MANIFOLD BANDITS: BAYESIAN CURRICULUM LEARNING OVER THE LATENT GEOMETRY OF LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

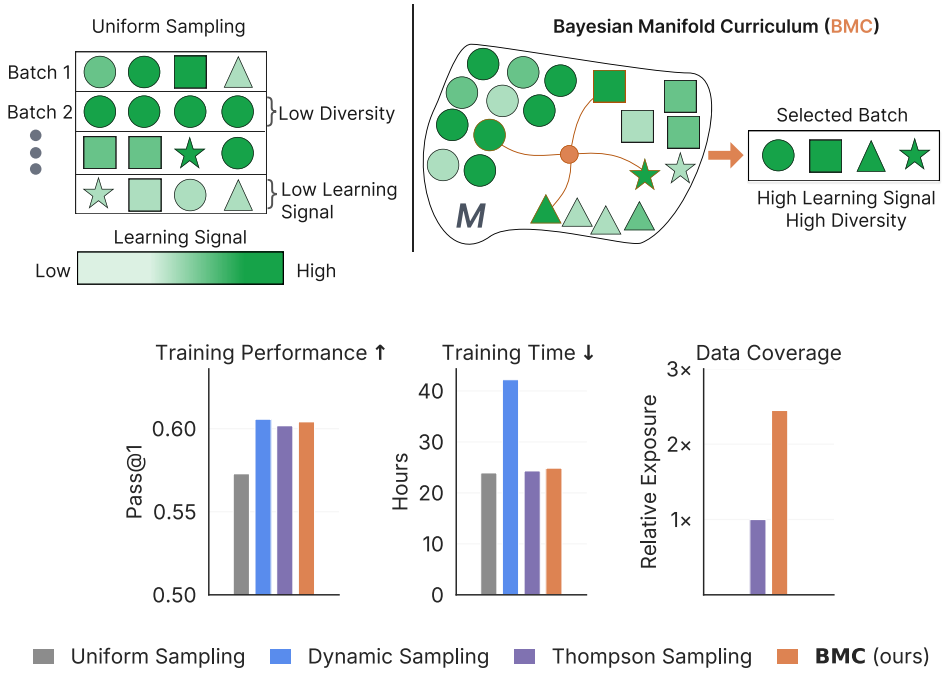

Figure 1. **Bayesian Manifold Curriculum (BMC).** BMC uses Bayesian learning over the policy model's latent task manifold to construct training batches from productive and diverse problem types. BMC improves the trade-off between training efficiency and broad data coverage.

ABSTRACT

Reinforcement learning (RL) is a central approach for improving reasoning capabilities in large language models (LLMs), where training efficiency depends critically on how problems are sampled during optimization. Existing adaptive curriculum learning methods typically prioritize prompts of intermediate difficulty, treating problem selection as a standard bandit problem with independent arms and overlooking the structured, heterogeneous nature of the task space. In this work, we instead frame problem sampling as a manifold-structured bandit problem with endogenous non-stationarity, where relationships between tasks and evolving model capabilities jointly influence learning dynamics. To operationalize this perspective, we introduce *Bayesian Manifold Curriculum (BMC)*, a structure-aware framework that leverages latent representations of LLMs to organize problems into a hierarchical task tree and applies Bayesian learning to guide sampling. Across multiple training setups and evaluation benchmarks, we find that different sampling strategies induce non-trivial tradeoffs between productivity (learning signal), diversity (coverage of the task manifold), and utility (evaluation relevance). These results show that prioritizing difficulty alone is insufficient to ensure strong downstream performance and highlight the importance of incorporating structure and diversity in problem sampling.

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.

1 INTRODUCTION

Reinforcement Learning (RL) has become a powerful paradigm for improving the reasoning capabilities of Large Language Models (LLMs). A dominant class of methods, including GRPO (Shao et al., 2024), DAPO (Yu et al., 2025), GSPO (Zheng et al., 2025a), and related approaches, uses group-relative optimization: multiple rollouts are generated from each prompt, and policy updates are driven by relative differences in their rewards.

This makes training efficiency depend critically on reward variation. When all rollouts for a prompt receive identical rewards, the relative advantage collapses to zero and the prompt contributes no policy gradient signal. Dynamic Sampling (Yu et al., 2025) addresses this by resampling until batches contain prompts with non-zero reward variance, but incurs substantial wall-clock overhead from repeated batch generation. To reduce this overhead, recent adaptive curriculum methods attempt to predict which prompts are likely to yield informative learning signals as the model evolves (Chen et al., 2025; Wang et al., 2025b; Qu et al., 2025; Zeng et al., 2025). These approaches can be viewed as implicitly or explicitly framing prompt selection as a non-stationary bandit problem over individual prompts.

However, this standard bandit framing misses key aspects of LLM training dynamics. First, unlike typical non-stationary bandit settings where reward drift is exogenous (driven by external factors), here non-stationarity is *endogenous*: as the model learns, the reward landscape changes as a consequence of the model's own updates and sampling decisions. Second, prompts are not independent. Problems of similar type often share reasoning patterns, so observations from one prompt can inform beliefs about related prompts. Third, effective sampling must balance difficulty with diversity: focusing only on prompts with high immediate learning signal can narrow coverage of heterogeneous and imbalanced training data.

These observations suggest that problem selection should be viewed not as an unstructured bandit problem, but as a structured bandit problem over a latent task manifold shaped by the model itself. Rather than relying on external labels or human-defined problem groups, curriculum strategies should adapt both to the model's evolving notion of difficulty and to its latent organization of the problem space, since human-defined task categories need not match the model's internal notion of problem type.

To operationalize this perspective, we introduce *Bayesian Manifold Curriculum (BMC)*, a structure-aware framework for problem sampling in RL training of LLMs. BMC consists of two main components: **Latent Task Trees**, which approximate the model-induced task manifold from latent embeddings, and **Bayesian decision-making** over this hierarchy, which enables adaptive sampling and information sharing across related prompts. Empirically, we analyze BMC and baselines along three axes—productivity, diversity, and utility—showing that problem sampling cannot be reduced to difficulty alone.

2 PRELIMINARIES

Reinforcement Learning for LLMs. We consider group-relative RL methods such as GRPO (Shao et al., 2024) and its derivatives, in which multiple responses $\{y_i\}_{i=1}^{G} \sim \pi_\theta(\cdot \mid x)$ are sampled per prompt and advantages are constructed from within-group reward statistics. Prompts with zero reward variance provide no relative learning signal, motivating adaptive problem selection.

Structured and Non-Stationary Bandits. Classical bandit formulations treat arms independently, whereas *structured bandits* exploit relationships between arms to share information (Lattimore & Szepesvári, 2020; Slivkins, 2024). Non-stationary bandits further allow rewards to drift over time, typically due to exogenous environmental change. Problem selection for LLM training combines both challenges: prompts are related through the model's latent task geometry, and their learning signal changes endogenously as the model updates and the sampling policy changes.

Latent Geometry. Representations learned by LLMs exhibit structured organization in high-dimensional space, often well-approximated by lower-dimensional geometric structure (Valeriani et al., 2023; Modell et al., 2025). We adopt this perspective to model relationships between problems using the model's own representations, inducing a latent task manifold over prompts.

3 LATENT TASK TREES

Building on the structured and endogenous bandit setting described above, we model problem selection as a decision process over a latent task manifold induced by the model's representations. Rather than treating prompts as independent arms, we view them as embedded within a shared latent space, where proximity reflects semantic or functional similarity. This perspective enables information sharing across related problems and provides a natural notion of structure for guiding exploration.

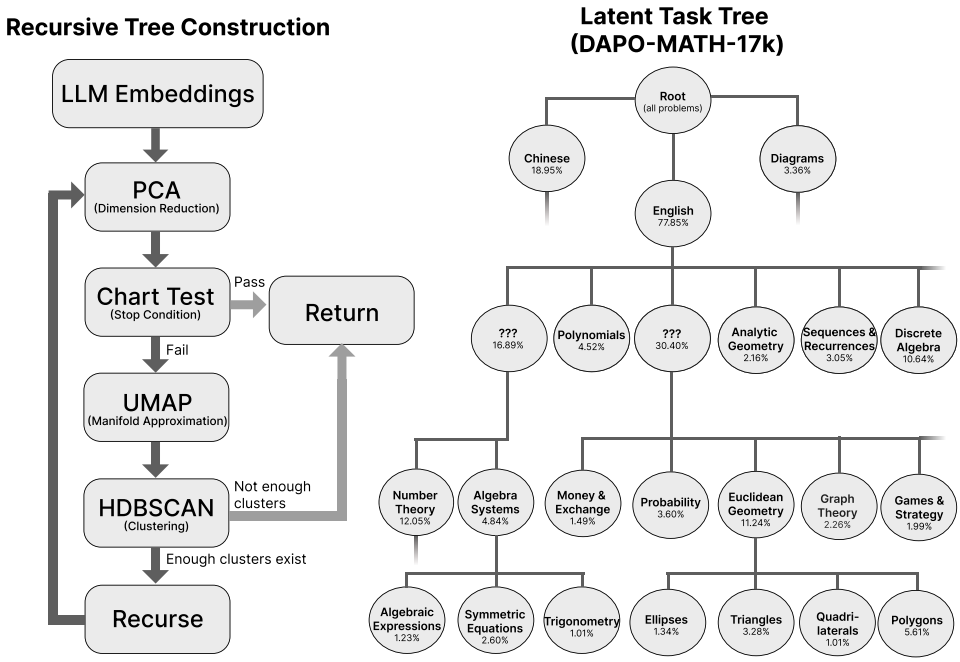

Figure 2. **Latent Task Tree construction and example structure. Left:** Recursive construction from model embeddings using local dimensionality reduction, chart-like stopping criteria, and density-based clustering. **Right:** A partial tree induced from `DAPO-MATH-17K`, revealing fine-grained and imbalanced problem types in the model's latent task space. Node percentages indicate the fraction of prompts assigned to each node.

However, latent manifolds in LLMs are complex, heterogeneous, and difficult to model globally (Li & Sarwate, 2025a). To address this, we propose a recursive, local manifold approximation procedure that produces a *Latent Task Tree*: a discrete hierarchy that captures fine-grained structure while remaining computationally tractable.

To construct the tree, we first extract a normalized mean embedding for each prompt from an intermediate layer of the model. Each node is recursively decomposed by standardizing its embeddings, reducing dimensionality with PCA using the smallest number of components explaining 95% of the variance, projecting with UMAP (McInnes et al., 2018), and clustering with HDBSCAN (Campello et al., 2013). Each detected cluster becomes a child node, producing a hierarchical partition of the latent task manifold.

Recursion is halted using a *Chart Test*, which asks whether the current node is already well-approximated by a single local coordinate system. The test is performed on PCA-reduced embeddings, before UMAP projection, and checks for low intrinsic dimensionality and local connectivity using Two-NN (Facco et al., 2017) and a k-nearest-neighbor graph. Recursion also stops when clustering produces at most one valid child or clusters fall below a minimum size threshold.

General Applicability. The construction uses only model embeddings and standard unsupervised procedures, requiring no annotations or external embedding models; it is therefore applicable across datasets, domains, and modalities. See Appendix F to see demonstrative visualizations.

4 BAYESIAN MANIFOLD CURRICULUM

We now introduce **Bayesian Manifold Curriculum (BMC)**, a structure-aware framework for problem selection over the Latent Task Tree. BMC uses the tree to allocate training effort across problem types while sharing information between related prompts at multiple levels of abstraction.

