 precision. Together with the TinyLoRA results in Figure 5, these comparisons cover all eight combinations of training objective, adapter type, and precision. We additionally evaluate a quantized rank-1 LoRA student with Qwen3.5-2B to examine whether reducing backbone size suffices to obtain a useful bound.

Adapter parameterization. As shown in Figure 6, rank-1 LoRA yields vacuous bounds for Qwen3.5-4B in all four configurations. With quantization, the bounds are -1.35 under OPD and -1.58 under direct RLVR; in FP32, they are -7.66 and -7.89, respectively. Quantization substantially reduces the description-length penalty, and OPD improves the bound under either precision setting, but neither compensates sufficiently for the larger description length of standard LoRA updates. In contrast, the corresponding TinyLoRA configurations all yield positive bounds (Figure 5).

Smaller backbone. The quantized Qwen3.5-2B student trained with OPD and rank-1 LoRA achieves a positive lower bound of 10.1%, but this remains below the 2B base model's empirical accuracy of 22.3%. Reducing backbone size therefore does not suffice, in this configuration, to obtain a bound exceeding the empirical base-model baseline. This comparison highlights the importance of controlling the description length of the learned update.

These ablations support Progressive RLVR as a practical pipeline for obtaining tight compression bounds. OPD improves bound tightness, while the quantized TinyLoRA configuration already exceeds the empirical base-model baseline under direct RLVR.

E Encoding Scheme and Description Length Accounting

The compression-based PAC-Bayes penalty in Theorem 1 depends on $C(h)$, the description length in bits of the hypothesis h under a fixed encoding scheme. The bound is valid only if $C(h)$ is sufficient to fully reconstruct h given a *hypothesis-independent* decoder. In this appendix, we make the encoding scheme used in our experiments explicit. We enumerate every ingredient of the student

model, declare which ingredients are fixed across the hypothesis space $\mathcal{H}$ (and therefore cost zero per-h bits), describe the encoding of the per-h information, and tabulate the resulting $C(h)$ used in our reported bounds.

E.1 Decoder Specification

The following ingredients are fixed before training and are identical for every $h \in \mathcal{H}$. They are part of the decoder specification and contribute zero bits to $C(h)$.

- **TinyLoRA adapter placement.** Adapters are inserted at a fixed set of $L = 249$ modules of `Qwen3.5-4B`: the language-model head, and for each transformer block, the attention projection matrices and the MLP up/gate/down projections.
- **TinyLoRA truncated SVD basis (U, Σ, V).** The bases are a deterministic function of the frozen pre-trained weights W_0 Morris et al. (2026) and need not be transmitted.
- **TinyLoRA random projection matrices $\{P_i\}_{i=1}^{u}$.** Each $P_i \in \mathbb{R}^{r\times r}$ is generated from a single master seed (42) shared across all $h \in \mathcal{H}$. The seed is part of the decoder spec.
- **LoRA scaling α/r, sampling temperature τ, maximum sequence length T.** Single values fixed for the entire hypothesis space.
- **Number of trainable components u and rank r.** Both u and r are fixed in advance.

Because all of the above are hypothesis-independent, the only per-h quantities that the decoder must receive are the trainable vector $\{v^{(\ell)}\}_{\ell=1}^{L}$ with $v^{(\ell)} \in \mathbb{R}^u$, giving a total of $N = u \cdot L = 3984$ float components per h, together with the small header described next.

E.2 Description Length Breakdown

We quantize each component of v using a single *global* codebook $\mathcal{C}(h) = \{c_1, \ldots, c_k\} \subset \mathbb{R}$ that is shared across all L adapter sites for a given hypothesis h. The codebook entries are fit per-h during training and therefore must be transmitted as part of $C(h)$.