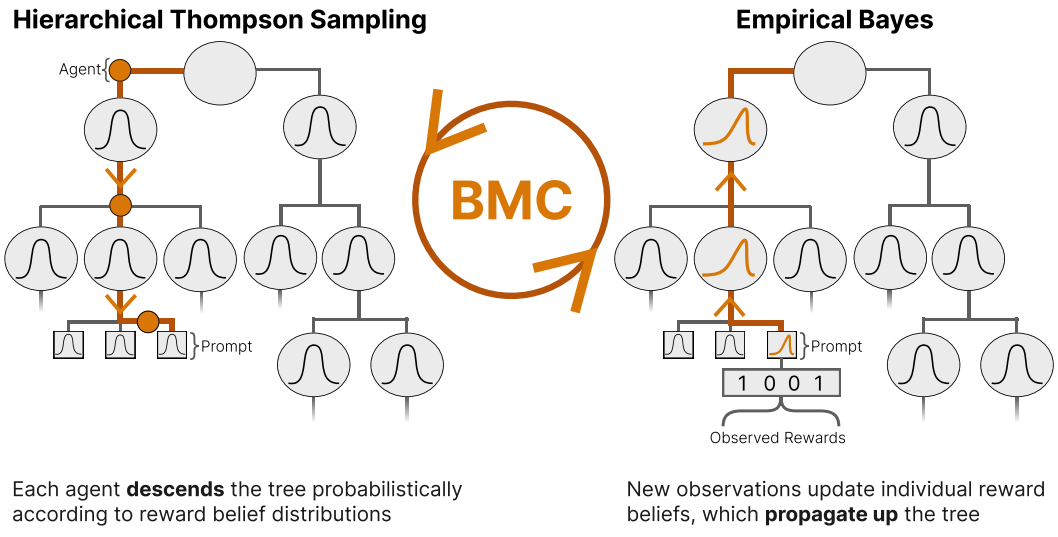

Figure 3. **Bayesian Manifold Curriculum (BMC). Left:** Each batch element independently descends the Latent Task Tree via hierarchical Thompson sampling, recursively selecting child nodes until reaching a prompt; the figure shows one agent trajectory. **Right:** After rollout rewards are observed, prompt-level beliefs are updated and propagated upward using empirical Bayes, allowing observations from individual prompts to inform higher-level regions of the task manifold.

BMC selects prompts to maximize a *learning signal* while maintaining coverage over the latent task manifold. Following prior work (Foster et al., 2025; Zhang et al., 2025a; Zeng et al., 2025), we instantiate this signal as normalized reward variance across k rollouts:

$$y_i = \phi(\mathrm{Var}(\{r_{i1}, r_{i2}, \ldots, r_{ik}\}))\,, \tag{1}$$

where $\phi : \mathbb{R} \to [0, 1]$ is an optional normalization function. We provide additional justification for this choice in Appendix M.1.

The method has three components: top-down selection via hierarchical Thompson sampling, non-stationary belief modeling via Bayesian filtering, and bottom-up belief propagation via empirical Bayes.

4.1 TOP-DOWN PROBLEM SELECTION (HIERARCHICAL THOMPSON SAMPLING)

To construct a batch, each sample acts as an independent agent that descends the Latent Task Tree from the root. At a node with children $c \in \mathcal{C}$, the agent samples a learning-signal estimate

$$\tilde{y}_c \sim \mathcal{N}(\mu_c, \sigma_c^2), \quad \forall c \in \mathcal{C}, \tag{2}$$

and selects $c^* = \arg\max_{c \in \mathcal{C}} \tilde{y}_c$. Repeating this procedure guides the agent from coarse regions of the task space to a leaf prompt selected for training.

Unlike prior bandit-based curricula that often couple an entire batch to a single task region (Wang et al., 2025b; Chen et al., 2025; Do et al., 2025), BMC samples one trajectory per batch element. This allows a batch to span multiple regions of the latent manifold while still focusing probability mass on regions believed to be productive, balancing *diversity* and *focus* within each update.

4.2 Non-Stationary Belief Modeling (Bayesian Filtering)

Each leaf maintains a Gaussian belief over its learning signal, $y_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$, where μ_i estimates expected productivity and σ_i^2 captures uncertainty. Because a prompt's usefulness changes as the model learns, beliefs must incorporate new evidence while discounting outdated observations.

We use a lightweight Bayesian filtering update inspired by Gaussian inference with unknown mean and variance. Rather than maintaining the full conjugate posterior, each leaf tracks (μ_i, σ_i^2) and an effective evidence weight λ_i. Given an observed learning signal $y_i^{(t)}$, we compute normalized surprise

$$s_i^{(t)} = \frac{y_i^{(t)} - \mu_i^{(t-1)}}{\sigma_i^{(t-1)}}, \tag{3}$$

which measures deviation relative to current uncertainty. Large surprise downweights stale evidence through $\lambda_i^{\text{eff}} = \lambda_i^{(t-1)} \exp[-(s_i^{(t)})^2]$, after which the mean is updated as

$$\mu_i^{(t)} = \frac{\lambda_i^{\text{eff}} \mu_i^{(t-1)} + y_i^{(t)}}{\lambda_i^{\text{eff}} + 1}. \tag{4}$$

The variance is updated analogously, contracting under repeated agreement while increasing under surprising observations or prolonged lack of feedback. Full update equations and intuition are provided in Appendix M.2.2.

4.3 Bottom-Up Belief Propagation (Empirical Bayes)

After leaf beliefs are updated, BMC propagates information upward through the Latent Task Tree using empirical Bayes. Although only leaves correspond to individual prompts, internal nodes summarize their children and guide future top-down selection. This induces coupling between related tasks: observing one prompt immediately updates its ancestors, thereby influencing the sampling probabilities of other prompts in the same region.

For each internal node, we aggregate child beliefs with a precision-weighted estimator. Given child beliefs $\{(\mu_i, \sigma_i^2)\}_{i=1}^K$, we use weights $w_i = (\sigma_i^2 + \tau_p)^{-1}$, where τ_p is a subtree-level heterogeneity correction that increases parent uncertainty when child means disagree. The parent belief is then

$$\mu_p = \frac{\sum_i w_i \mu_i}{\sum_i w_i}, \qquad \sigma_p^2 = \frac{1}{\sum_i w_i}. \tag{5}$$

This allows confident, consistent children to shape parent beliefs while preventing heterogeneous regions from becoming overconfident. Beliefs are propagated recursively from leaves to root after each batch update, enabling local observations to inform broader regions of the task space. Full aggregation details are provided in Appendix M.2.3.

5 Experiments

We evaluate BMC on RL reasoning tasks using `DAPO-Math-17K` (Yu et al., 2025) with `Qwen3-4B-Base` and `Qwen3-8B-Base` (Yang et al., 2025). We use GSPO (Zheng et al., 2025a) and GRPO (Guo et al., 2025) as backbone policy optimization algorithms, with all methods implemented in `verl` (Sheng et al., 2024). All runs share identical initialization, reward function, and hyperparameters; only the prompt sampling strategy and associated rollout allocation vary.

We analyze sampling strategies along three axes: *productivity*, measuring learning signal and training-set learning speed; *diversity*, measuring coverage and information sharing across the task manifold; and *utility*, measuring performance on held-out evaluations. These objectives are not necessarily aligned: strategies that maximize training productivity or coverage need not yield the strongest downstream performance.

Baselines and ablations. We compare against ***Uniform sampling***, the default strategy; ***Dynamic sampling***, which resamples until prompts have non-zero reward variance (Yu et al., 2025; Zheng et al., 2025a); ***Difficulty Only (Thompson Sampling)***, which models prompts independently and

selects those with high posterior samples (Thompson, 1933; Daniel et al., 2018); and ***Tree only***, which samples through the Latent Task Tree using fixed beliefs. To control compute, all adaptive methods and ablations start from a shared step-136 checkpoint, where sampling switches from round-robin to the corresponding strategy. Difficulty Only serves as both an ablation and a representative independent-arm adaptive curriculum baseline, reflecting the common pattern of prioritizing prompts by estimated productivity without modeling relationships between problem types (Wang et al., 2025b; Chen et al., 2025; Qu et al., 2025); Tree only isolates structured sampling without adaptive belief updates.

5.1 Training Efficiency (Productivity)

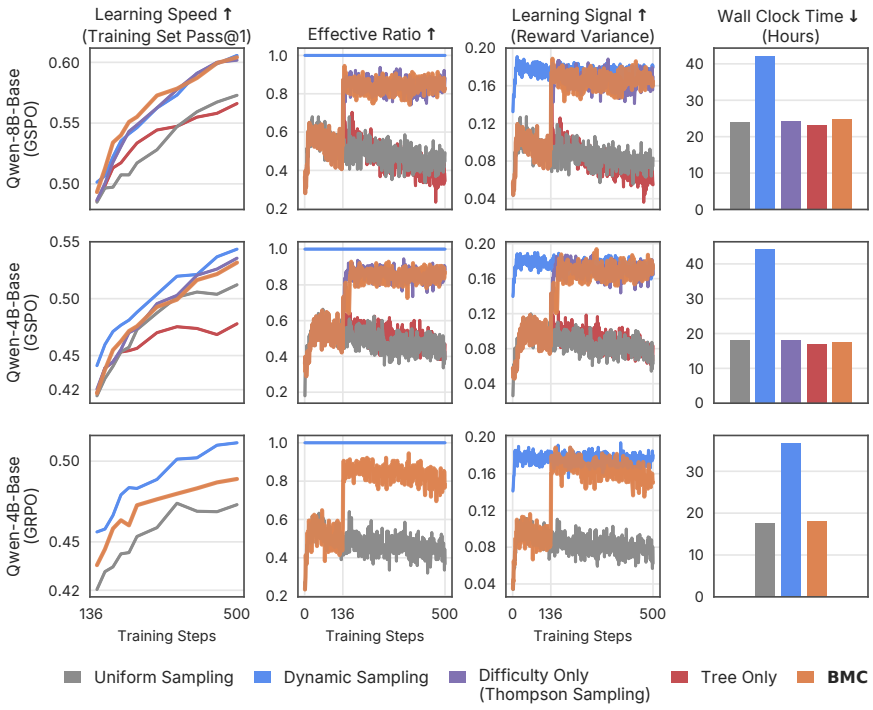

Figure 4. **Productivity and training efficiency across sampling strategies.** Dynamic sampling achieves the highest effective ratio and learning signal, but incurs substantially higher wall-clock time due to repeated batch regeneration. In contrast, Difficulty Only and *Bayesian Manifold Curriculum (BMC)* achieve comparable learning speed while maintaining training times close to uniform sampling. Uniform sampling exhibits lower learning signal and slower training progress.

We measure *productivity* by how efficiently a sampler produces useful learning signal and how quickly this translates into training-set progress. Specifically, we track pass@1 on the full training set, the *effective ratio* of sampled prompts with non-zero reward variance, and the average reward variance.

Figure 4 shows that higher reward variance generally corresponds to higher effective ratios and faster improvement on the training set. Dynamic sampling maintains an effective ratio of 1 by construction, but does so at substantially higher wall-clock cost. In contrast, Difficulty Only and BMC recover much of dynamic sampling's learning-speed advantage while maintaining training times close to uniform sampling; on the 8B model, both methods match dynamic sampling in training-set progress without incurring its resampling overhead. However, similar training-set pass@1 does not imply that the same problems are being solved: sampling strategies with comparable productivity can allocate training effort to different regions of the training distribution.

Diversity. Due to space limitations, we defer the full Diversity axis analysis to Appendix B. Even when productivity is similar, BMC changes which regions of the latent task manifold receive training effort, balancing exposure to underrepresented problem types with productive sampling. We quantify

this behavior using *rarity-weighted exposure* and *structure gain*, which measure coverage of rarer problem types and shared structure in learning signal (difficulty) across related problem types, respectively.

5.2 Evaluation Performance (Utility)

We now evaluate how sampling behavior transfers to held-out benchmarks. While productivity and diversity describe learning dynamics within the training distribution, evaluation performance reflects *utility*: relevance to downstream tasks. Because both training and evaluation data are heterogeneous, problem types can differ substantially in utility, so methods that maximize learning signal or coverage need not uniformly improve benchmark performance. We focus the evaluation analysis on the 8B GSPO setting, where all sampling strategies are compared under the same training setup.

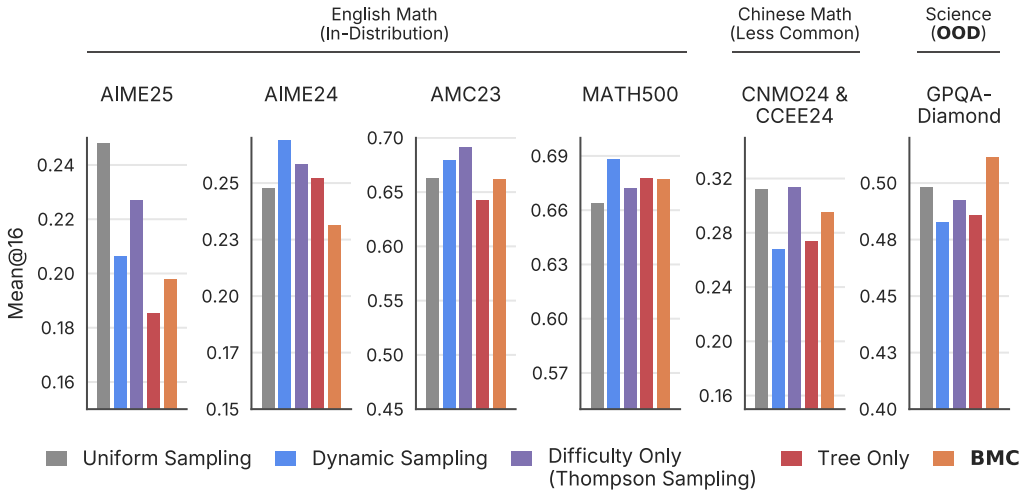

Figure 5. **Final evaluation performance across benchmark categories.** No single sampling strategy dominates across all evaluations, reflecting tradeoffs between productivity, diversity, and utility. Sampling methods with similar learning signal can produce different evaluation outcomes due to differences in problem-type exposure. BMC improves performance on the OOD GPQA-Diamond benchmark despite not uniformly outperforming other methods on standard math evaluations, suggesting that different curricula induce different capability profiles.

We evaluate on standard English math benchmarks (AIME, AMC, MATH500), less-represented but related Chinese math benchmarks (CNMO24, CCEE24), and the out-of-domain science benchmark GPQA-Diamond; dataset details are provided in Appendix L.4. As shown in Figure 5, evaluation outcomes vary substantially across sampling strategies. **Dynamic sampling** maximizes effective ratio and learning signal, but its first-come-first-serve batch construction can skip excess prompts, which is harmful in imbalanced datasets. **Difficulty Only** avoids this resampling artifact and performs better on less-represented benchmarks, but remains type-agnostic because it selects globally over individual prompts. **Tree Only** improves diversity without prioritizing informative problems, limiting overall performance. **Uniform sampling** provides broad but unstructured coverage, occasionally benefiting specific benchmarks while lacking systematic allocation to either productive or underrepresented problem types. **BMC** jointly increases learning signal and task-manifold coverage, yielding strong OOD GPQA-Diamond performance, but with tradeoffs on standard math evaluations.

5.3 Utility-Aware Sampling

The preceding results leave an important ambiguity: when sampling strategies produce different downstream evaluation outcomes, do these differences simply reflect better or worse training productivity, or do they depend on *which* problem types receive training effort? To separate these explanations, we introduce **BMC-T**, a utility-aware targeting variant of BMC.

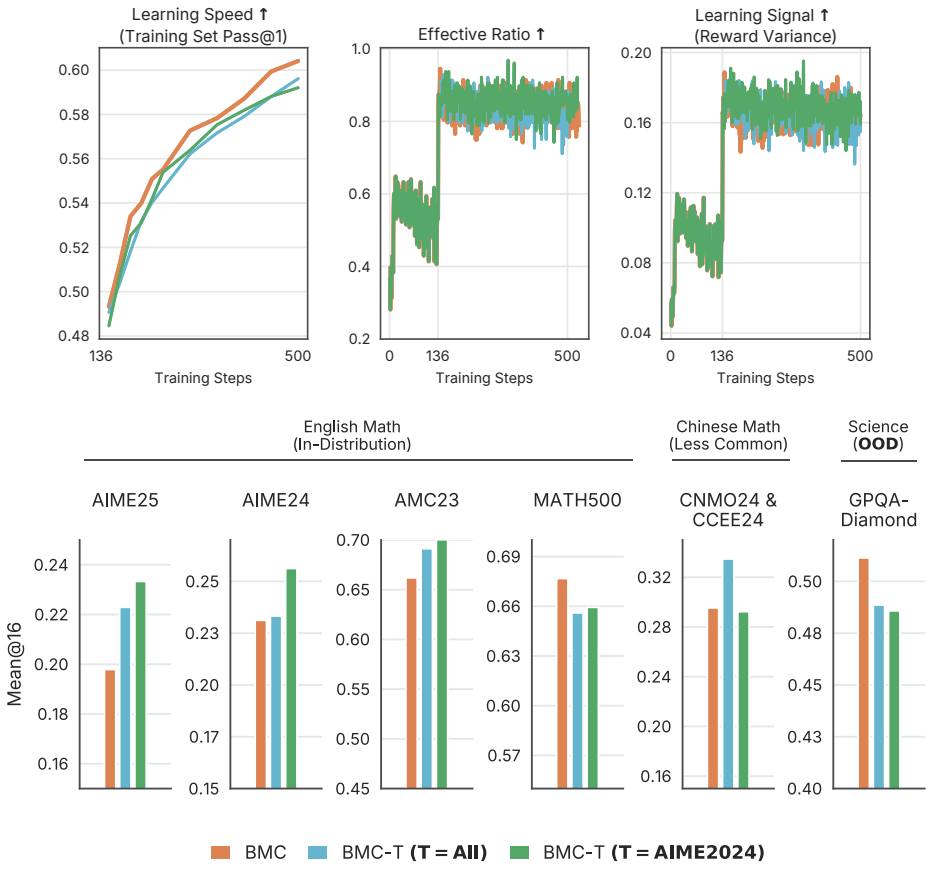

Figure 6. **Similar training productivity can produce different downstream capability profiles.** BMC and BMC-T achieve comparable learning speed, effective ratio, and learning signal, yet changing the target distribution shifts downstream evaluation performance. Targeting AIME2024 improves AIME-style English competition math, whereas targeting the full evaluation mixture shifts performance toward a broader set of evaluations, including stronger Chinese math performance.

BMC-T is given a target distribution T, consisting of held-out examples that represent a desired downstream capability or evaluation distribution. These target examples are never sampled for training. Instead, BMC-T places training and target examples within the same Latent Task Tree and uses their latent overlap as a coarse proxy for downstream relevance. During top-down sampling, this utility signal biases BMC toward productive regions of the training distribution that are also more relevant to T. Full implementation details are provided in Appendix M.8. Discussions on the relationship between utility-awareness and evaluation fairness are discussed in Appendix I.

We evaluate BMC-T on `DAPO-Math-17K` with `Qwen3-8B-Base` and GSPO using two target distributions: the full evaluation mixture, BMC-T (T = All), and AIME2024 alone, BMC-T (T = AIME2024). We compare both against standard BMC while tracking the same productivity and downstream evaluation metrics.

As shown in Figure 6, BMC and the two BMC-T variants exhibit similar training-set pass@1, effective ratio, and learning signal, yet produce different downstream evaluation profiles. Thus, **higher training productivity does not imply better downstream evaluation performance**. Evaluation outcomes depend not only on how much learning signal a curriculum extracts, but also on the downstream relevance (utility) of the problem *types* receiving training effort. BMC-T makes this distinction explicit: changing which regions of the training distribution are emphasized can shift the resulting capability profile without materially changing training productivity.

Key Takeaways

Human–Policy Task Mismatch. Human-defined task categories need not match the policy model's internal organization of the problem space. Latent Task Trees recover a finer-grained, model-induced task structure directly from the policy's representations.

Type matters. Sampling strategies with similar productivity can yield different downstream capability profiles because they allocate training effort to different problem types.

Difficulty alone is insufficient. Maximizing learning signal does not guarantee superior downstream performance. A sampling strategy with higher productivity can still be outperformed by a less productive strategy due to differences in problem type exposure.

Utility is distinct from productivity and diversity. Not all problem types contribute equally to downstream evaluations; even strategies that improve both productivity and diversity can underperform when they allocate training effort to problem types with lower evaluation relevance.

Additional Evidence and Robustness. The main text focuses on the central relationship between productivity, problem-type coverage (diversity), and downstream utility; the appendix provides complementary analyses of robustness, generality, and mechanism. We study latent-structure drift and periodic tree reconstruction in Appendix C, additional comparisons against external curriculum-learning baselines in Appendix D, tree-construction stability in Appendix F, tree-construction sensitivity in Appendix G, and the domain generality of BMC through medical-domain RL experiments in Appendix E. Full diversity and information-sharing diagnostics are provided in Appendix B, while implementation and evaluation-protocol details for BMC-T are provided in Appendix M.8 and Appendix I. Finally, we provide the **Appendix Overview**, for convenience.

6 Related Work

Recent RL training methods for LLMs increasingly recognize that prompts of intermediate difficulty provide stronger learning signal. LILO (Foster et al., 2025) formalizes learnability through rollout-level success variance, while DAPO (Yu et al., 2025) introduces dynamic sampling to retain prompts with non-zero reward variance at the cost of additional wall-clock time. Subsequent work improves or approximates this filtering process using saturation filters, partial rollouts, difficulty-aware selection, temporal reward dynamics, Bayesian difficulty modeling, or learned value models (Zhang et al., 2025b; An et al., 2025; Zhang et al., 2025a; Shi et al., 2026; Jiang et al., 2025; Zheng et al., 2025b; Zeng et al., 2025; Gao et al., 2025). These methods primarily target prompt difficulty or productivity, but do not explicitly model hierarchical relationships between problem types.

A complementary line frames problem selection as a non-stationary bandit problem, using bandit schedulers to prioritize prompts or prompt groups according to estimated learning productivity (Wang et al., 2025b; Chen et al., 2025; Do et al., 2025; Anonymous, 2026; Qu et al., 2025; Shen et al., 2026; Hu et al., 2025). In contrast, BMC learns structural relationships between prompts directly from the model's latent representation space and performs hierarchical Bayesian decision-making over both regions and individual prompts. This separates problem *difficulty* from problem *type*, treating them as complementary factors in automated curriculum learning. See Appendix A for an extended discussion of related work beyond curriculum learning for LLMs.