Index stream. Each component $v_i^{(\ell)}$ is encoded as the index $j \in \{1, \ldots, k\}$ of its codebook entry. Let $n_j(h)$ denote the empirical count of level j across the N components. We use the universal Krichevsky–Trofimov (KT) arithmetic code Krichevsky & Trofimov (1981) on the index stream, whose self-decodable length is

$$L_{\mathrm{KT}}(h) \;=\; \left\lceil -\log_2 \frac{\prod_{j=1}^{k} \Gamma\big(n_j(h)+\tfrac{1}{2}\big)\,/\Gamma\big(\tfrac{1}{2}\big)^{k}}{\Gamma\big(N+\tfrac{k}{2}\big)\,/\Gamma\big(\tfrac{k}{2}\big)} \right\rceil .$$

The KT code matches the empirical entropy $H(h) = -\sum_j (n_j/N)\log_2(n_j/N)$ of the index stream up to a $\frac{k-1}{2}\log_2 N$ overhead and requires no separate frequency header. As a sanity check, we also report (i) the empirical arithmetic code at length $\lceil N \cdot H(h) \rceil + 2$ bits augmented by an explicit $(k-1)\lceil \log_2(N+1) \rceil$-bit frequency header and (ii) a uniform fixed-length code at $N\lceil \log_2 k \rceil = 3N$ bits with no header.

Codebook header. Each of the k codebook entries c_j is transmitted as an fp16 scalar, contributing $16k$ bits per h.

The total per-h description length is therefore

$$C(h_k) \;=\; L_{\mathrm{KT}}(h_k) \;+\; \underbrace{16k}_{\text{codebook}} \tag{1}$$

The self-delimiting prefix-code overhead $2\log C(h)$ that converts $C(h)\log 2$ to $\log(1/P(h))$ in Theorem 1 is already embedded in the bound's penalty and is not included in equation 1.

F Out-of-domain Evaluation

We now extend the experiments to evaluating the OOD performance of the models trained via Progressive RLVR. For each pair of domains ($\mathcal{P}_0$, $\mathcal{Q}_0$), we evaluate the quantized student trained on $\mathcal{P}_0$ against disjoint corpus from $\mathcal{Q}_0$. In Figure 7, we show cross-domain bounds for each ordered pair of domains: a student trained on the source (column) is evaluated on a labeled corpus from the target (row), and the cell shows the resulting bound (with improvement over the base model in parentheses). All twelve off-diagonal pairs yield non-vacuous bounds, and nine show improvements over the base model, indicating that RLVR specialists could preserve useful capability outside their training domain.

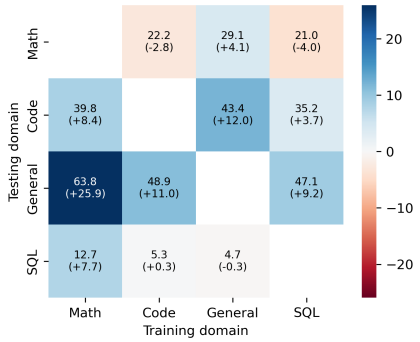

Figure 7: Cross-domain generalization bounds.

We find that the transfer pattern is markedly asymmetric. Math is the strongest source domain, with positive transfer to every other target and the largest single improvement of any pair (+25.9 points on General). In contrast, Math is also the hardest target. Other sources transfer neutrally or negatively, with the worst case being SQL→Math at -4.0 points. This asymmetry suggests that RLVR on mathematical reasoning yields broadly transferable representations, while reasoning patterns from other domains do not recover the symbolic precision mathematics requires. General reasoning is the easiest target, resulting in gain from every source, while SQL is the most isolated, with all cross-domain bounds below 13% accuracy.

G Generalization Bound for the Rewards on Text-to-SQL Tasks

The text-to-SQL bound reported in Figure 3 is computed on task accuracy rather than the tri-level reward ($r \in \{-1, 0, 1\}$) used in training. Here, we re-calculate the bound under the raw reward. Theorem 1 applies directly with $\Delta = 2$, yielding a bound on the expected reward.

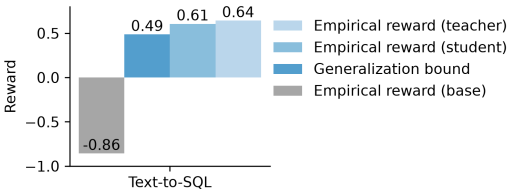

Figure 8: Non-vacuous accuracy bound for text-to-SQL at $\delta = 0.05$.

We show the result in Figure 8. We obtain a non-vacuous accuracy bound of 0.49 for the trained model, exceeding the base model's empirical accuracy of -0.86 and lying within 0.12 of the student's empirical accuracy of 0.61. This tightness is consistent with the binary-reward bounds for the other three domains (Figure 3).