7 Conclusion

We introduced *Bayesian Manifold Curriculum (BMC)*, a structure-aware framework for problem sampling in reinforcement learning for LLMs that combines latent geometry with Bayesian learning. Our results suggest that effective problem sampling cannot be reduced to difficulty alone: it involves a multi-objective tradeoff between productivity, diversity, and utility. More broadly, BMC connects the *manifold hypothesis* (Fefferman et al., 2013) with the spirit of the *Bitter Lesson* (Sutton, 2019), using structure derived from the model's own representations to guide search and learning over the problem space. As RL training is applied to increasingly heterogeneous datasets and reward signals, framing problem selection as a structured bandit problem with endogenous non-stationarity offers a principled direction for scalable curriculum design.

AI USE STATEMENT

In this work, AI was heavily involved in the brainstorming that led to the final iteration(s) of this work. It was also relied on heavily to write the paper—specifically using an iterative process rather than having it provide generations outright. This iterative process also doubly served as validation, ensuring that the rewrites were in line with our intended narrative. In contrast, the vast majority of the literature review was done manually, though AI did help provide arguments and summaries for some works (particularly contemporary works). Most of the code (which involved modifying the verl library) was also written manually, though AI was used to check and correct developments. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies how to improve the computational efficiency of reinforcement learning with verifiable rewards (RLVR) for large language models by reallocating training compute toward prompts that are expected to provide higher learning signal. While reducing costs of RLVR indeed helps democratize the technology, it may also increase the risk of downstream misuse (*e.g.*, scalable generation of persuasive or harmful content) if deployed without appropriate safeguards. Our method does *not* introduce new reward sources, data, or deployment pathways beyond standard RLVR, and operates only at training time; we therefore view its risks as similar to those of other efficiency improvements in LLM post-training.

REPRODUCIBILITY STATEMENT

To support reproducibility, we specify the public datasets, policy optimization algorithms, exact LLMs, evaluation procedures, and hyperparameters used in our experiments throughout the main text and appendix. Additional experimental settings and implementation details are provided in Appendix L and Appendix M. We also provide a supplementary ZIP archive containing our code, built on `verl`, together with a `README.md` describing environment setup, data preparation, and the configurations required to reproduce the main experiments.

REFERENCES

AI-MO. Aimo validation amc dataset. https://huggingface.co/datasets/AI-MO/aimo-validation-amc, 2024. Hugging Face dataset.

Chenxin An, Zhihui Xie, Xiaonan Li, Lei Li, Jun Zhang, Shansan Gong, Ming Zhong, Jingjing Xu, Xipeng Qiu, Mingxuan Wang, and Lingpeng Kong. Polaris: A post-training recipe for scaling reinforcement learning on advanced reasoning models, 2025. URL https://hkunlp.github.io/blog/2025/Polaris.

Anonymous. Adaptive curriculum learning for rlhf with influence-based cluster bandits. OpenReview preprint, 2026. URL https://openreview.net/forum?id=8HvWBamUkS.

Dana Arad, Aaron Mueller, and Yonatan Belinkov. Saes are good for steering – if you select the right features. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 10252–10270. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.519. URL http://dx.doi.org/10.18653/v1/2025.emnlp-main.519.

Abdus Salam Azad, Izzeddin Gur, Jasper Emhoff, Nathaniel Alexis, Aleksandra Faust, Pieter Abbeel, and Ion Stoica. Clutr: Curriculum learning via unsupervised task representation learning, 2023. URL https://arxiv.org/abs/2210.10243.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren,

Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report, 2025. URL https://arxiv.org/abs/2511.21631.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pp. 41–48, 2009.

Vinay Praneeth Boda and Prashanth L. A. Correlated bandits or: How to minimize mean-squared error online, 2019. URL https://arxiv.org/abs/1902.02953.

Ricardo J. G. B. Campello, Davoud Moulavi, and Joerg Sander. Density-based clustering based on hierarchical density estimates. In Jian Pei, Vincent S. Tseng, Longbing Cao, Hiroshi Motoda, and Guandong Xu (eds.), *Advances in Knowledge Discovery and Data Mining*, pp. 160–172, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. ISBN 978-3-642-37456-2.

Shuyang Cao and Lu Wang. Hibrids: Attention with hierarchical biases for structure-aware long document summarization, 2022. URL https://arxiv.org/abs/2203.10741.

Xiaoyin Chen, Jiarui Lu, Minsu Kim, Dinghuai Zhang, Jian Tang, Alexandre Piché, Nicolas Gontier, Yoshua Bengio, and Ehsan Kamalloo. Self-evolving curriculum for llm reasoning. *arXiv preprint arXiv:2505.14970*, 2025.

Jorge Zhoujun Cheng, Shibo Hao, Tianyang Liu, Fan Zhou, Yutao Xie, Feng Yao, Yuexin Bian, Nilabjo Dey, Yonghao Zhuang, Yuheng Zha, et al. Revisiting reinforcement learning for llm reasoning from a cross-domain perspective. *Advances in Neural Information Processing Systems*, 38, 2026.

Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models, 2023. URL https://arxiv.org/abs/2309.08600.

J Russo Daniel, Van Roy Benjamin, Kazerouni Abbas, Osband Ian, and Wen Zheng. A tutorial on thompson sampling. *Foundations and Trends® in Machine Learning*, 11(1):1–99, 2018.

Michael Dennis, Natasha Jaques, Eugene Vinitsky, Alexandre Bayen, Stuart Russell, Andrew Critch, and Sergey Levine. Emergent complexity and zero-shot transfer via unsupervised environment design, 2021. URL https://arxiv.org/abs/2012.02096.

Rebecca DerSimonian and Nan Laird. Meta-analysis in clinical trials. *Controlled Clinical Trials*, 7 (3):177–188, 1986.

Dai Do, Manh Nguyen, Svetha Venkatesh, and Hung Le. Sparft: Self-paced reinforcement fine-tuning for large language models. *arXiv preprint arXiv:2508.05015*, 2025.

Diego Doimo, Alessandro Serra, Alessio Ansuini, and Alberto Cazzaniga. The representation landscape of few-shot learning and fine-tuning in large language models, 2024. URL https://arxiv.org/abs/2409.03662.

Elena Facco, Maria d'Errico, Alex Rodriguez, and Alessandro Laio. Estimating the intrinsic dimension of datasets by a minimal neighborhood information. *Scientific reports*, 7(1):12140, 2017.

Charles Fefferman, Sanjoy Mitter, and Hariharan Narayanan. Testing the manifold hypothesis, 2013. URL https://arxiv.org/abs/1310.0425.

Thomas Foster, Anya Sims, Johannes Forkel, Mattie Fellows, and Jakob Foerster. Learning to reason at the frontier of learnability. *arXiv preprint arXiv:2502.12272*, 2025.

Zhaolin Gao, Joongwon Kim, Wen Sun, Thorsten Joachims, Sid Wang, Richard Yuanzhe Pang, and Liang Tan. Prompt curriculum learning for efficient llm post-training, 2025. URL https://arxiv.org/abs/2510.01135.

Josep Ginebra and Murray K Clayton. Response surface bandits. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 57(4):771–784, 1995.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan,

Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Alex Graves, Marc G Bellemare, Jacob Menick, Remi Munos, and Koray Kavukcuoglu. Automated curriculum learning for neural networks. In *international conference on machine learning*, pp. 1311–1320. Pmlr, 2017.

Zhengyao Gu, Jonathan Light, Raul Astudillo, Ziyu Ye, Langzhou He, Henry Peng Zou, Wei Cheng, Santiago Paternain, Philip S. Yu, and Yisong Yue. Actor-curator: Co-adaptive curriculum learning via policy-improvement bandits for rl post-training, 2026. URL https://arxiv.org/abs/2602.20532.

Dong Guo et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/pdf/2501.12948.

Samarth Gupta, Shreyas Chaudhari, Gauri Joshi, and Osman Yağan. Multi-armed bandits with correlated arms. *IEEE Transactions on Information Theory*, 67(10):6711–6732, October 2021. ISSN 1557-9654. doi: 10.1109/tit.2021.3081508. URL http://dx.doi.org/10.1109/TIT.2021.3081508.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models, 2024. URL https://arxiv.org/abs/2301.04104.

Kevin Han, Yuhang Zhou, Mingze Gao, Gedi Zhou, Serena Li, Abhishek Kumar, Xiangjun Fan, Weiwei Li, and Lizhu Zhang. Ebpo: Empirical bayes shrinkage for stabilizing group-relative policy optimization, 2026. URL https://arxiv.org/abs/2602.05165.

Nicklas Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control, 2024. URL https://arxiv.org/abs/2310.16828.

Zirui He, Haiyan Zhao, Yiran Qiao, Fan Yang, Ali Payani, Jing Ma, and Mengnan Du. Saif: A sparse autoencoder framework for interpreting and steering instruction following of language models, 2025. URL https://arxiv.org/abs/2502.11356.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset, 2021. URL https://arxiv.org/abs/2103.03874.

Joey Hong, Branislav Kveton, Sumeet Katariya, Manzil Zaheer, and Mohammad Ghavamzadeh. Deep hierarchy in bandits, 2022a. URL https://arxiv.org/abs/2202.01454.

Joey Hong, Branislav Kveton, Manzil Zaheer, and Mohammad Ghavamzadeh. Hierarchical bayesian bandits, 2022b. URL https://arxiv.org/abs/2111.06929.

Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning, 2011. URL https://arxiv.org/abs/1112.5745.

Zengjie Hu, Jiantao Qiu, Tianyi Bai, Haojin Yang, Binhang Yuan, Qi Jing, Conghui He, and Wentao Zhang. Vade: Variance-aware dynamic sampling via online sample-level difficulty estimation for multimodal rl, 2025. URL https://arxiv.org/abs/2511.18902.

Hyeon Jeon, Jeongin Park, Sungbok Shin, and Jinwook Seo. Stop misusing t-sne and umap for visual analytics, 2025. URL https://arxiv.org/abs/2506.08725.

Guochao Jiang, Wenfeng Feng, Guofeng Quan, Chuzhan Hao, Yuewei Zhang, Guohua Liu, and Hao Wang. Vcrl: Variance-based curriculum reinforcement learning for large language models. *arXiv preprint arXiv:2509.19803*, 2025.

Xia Jiang and Rong J. B. Zhu. Empirical bayesian multi-bandit learning, 2025. URL https://arxiv.org/abs/2510.26284.

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, 2020. URL https://arxiv.org/abs/2009.13081.

Yi Jing, Zao Dai, Jinwu Hu, Zijun Yao, Lei Hou, Juanzi Li, and Xiaozhi Wang. Guiding llm post-training data engineering with model internals from sparse autoencoders, 2026. URL https://arxiv.org/abs/2605.27354.

Subhash Kantamneni, Joshua Engels, Senthooran Rajamanoharan, Max Tegmark, and Neel Nanda. Are sparse autoencoders useful? a case study in sparse probing, 2025. URL https://arxiv.org/abs/2502.16681.

Hyeonjin Kim, Hangyeol Jung, Heechan Yun, Sungjun Yun, and Dong-Jun Han. Disentangled sparse representations for concept-separated diffusion unlearning, 2026. URL https://arxiv.org/abs/2605.12122.

Andreas Kirsch, Joost van Amersfoort, and Yarin Gal. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning, 2019. URL https://arxiv.org/abs/1906.08158.

M Kumar, Benjamin Packer, and Daphne Koller. Self-paced learning for latent variable models. *Advances in neural information processing systems*, 23, 2010.

Finnian Lattimore, Tor Lattimore, and Mark D. Reid. Causal bandits: Learning good interventions via causal inference, 2016. URL https://arxiv.org/abs/1606.03203.

Tor Lattimore and Csaba Szepesvári. *Bandit Algorithms*. Cambridge University Press, 2020. URL https://www.cambridge.org/core/books/bandit-algorithms/8E39FD004E6CE036680F90DD0C6F09FC.

Jin Hwa Lee, Thomas Jiralerspong, Lei Yu, Yoshua Bengio, and Emily Cheng. Geometric signatures of compositionality across a language model's lifetime, 2025. URL https://arxiv.org/abs/2410.01444.

Ge Lei and Samuel J. Cooper. Layerwise recall and the geometry of interwoven knowledge in llms, 2025. URL https://arxiv.org/abs/2502.10871.

Xin Li and Anand Sarwate. Unraveling the localized latents: Learning stratified manifold structures in llm embedding space with sparse mixture-of-experts. *arXiv preprint arXiv:2502.13577*, 2025a.

Xin Li and Anand Sarwate. Unraveling the localized latents: Learning stratified manifold structures in llm embedding space with sparse mixture-of-experts, 2025b. URL https://arxiv.org/abs/2502.13577.

Zhongyang Li, Ziyue Li, and Tianyi Zhou. Routing manifold alignment improves generalization of mixture-of-experts llms. *arXiv preprint arXiv:2511.07419*, 2025.

Che Liu, Haozhe Wang, Jiazhen Pan, Zhongwei Wan, Yong Dai, Fangzhen Lin, Wenjia Bai, Daniel Rueckert, and Rossella Arcucci. Beyond distillation: Pushing the limits of medical llm reasoning with minimalist rule-based rl, 2025. URL https://arxiv.org/abs/2505.17952.

Zhu Liu, Cunliang Kong, Ying Liu, and Maosong Sun. Fantastic semantics and where to find them: Investigating which layers of generative llms reflect lexical semantics, 2024. URL https://arxiv.org/abs/2403.01509.

Pan Lu, Ran Gong, Shibiao Jiang, Liang Qiu, Siyuan Huang, Xiaodan Liang, and Song-Chun Zhu. Inter-gps: Interpretable geometry problem solving with formal language and symbolic reasoning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 6774–6786, 2021.

Michael Luo, Sijun Tan, Roy Huang, Ameen Patel, Alpay Ariyak, Qingyang Wu, Xiaoxiang Shi, Rachel Xin, Colin Cai, Maurice Weber, Ce Zhang, Li Erran Li, Raluca Ada Popa, and Ion Stoica. Deepcoder: A fully open-source 14b coder at o3-mini level. https://pretty-radio-b75.notion.site/DeepCoder-A-Fully-Open-Source-14B-Coder-at-O3-mini-Level-1cf81902c14680b3bee5eb349 2025. Notion Blog.

Ilia Mahrooghi, Aryo Lotfi, and Emmanuel Abbe. Goldilocks rl: Tuning task difficulty to escape sparse rewards for reasoning, 2026. URL https://arxiv.org/abs/2602.14868.

Tambet Matiisen, Avital Oliver, Taco Cohen, and John Schulman. Teacher-student curriculum learning, 2017. URL https://arxiv.org/abs/1707.00183.

Maxwell-Jia. Aime 2024 dataset. https://huggingface.co/datasets/Maxwell-Jia/AIME_2024, 2024. Hugging Face dataset.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

Adam J Mersereau, Paat Rusmevichientong, and John N Tsitsiklis. A structured multiarmed bandit problem and the greedy policy. *IEEE Transactions on Automatic Control*, 54(12):2787–2802, 2009.

Alexander Modell, Patrick Rubin-Delanchy, and Nick Whiteley. The origins of representation manifolds in large language models, 2025. URL https://arxiv.org/abs/2505.18235.

Sagnik Mukherjee, Lifan Yuan, Dilek Hakkani-Tur, and Hao Peng. Reinforcement learning finetunes small subnetworks in large language models, 2025. URL https://arxiv.org/abs/2505.11711.

Corey J. Nolet, Victor Lafargue, Edward Raff, Thejaswi Nanditale, Tim Oates, John Zedlewski, and Joshua Patterson. Bringing umap closer to the speed of light with gpu acceleration, 2021. URL https://arxiv.org/abs/2008.00325.

Bruno A Olshausen and David J Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996.

Bruno A Olshausen and David J Field. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision research*, 37(23):3311–3325, 1997.

OpenCompass. Livemathbench. https://huggingface.co/datasets/opencompass/LiveMathBench, 2024. Hugging Face dataset, v202412.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. Medmcqa : A large-scale multi-subject multi-choice dataset for medical domain question answering, 2022. URL https://arxiv.org/abs/2203.14371.

Sandeep Pandey, Deepak Agarwal, Deepayan Chakrabarti, and Vanja Josifovski. Bandits for taxonomies: A model-based approach. In *Proceedings of the 2007 SIAM international conference on data mining*, pp. 216–227. SIAM, 2007.

Letitia Parcalabescu and Anette Frank. On measuring faithfulness or self-consistency of natural language explanations, 2024. URL https://arxiv.org/abs/2311.07466.

Kiho Park, Yo Joong Choe, Yibo Jiang, and Victor Veitch. The geometry of categorical and hierarchical concepts in large language models, 2025. URL https://arxiv.org/abs/2406.01506.

Gabor Petnehazi and Bernadett Aradi. Hercules: Hierarchical embedding-based recursive clustering using llms for efficient summarization, 2025. URL https://arxiv.org/abs/2506.19992.

Valentina Pyatkin, Saumya Malik, Victoria Graf, Hamish Ivison, Shengyi Huang, Pradeep Dasigi, Nathan Lambert, and Hannaneh Hajishirzi. Generalizing verifiable instruction following, 2025. URL https://arxiv.org/abs/2507.02833.

Yun Qu, Qi Wang, Yixiu Mao, Vincent Tao Hu, Björn Ommer, and Xiangyang Ji. Can prompt difficulty be online predicted for accelerating rl finetuning of reasoning models? *arXiv preprint arXiv:2507.04632*, 2025.

Yun Qu, Qi Wang, Yixiu Mao, Heming Zou, Yuhang Jiang, Weijie Liu, Clive Bai, Kai Yang, Yangkun Chen, Saiyong Yang, and Xiangyang Ji. Small generalizable prompt predictive models can steer efficient rl post-training of large reasoning models, 2026. URL https://arxiv.org/abs/2602.01970.

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

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. Gpqa: A graduate-level google-proof q&a benchmark, 2023. URL https://arxiv.org/abs/2311.12022.

Michael Robinson, Sourya Dey, and Tony Chiang. Token embeddings violate the manifold hypothesis, 2025. URL https://arxiv.org/abs/2504.01002.

Baturay Saglam, Paul Kassianik, Blaine Nelson, Sajana Weerawardhena, Yaron Singer, and Amin Karbasi. Large language models encode semantics and alignment in linearly separable representations, 2026. URL https://arxiv.org/abs/2507.09709.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D. Manning. Raptor: Recursive abstractive processing for tree-organized retrieval, 2024. URL https://arxiv.org/abs/2401.18059.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay, 2016. URL https://arxiv.org/abs/1511.05952.

Or Shafran, Shaked Ronen, Omri Fahn, Shauli Ravfogel, Atticus Geiger, and Mor Geva. From directions to regions: Decomposing activations in language models via local geometry, 2026. URL https://arxiv.org/abs/2602.02464.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y.K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Qianli Shen, Daoyuan Chen, Yilun Huang, Zhenqing Ling, Yaliang Li, Bolin Ding, and Jingren Zhou. Bots: A unified framework for bayesian online task selection in llm reinforcement finetuning. *arXiv preprint arXiv:2510.26374*, 2025.

Qianli Shen, Daoyuan Chen, Yilun Huang, Zhenqing Ling, Yaliang Li, Bolin Ding, and Jingren Zhou. Bots: A unified framework for bayesian online task selection in llm reinforcement finetuning, 2026. URL https://arxiv.org/abs/2510.26374.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

Taiwei Shi, Yiyang Wu, Linxin Song, Tianyi Zhou, and Jieyu Zhao. Efficient reinforcement finetuning via adaptive curriculum learning, 2026. URL https://arxiv.org/abs/2504.05520.

Oscar Skean, Md Rifat Arefin, Yann LeCun, and Ravid Shwartz-Ziv. Does representation matter? exploring intermediate layers in large language models, 2024. URL https://arxiv.org/abs/2412.09563.

Oscar Skean, Md Rifat Arefin, Dan Zhao, Niket Patel, Jalal Naghiyev, Yann LeCun, and Ravid Shwartz-Ziv. Layer by layer: Uncovering hidden representations in language models, 2025. URL https://arxiv.org/abs/2502.02013.

Aleksandrs Slivkins. Introduction to multi-armed bandits, 2024. URL https://arxiv.org/abs/1904.07272.

Richard Sutton. The bitter lesson. http://www.incompleteideas.net/IncIdeas/BitterLesson.html, 2019. Accessed: 2026-05-04.

Xinyu Tang, Zhenduo Zhang, Yurou Liu, Wayne Xin Zhao, Zujie Wen, Zhiqiang Zhang, and Jun Zhou. Towards high data efficiency in reinforcement learning with verifiable reward, 2025. URL https://arxiv.org/abs/2509.01321.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, Pier Giuseppe Sessa, Aakanksha Chowdhery, Adam Roberts, Aditya Barua, Alex Botev, Alex Castro-Ros, Ambrose Slone, Amélie Héliou, Andrea Tacchetti, Anna Bulanova, Antonia Paterson, Beth Tsai, Bobak Shahriari, Charline Le Lan, Christopher A. Choquette-Choo, Clément Crepy, Daniel Cer, Daphne Ippolito, David Reid, Elena Buchatskaya, Eric Ni, Eric Noland, Geng Yan, George Tucker, George-Christian Muraru, Grigory Rozhdestvenskiy, Henryk Michalewski, Ian Tenney, Ivan Grishchenko, Jacob Austin, James Keeling, Jane Labanowski, Jean-Baptiste Lespiau, Jeff Stanway, Jenny Brennan, Jeremy Chen, Johan Ferret, Justin Chiu, Justin Mao-Jones, Katherine Lee, Kathy Yu, Katie Millican, Lars Lowe Sjoesund, Lisa Lee, Lucas Dixon, Machel Reid, Maciej Mikuła, Mateo Wirth, Michael Sharman, Nikolai Chinaev, Nithum Thain, Olivier Bachem, Oscar Chang, Oscar Wahltinez, Paige Bailey, Paul Michel, Petko Yotov, Rahma Chaabouni, Ramona Comanescu, Reena Jana, Rohan Anil, Ross McIlroy, Ruibo Liu, Ryan Mullins, Samuel L Smith, Sebastian Borgeaud, Sertan Girgin, Sholto Douglas, Shree Pandya, Siamak Shakeri, Soham De, Ted Klimenko, Tom Hennigan, Vlad Feinberg, Wojciech Stokowiec, Yu hui Chen, Zafarali Ahmed, Zhitao Gong, Tris Warkentin, Ludovic Peran, Minh Giang, Clément Farabet, Oriol Vinyals, Jeff Dean, Koray Kavukcuoglu, Demis Hassabis, Zoubin Ghahramani, Douglas Eck, Joelle Barral, Fernando Pereira, Eli Collins, Armand Joulin, Noah Fiedel, Evan Senter, Alek Andreev, and Kathleen Kenealy. Gemma: Open models based on gemini research and technology, 2024. URL https://arxiv.org/abs/2403.08295.

William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.

Lucrezia Valeriani, Diego Doimo, Francesca Cuturello, Alessandro Laio, Alessio Ansuini, and Alberto Cazzaniga. The geometry of hidden representations of large transformer models. *Advances in Neural Information Processing Systems*, 36:51234–51252, 2023.

Huy V. Vo, Vasil Khalidov, Timothée Darcet, Théo Moutakanni, Nikita Smetanin, Marc Szafraniec, Hugo Touvron, Camille Couprie, Maxime Oquab, Armand Joulin, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. Automatic data curation for self-supervised learning: A clustering-based approach, 2024. URL https://arxiv.org/abs/2405.15613.

Lev S. Vygotsky. *Mind in Society: The Development of Higher Psychological Processes*. Harvard University Press, Cambridge, MA, 1978.

Shaobo Wang, Xuan Ouyang, Tianyi Xu, Yuzheng Hu, Jialin Liu, Guo Chen, Tianyu Zhang, Junhao Zheng, Kexin Yang, Xingzhang Ren, Dayiheng Liu, and Linfeng Zhang. Opus: Towards efficient and principled data selection in large language model pre-training in every iteration, 2026. URL https://arxiv.org/abs/2602.05400.

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. Reinforcement learning for reasoning in large language models with one training example, 2025a. URL https://arxiv.org/abs/2504.20571.

Zhenting Wang, Guofeng Cui, Yu-Jhe Li, Kun Wan, and Wentian Zhao. Dump: Automated distribution-level curriculum learning for rl-based llm post-training. *arXiv preprint arXiv:2504.09710*, 2025b.

Runzhe Wu, Ankur Samanta, Ayush Jain, Scott Fujimoto, Jeongyeol Kwon, Ben Kretzu, Youliang Yu, Kaveh Hassani, Boris Vidolov, and Yonathan Efroni. Imbalanced gradients in rl post-training of multi-task llms, 2025. URL https://arxiv.org/abs/2510.19178.

Zhenda Xie, Yixuan Wei, Huanqi Cao, Chenggang Zhao, Chengqi Deng, Jiashi Li, Damai Dai, Huazuo Gao, Jiang Chang, Liang Zhao, et al. mhc: Manifold-constrained hyper-connections. *arXiv preprint arXiv:2512.24880*, 2025.

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

Ningyuan Yang, Weihua Du, Weiwei Sun, Sean Welleck, and Yiming Yang. Gradalign: Gradient-aligned data selection for llm reinforcement learning, 2026. URL https://arxiv.org/abs/2602.21492.

yentinglin. Aime 2025 dataset. https://huggingface.co/datasets/yentinglin/aime_2025, 2025. Hugging Face dataset.

Mingjia Yin, Chuhan Wu, Yufei Wang, Hao Wang, Wei Guo, Yasheng Wang, Yong Liu, Ruiming Tang, Defu Lian, and Enhong Chen. Entropy law: The story behind data compression and llm performance, 2024. URL https://arxiv.org/abs/2407.06645.

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.

Yongcheng Zeng, Zexu Sun, Bokai Ji, Erxue Min, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, Haifeng Zhang, Xu Chen, and Jun Wang. Cures: From gradient analysis to efficient curriculum learning for reasoning llms. *arXiv preprint arXiv:2510.01037*, 2025.

Ruiqi Zhang, Daman Arora, Song Mei, and Andrea Zanette. Speed-rl: Faster training of reasoning models via online curriculum learning. *arXiv preprint arXiv:2506.09016*, 2025a.

Xiaojiang Zhang, Jinghui Wang, Zifei Cheng, Wenhao Zhuang, Zheng Lin, Minglei Zhang, Shaojie Wang, Yinghan Cui, Chao Wang, Junyi Peng, Shimiao Jiang, Shiqi Kuang, Shouyu Yin, Chaohang Wen, Haotian Zhang, Bin Chen, and Bing Yu. Srpo: A cross-domain implementation of large-scale reinforcement learning on llm, 2025b. URL https://arxiv.org/abs/2504.14286.

Yang Zhang, Yanfei Dong, and Kenji Kawaguchi. Investigating layer importance in large language models, 2024. URL https://arxiv.org/abs/2409.14381.

Yaoze Zhang, Rong Wu, Pinlong Cai, Xiaoman Wang, Guohang Yan, Song Mao, Ding Wang, and Botian Shi. Leanrag: Knowledge-graph-based generation with semantic aggregation and hierarchical retrieval, 2025c. URL https://arxiv.org/abs/2508.10391.

Yunyi Zhang, Soji Adeshina, Sheng Guan, Ashwin Ganesh, Zhen Han, Vassilis N. Ioannidis, Huzefa Rangwala, and George Karypis. Scalable prompt routing via fine-grained latent task discovery, 2026. URL https://arxiv.org/abs/2603.19415.

Xiaoke Zhao, Zhaowen Zhou, Lin Chen, Lihong Wang, Zhiyi Huang, Kaiyuan Zheng, Yanjun Zheng, Xiyang Du, Longfei Liao, Jiawei Liu, Xiang Qi, Bo Zhang, Peng Zhang, Wei Wang, and Zhe Li. Agentar-deepfinance-100k: A large-scale financial dataset via systematic chain-of-thought synthesis optimization, 2025. URL https://arxiv.org/abs/2507.12901.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, et al. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025a.

Haizhong Zheng, Yang Zhou, Brian R. Bartoldson, Bhavya Kailkhura, Fan Lai, Jiawei Zhao, and Beidi Chen. Act only when it pays: Efficient reinforcement learning for llm reasoning via selective rollouts, 2025b. URL https://arxiv.org/abs/2506.02177.

Lucia Zheng, Neel Guha, Javokhir Arifov, Sarah Zhang, Michal Skreta, Christopher D. Manning, Peter Henderson, and Daniel E. Ho. A reasoning-focused legal retrieval benchmark. In *Proceedings of the Symposium on Computer Science and Law on ZZZ*, CSLAW '25, pp. 169–193. ACM, March 2025c. doi: 10.1145/3709025.3712219. URL http://dx.doi.org/10.1145/3709025.3712219.

Hanqing Zhu, Zhenyu Zhang, Hanxian Huang, DiJia Su, Zechun Liu, Jiawei Zhao, Igor Fedorov, Hamed Pirsiavash, Zhizhou Sha, Jinwon Lee, David Z. Pan, Zhangyang Wang, Yuandong Tian, and Kai Sheng Tai. The path not taken: Rlvr provably learns off the principals, 2025. URL https://arxiv.org/abs/2511.08567.

Yuxin Zuo, Shang Qu, Yifei Li, Zhangren Chen, Xuekai Zhu, Ermo Hua, Kaiyan Zhang, Ning Ding, and Bowen Zhou. Medxpertqa: Benchmarking expert-level medical reasoning and understanding, 2025. URL https://arxiv.org/abs/2501.18362.

APPENDIX OVERVIEW

The appendix is organized into four groups.

Extended Related Work: Appendix A.
Additional discussion of related and contemporary work.

Additional Experiments: Appendices C–F.
Additional empirical studies, including structure drift investigations, external adaptive curriculum learning baselines and discussions, RL experiments on medical Q&A data, tree visualizations across different model-data settings, and a sensitivity analysis for the tree construction pipeline.

Discussions: Appendices H–K.
Broader discussion of frontier imbalance, relationships between utility-awareness and evaluation protocols, usage of external models, and limitations

Additional Experimental and Algorithmic Details: Appendices L–M.
Hyperparameters, implementation details, evaluation curves, and algorithmic details.

A Extended Related Work 21

B Coverage and Information Sharing (Diversity) 26

C Structure Drift 27

D The Standard Bandit Pattern 30

E Medical Data Experiments 34

F Tree-Construction Stability Analysis & Visualizations 36

G Tree-Construction Sensitivity Analysis 46

H Frontier Imbalance 48

I Utility-Aware Sampling and the Evaluation-Protocol Tension 51

J External Models 52

K Limitations 54

L Additional Experimental Details 56

M Additional Algorithmic Details 62

M.7 BMC-T 70

A Extended Related Work

A.1 Contemporary Work

A number of closely related papers appeared during the development and revision of this manuscript. We discuss them here as *contemporary work* because they address overlapping questions in adaptive sampling, curriculum learning, empirical Bayes estimation, task discovery, and structured exploration for LLM training. Taken together, these works reflect a broader shift from static data curation toward representation-, optimization-, and feedback-aware selection mechanisms that account for the model's evolving state or internal structure. They also highlight several recurring axes of curriculum design: *productivity*, or whether a sample provides useful learning signal; *diversity*, or whether selected batches cover non-redundant regions of the data; and *utility*, or whether the selected data supports a target evaluation distribution. BMC is complementary to these efforts: it treats curriculum selection as a structured bandit problem over a latent task manifold derived from the policy model's own representations.

Optimizer-induced Projected Utility Selection (OPUS) (Wang et al., 2026) is a dynamic data-selection method for LLM pre-training. It defines utility as expected one-step improvement on a proxy distribution, measured in the geometry induced by the optimizer, and scores candidates by the alignment between their optimizer-induced updates and a benchmark-relevant proxy direction. OPUS also includes a redundancy penalty and Boltzmann sampling to preserve diversity during batch construction. This makes OPUS closely related at the level of motivation: both works argue that effective data selection requires balancing informativeness, diversity, and target utility. However, the source of geometry differs. OPUS operates in *optimizer geometry*: it asks which pre-training sequences induce parameter updates aligned with a target gradient direction. BMC operates in *latent geometry*: it asks which RL prompts provide useful learning signal while preserving coverage over the model's representation-induced task manifold. Thus, OPUS and BMC study analogous selection problems in different training regimes (pre-training versus RL post-training), and instantiate geometry, utility, and diversity through different mechanisms.

Generalizable Predictive Prompt Selection (GPS) (Qu et al., 2026) is an RLVR prompt-selection method that combines predicted prompt difficulty with batch-level diversity. GPS trains a lightweight generalizable prompt predictive model (PPM) over the shared optimization history, allowing difficulty predictions to transfer across prompts. Its acquisition objective combines intermediate-difficulty prioritization with history-anchored diversity, using intra-batch dispersion and inter-step exploration to reduce redundancy and improve coverage. GPS is therefore closely related to BMC in emphasizing that adaptive curricula require more than prioritizing difficulty alone. The main difference is the source and role of structure: GPS uses an external embedding representation and a learned PPM/critic-style predictor to guide max-sum diversity over prompt distances, whereas BMC induces a Latent Task Tree from the policy model's own hidden representations and performs structured Thompson Sampling over this representation-derived task geometry. BMC-T further separates *utility* from productivity and diversity, allowing the curriculum to bias sampling toward a chosen target distribution.

GradAlign (Yang et al., 2026) is a contemporary RL data-selection method that uses gradient alignment to select training problems likely to improve downstream performance. It computes a target direction from the average policy gradient on a small trusted validation set and ranks candidate problems by cosine similarity between their policy gradients and this validation gradient. In this sense, GradAlign addresses the *utility* dimension of curriculum design: it distinguishes prompts that merely provide training signal from prompts whose updates are aligned with a target task. The main difference is the geometry used for selection. GradAlign operates in policy-gradient geometry, while BMC operates in latent geometry. BMC uses the policy model's own hidden representations to induce a task manifold and performs structured sampling over that space, allowing productivity, diversity, and utility to be treated as separate axes. Thus, GradAlign provides a complementary validation-alignment mechanism, but does not explicitly control *diversity* over high-utility problem types.

Empirical Bayes Policy Optimization (EBPO) (Han et al., 2026) is a contemporary method that uses empirical Bayes shrinkage to stabilize GRPO-style advantage estimation. It replaces the local group mean baseline with a shrinkage estimator that blends local reward statistics with global policy statistics estimated online, reducing variance and providing non-vanishing signals in saturated failure regimes. This is complementary to BMC: EBPO applies empirical Bayes within the RL objective

to regularize advantage baselines, whereas BMC uses Bayesian belief updates and empirical Bayes aggregation over a latent task hierarchy for curriculum scheduling. EBPO also studies clustered sampling, using externally annotated topic clusters or difficulty-based ordering to improve prior estimation. In contrast, BMC induces a Latent Task Tree from the policy model's own prompt representations and uses this hierarchy as the scheduler's action structure, separating problem *type* from problem *difficulty*. This separation is important because prompts with similar difficulty need not represent the same problem type, and prompts of the same type may span a wide range of difficulty.

DEPO (Tang et al., 2025) is a data-efficient RLVR pipeline that combines offline subset construction with online rollout pruning. Its offline stage constructs a representation-based sample graph to select diverse and influential data, then applies difficulty-aware sampling to retain appropriately challenging examples. During training, DEPO filters samples using a sample-level explorability metric and replays under-explored samples to reduce rollout cost while maintaining performance. DEPO is related to BMC in recognizing that sample relationships, diversity, and difficulty matter for efficient RLVR training. However, DEPO uses graph structure primarily for offline data curation, whereas BMC uses a policy-induced Latent Task Tree as the action structure for online curriculum scheduling. These perspectives are complementary: Latent Task Trees could also be used for dataset minimization, while BMC focuses on how latent structure can guide adaptive sampling decisions during RL training.

SAERL (Jing et al., 2026) is a recent model-internal data engineering method for RL post-training. It uses sparse autoencoder activations to estimate data diversity, difficulty, and quality, and applies these signals to offline data filtering, batch construction, and curriculum ordering. SAERL is related to BMC in emphasizing that model-internal structure can guide post-training data selection. However, the role of this structure differs: SAERL uses SAE features as an additional representation layer for preprocessing and data engineering, whereas BMC induces a Latent Task Tree directly from the policy model's hidden representations and uses it as the action structure for online curriculum scheduling. BMC updates tree-structured beliefs from observed learning signal during training, while also separating productivity, diversity, and target utility as distinct axes of curriculum design. Thus, SAERL and BMC explore complementary uses of model-internal structure: SAE-based feature analysis for data engineering, and lightweight policy-induced latent geometry for adaptive RL sampling.

FineRouter (Zhang et al., 2026) is a prompt routing method that automatically discovers fine-grained latent task types. It builds a prompt graph from LLM-generated task descriptions, external text embeddings, and model-preference rankings, then clusters this graph to train a task-aware router with specialized prediction heads. FineRouter is related to BMC in arguing that manually defined task taxonomies are too coarse to capture fine-grained model behavior. However, it addresses inference-time model routing rather than RL curriculum scheduling, and its task discovery relies on external task descriptions, embedding models, reward-model scores, and model-preference patterns. BMC instead induces a Latent Task Tree from the policy model's own hidden representations and uses the resulting hierarchy as the scheduler's action structure, enabling belief sharing, diversity-aware sampling, and multiple levels of task abstraction.

Geometries of adaptive selection. More broadly, OPUS, GradAlign, and BMC highlight that adaptive selection can be defined over different geometries. OPUS uses optimizer-induced update geometry, GradAlign uses policy-gradient geometry, and BMC uses latent geometry induced from the model's own representations. Each perspective captures a different signal about the training process, and understanding their tradeoffs or combinations remains an interesting open direction.

A.2 Latent Geometry of LLMs

A.2.1 Layer Heterogeneity

A substantial body of work demonstrates that transformer layers are functionally and geometrically heterogeneous. Analyses of hidden-state geometry show that representations evolve non-uniformly across depth, often exhibiting changes in intrinsic dimensionality and neighborhood structure that peak in intermediate layers (Valeriani et al., 2023). Layer-specific probing studies indicate that lexical semantics and structured knowledge are often most accessible in lower-to-intermediate layers, whereas later layers increasingly specialize toward next-token prediction and output reconstruction (Liu et al., 2024; Zhang et al., 2024; Lei & Cooper, 2025). As a result, further studies report that

intermediate layer representations frequently outperform final-layer embeddings on downstream tasks, suggesting that semantically rich features are concentrated prior to the final, generation-focused layers (Skean et al., 2024; 2025). Complementary perspectives connect these phenomena to compression dynamics, arguing that intermediate layers behave like information bottlenecks where salient structure is compactly organized (Yin et al., 2024).

Motivated by this literature, we construct Latent Task Trees using intermediate-layer embeddings rather than final-layer representations. Our choice reflects prior evidence that these layers capture semantically meaningful structure while remaining less entangled with token-generation objectives.

A.2.2 Manifold Representations & Regularization

Beyond layer heterogeneity, another ever-growing body of literature studies the geometric structure of representations in large language models. Several works provide empirical evidence that LLM embeddings exhibit manifold-like organization, including low intrinsic dimensionality, locally linear structure, and coherent semantic clustering (Modell et al., 2025; Doimo et al., 2024; Saglam et al., 2026; Lee et al., 2025). These studies suggest that semantic relationships are not uniformly distributed in representation space but instead occupy structured, lower-dimensional regions. Most directly related to our hierarchical construction, Park et al. (2025) show that categorical concepts can be represented as polytopes and that semantic hierarchies are reflected geometrically through orthogonality between representation subspaces. Recent work further characterizes activation geometry using local decomposition techniques, arguing that model representations are better understood as structured regions rather than isolated directions in feature space (Shafran et al., 2026).

At the same time, the manifold hypothesis has been critically examined. One concern is methodological: UMAP can be over-interpreted, since their low-dimensional projections may preserve local neighborhoods while distorting global distances, densities, and inter-cluster relationships (Jeon et al., 2025). We therefore use UMAP only as an intermediate neighborhood-preserving transformation for clustering, not as visual evidence of global geometry. A second concern is structural: Robinson et al. (2025) show that token input embeddings can contain singularities and non-constant local dimension, violating the requirements of a smooth global manifold. We view this as a critique of a strong *global* manifold assumption, whereas BMC adopts a weaker operational view: prompt representations need only exhibit enough *local* geometric regularity to support neighborhood-based partitioning. This perspective is consistent with subsequent work suggesting that LLM representations may be better described as stratified or piecewise manifolds rather than globally smooth ones (Li & Sarwate, 2025b).

Aside from representation analysis, several methods attempt to explicitly regularize or align manifold structure during training. These include mixture-of-experts routing based on localized manifold structure (Li et al., 2025), and architectural constraints designed to preserve geometric coherence across layers (Xie et al., 2025).

Taken together, these works suggest that while strict manifold assumptions may not hold globally, latent representations in large language models exhibit structured geometric organization that can be approximated and exploited. Our Latent Task Tree adopts this empirical viewpoint: rather than assuming a smooth manifold, we recursively approximate locally coherent regions in intermediate-layer latent space and leverage this structure for curriculum learning.

A.2.3 Mechanistic Interpretability & Sparse Autoencoders

In parallel to geometric analyses of representation space, mechanistic interpretability studies often use sparse representation learning to expose latent features in neural networks. This perspective is rooted in classical sparse coding, where sparsity constraints were shown to produce localized and interpretable visual features (Olshausen & Field, 1996; 1997). Recent work applies related ideas to language models through sparse autoencoders (SAEs), which learn sparse decompositions of LLM activations and can reveal interpretable latent factors (Cunningham et al., 2023). Related approaches use sparsity-inducing probes, dictionary learning, and factor-analysis-style methods to identify disentangled subspaces associated with lexical, syntactic, semantic, or world-knowledge features (Kantamneni et al., 2025; Kim et al., 2026). These methods support human interpretation and modular intervention, including causal steering (Arad et al., 2025; He et al., 2025).

Our objective is different. We do not aim to recover human-aligned latent factors or decompose activations into individually interpretable features. Instead, we use raw intermediate representations to induce a task-level geometry over prompts, which then serves as a structure for adaptive curriculum learning. Although the resulting Latent Task Trees are often human-interpretable, this interpretability is incidental rather than required: the central goal is to allocate training according to the model's implicit organization of the problem space. Further discussion of SAEs in relation to our approach is provided in Section J.3.

A.3 Hierarchical Retrieval and Structured Aggregation

A parallel line of work explores hierarchical organization for long-document summarization. Methods such as HIBRIDS (Cao & Wang, 2022) inject hierarchical document-structure biases into attention score calculation, improving the modeling of long-range document structure. RAPTOR (Sarthi et al., 2024) proposes recursive abstractive summarization to build tree-structured retrieval indices, enabling coarse-to-fine information access. LeanRAG (Zhang et al., 2025c) combines hierarchical retrieval with knowledge graph aggregation to improve structured reasoning over long contexts. Automatic Data Curation for Self-Supervised Learning (Vo et al., 2024) applies recursive clustering to organize data prior to generation, which later inspired HERCULES (Petnehazi & Aradi, 2025), a hierarchical embedding-based recursive clustering framework for efficient summarization and retrieval.

These works leverage hierarchical structure over embedding spaces to improve inference-time processing. In contrast, our method constructs hierarchical structure over prompt representations to guide adaptive sampling during reinforcement learning. Rather than organizing knowledge for retrieval over static corpora, we use latent geometry to allocate training effort during policy optimization.

A.4 Curriculum Learning & Active Learning

Outside of using RL to train LLMs, curriculum learning has long been studied as a strategy for improving optimization and generalization in neural networks. The foundational work of Bengio et al. (2009) proposed presenting training examples in a meaningful order, typically progressing from easy to difficult, to stabilize and accelerate learning. Subsequent work on self-paced learning (Kumar et al., 2010) formalized this intuition by introducing adaptive sample weighting based on model-dependent difficulty, allowing the curriculum to evolve as training progresses. Extensions such as teacher-student curriculum learning and automated curriculum selection (Matiisen et al., 2017; Graves et al., 2017) further introduced adaptive mechanisms in which a policy selects training examples based on estimated learning progress.

In reinforcement learning, related ideas appear in unsupervised environment design (UED), where tasks or environments are adaptively generated to maximize learning signal (Dennis et al., 2021). Approaches such as CLUTR (Azad et al., 2023) further explore organizing tasks within a learned latent representation space, treating tasks as residing on a structured manifold. More generally, Prioritized Experience Replay (Schaul et al., 2016) allocates updates based on estimated informativeness, sampling transitions with high learning signal more frequently.

Related notions of informative batch construction also appear in active learning. Bayesian Active Learning by Disagreement (BALD) selects examples that maximize the mutual information between predictions and model parameters, prioritizing points expected to provide high epistemic information (Houlsby et al., 2011). However, greedily selecting the highest-scoring examples can produce redundant batches, since each point is evaluated independently. BatchBALD addresses this by maximizing joint mutual information over a batch, penalizing redundant selections and encouraging complementary examples (Kirsch et al., 2019). This is closely aligned with our motivation: curriculum design should not only identify individually productive prompts, but should also account for diversity and structure when forming training batches.

A.5 Structured & Causal Multi-Arm Bandits

The general notion that bandit arms can share correlations (Boda & A, 2019; Gupta et al., 2021), reside on a shared structure (Ginebra & Clayton, 1995; Pandey et al., 2007; Mersereau et al., 2009),

or support learnable causal interventions (Lattimore et al., 2016) served as a main catalyst for this work.

Additionally, several variants of Thompson Sampling are especially relevant to our setting. Hierarchical Thompson Sampling (HierTS) was introduced for hierarchical Bayesian bandits, where related bandit tasks share information through a common hyper-posterior (Hong et al., 2022b). This is conceptually aligned with BMC, where prompts can be viewed as related tasks whose learning-signal estimates should not be treated as independent. Subsequent work extends this idea to tree-structured action spaces, where actions correspond to leaves and internal latent variables induce correlations among action rewards (Hong et al., 2022a). More recently, empirical Bayesian multi-bandit Thompson Sampling (ebmTS) estimates shared prior covariance across bandit instances from data and uses the resulting empirical posterior for exploration (Jiang & Zhu, 2025). BMC differs from these works by inducing a latent task tree from the model's own prompt representations, rather than relying on a given task relation, action taxonomy, or covariance structure. The resulting hierarchy supports curriculum scheduling in RL by combining structured exploration, diversity-aware sampling, and belief sharing over prompts.

B Coverage and Information Sharing (Diversity)

As summarized in the main text, similar productivity does not imply that sampling strategies allocate training effort to the same regions of the task manifold. We therefore analyze Diversity along two complementary dimensions: *coverage*, measuring exposure to underrepresented problem types, and *information sharing*, measuring whether related problem types exhibit shared structure in their learning signal. We operationalize these using *rarity-weighted exposure* and *structure gain*, respectively. Structure gain measures how much variation in learning signal is explained by the Latent Task Tree relative to matched random partitions.

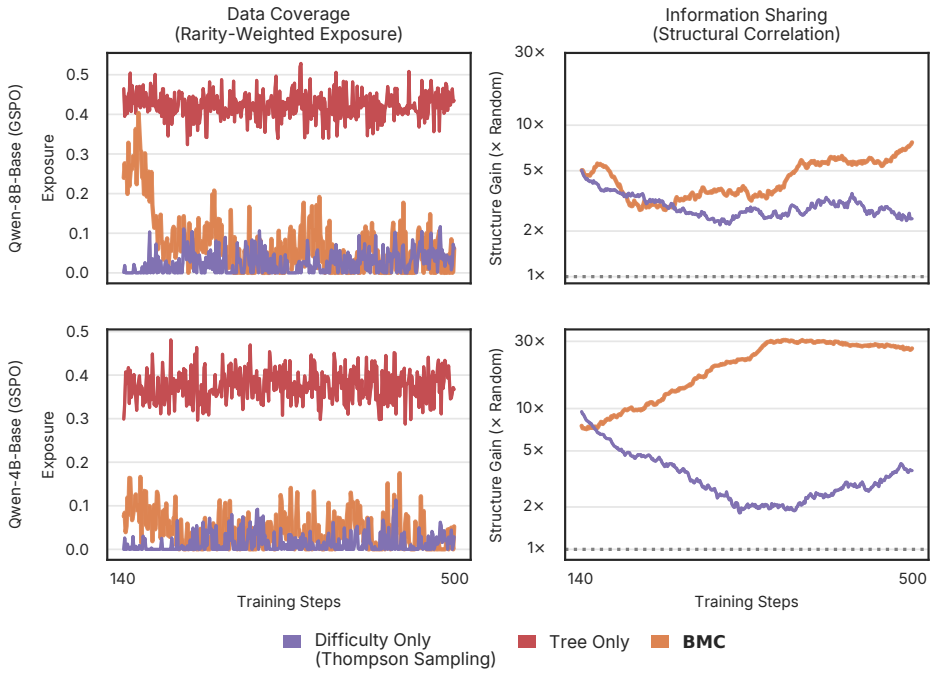

Figure 7. **Coverage and information sharing across sampling strategies. Left**: *Rarity-weighted exposure*, measuring coverage of underrepresented regions of the task manifold. BMC balances the diversity-focused behavior of the Tree Only ablation with the productivity-focused behavior of Difficulty Only. **Right**: *Structure gain*, measuring how much variation in learning signal is explained by the Latent Task Tree relative to random partitions. Both BMC and Difficulty Only exhibit substantial structure gain, indicating that learning signal already possesses latent structure even when the tree is not explicitly used for sampling.

As shown in Figure 7, BMC interpolates between the Tree Only ablation, which emphasizes rare problem types, and Difficulty Only, which emphasizes productivity but has comparatively limited coverage. This shows that BMC changes not only how much learning signal is obtained, but also which regions of the task manifold receive training effort.

For information sharing, we approximate shared learning-relevant difficulty using learned prompt beliefs μ_i, which estimate expected learning signal, as a proxy for shared difficulty. Structure gain is defined as the ratio between the R^2 explained by penultimate Latent Task Tree clusters and the R^2 explained by random partitions with matched cluster sizes. Although absolute R^2 values are modest, this ratio measures whether the tree explains more variation in learning signal than expected by chance.

Notably, this structure is present even for Difficulty Only, which does not use the tree to guide selection, indicating that learning signal *already* abides by a latent organization. When the tree is actively used, as in BMC, sampling and belief updates become increasingly more aligned. Intuitively, we say that BMC does not *create* structure or correlations in the learning signal; it *exploits* structure that *already exists*. Exact formulas and additional discussion are provided in Appendix M.7.

C STRUCTURE DRIFT

BMC constructs the Latent Task Tree at the start of training and keeps its topology fixed throughout RL. The beliefs associated with each node are updated over time, but the tree structure itself remains static. This accounts for changes in the model's observed reward landscape, but not for changes in the model's latent representation of prompts during training.

We keep the tree fixed in the main experiments for two reasons. First, RLVR typically induces relatively small parameter updates compared with large-scale supervised fine-tuning (Mukherjee et al., 2025; Zhu et al., 2025). Second, the tree is constructed from prompt representations, and prompt tokens are masked from the policy-gradient objective. As a result, the representations used to build the tree may be less directly perturbed by RL updates than the response tokens being optimized. Nonetheless, it is unreasonable to assume that the model's perception of prompts remains completely fixed. We therefore investigate structural drift in two ways: (1) by comparing Latent Task Trees before and after RL, and (2) by running BMC with periodic tree reconstruction.

C.1 TREE STRUCTURE BEFORE AND AFTER RL

We first compare Latent Task Trees constructed from pre-RL and post-RL model checkpoints. This analysis is not intended to isolate the causal effect of a particular RL algorithm, since the compared checkpoints may differ in training data, optimization details, and model lineage. Instead, the goal is diagnostic: to test whether the latent task geometry used by BMC can change after RL-style post-training, and whether those changes follow a consistent pattern. To reduce variation from the tree-construction procedure itself, we fix all random seeds and construct the compared trees on the same machine. Under this setting, tree construction is reproducible across repeated runs, though we do not assume exact reproducibility across different hardware or software environments.

Table 1. **Latent Task Tree structure before and after RL for Qwen3-8B.** We compare the tree induced by `Qwen3-8B-Base` before RL training and the corresponding tree after RL, trained using BMC.

Metric	Pre-RL	Post-RL	Change
Total nodes	50	48	−2
Internal nodes	7	7	0
Terminal nodes	43	41	−2
Maximum width	16	20	+4
Maximum depth	4	4	0
Maximum branching factor	13	17	+4
Mean internal branching factor	7.00	6.71	−0.29

Table 1 summarizes the structural changes in the Latent Task Tree before and after RL training for `Qwen3-8B`. The post-RL tree does not become deeper, but it becomes wider at its largest level and exhibits a larger maximum branching factor. This suggests that RL can change the latent organization of task regions by altering the breadth of certain partitions, even when the overall hierarchy depth remains stable.

We repeat the same analysis for a pre-RL and post-RL model pair that did not use our sampling methods. Specifically, we compare the trees generated for the `DeepCoder-Preview` dataset using `DeepSeek-R1-Distilled-Qwen-14B` (Guo et al., 2025) as the pre-RL model and `DeepCoder-14B-Preview` (Luo et al., 2025) as the post-RL model that was trained on the dataset. Table 2 summarizes the structural differences between the pre-RL and post-RL coding trees. In contrast to the `Qwen3-8B` comparison, the post-RL coding tree does not become wider; instead, it is slightly smaller, with fewer terminal nodes and a lower maximum branching factor. Both trees remain shallow, suggesting a modest contraction of the induced latent task partition rather than an increase in hierarchical complexity.

Finally, we compare the tree generated for the `GURU-92k` dataset using `Qwen2.5-7B` as the pre-RL model and `Guru-7B` as the corresponding post-RL model that was trained on the dataset (Cheng et al., 2026). Table 3 shows a third pattern. Unlike the `Qwen3-8B-Base` comparison, where the

Table 2. **Latent Task Tree structure before and after RL for coding models.** We compare the tree induced by `DeepSeek-R1-Distilled-Qwen-14B` and the corresponding post-RL model, `DeepCoder-14B-Preview`.

Metric	DeepSeek-R1-Distilled (Pre-RL)	DeepCoder (Post-RL)	Change
Total nodes	21	19	-2
Internal nodes	2	2	0
Terminal nodes	19	17	-2
Maximum width	17	15	-2
Maximum depth	2	2	0
Maximum branching factor	17	15	-2
Mean internal branching factor	10.00	9.00	-1.00

Table 3. **Latent Task Tree for before and after RL for Guru-7B.** We compare the tree induced by `Qwen2.5-7B` before RL and the corresponding post-RL model, `Guru-7B`.

Metric	Qwen2.5-7B (Pre-RL)	Guru-7B (Post-RL)	Change
Total nodes	50	56	$+6$
Internal nodes	8	12	$+4$
Terminal nodes	42	44	$+2$
Maximum width	18	13	-5
Maximum depth	4	7	$+3$
Maximum branching factor	14	13	-1
Mean internal branching factor	6.13	4.58	-1.54

largest partition becomes wider, and the `DeepCoder` comparison, where the tree slightly contracts, the `Guru-7B` tree becomes deeper and contains more internal nodes. At the same time, its maximum width and maximum branching factor decrease. This suggests a shift from a broader partition toward a more hierarchical decomposition of latent task regions.

Taken together, these comparisons suggest that RL can alter latent task geometry, but not in a single universal direction. Across the three settings, post-RL trees differ in whether they become wider, smaller, or more hierarchical. We therefore treat these measurements as a diagnostic of structure drift, rather than as evidence for a fixed pattern of representational change. Whether these structural changes connect to broader observations about how RLVR alters response diversity and representation geometry is an interesting direction for future work.

For BMC, the practical question is not only whether the latent tree drifts, but whether rebuilding it during training improves curriculum decisions. We therefore treat periodic reconstruction as an empirical design choice rather than an obvious requirement.

C.2 Periodic Tree Construction Experiment

To test whether reconstruction is useful in practice, we compare standard BMC, which keeps the Latent Task Tree fixed throughout training, against a variant that periodically reconstructs the tree during RL. If the model's latent representation of prompts drifts in a way that materially affects sampling decisions, then updating the tree should provide a better approximation to the current task geometry and improve downstream performance. Conversely, if the static tree remains sufficiently aligned with the relevant curriculum structure, periodic reconstruction should provide little benefit.

Figure 8 shows that periodic reconstruction produces only modest differences across evaluations. Reconstruction slightly improves several concentrated math benchmarks, including AIME-style benchmarks, AMC, and the Chinese math aggregate, while slightly reducing performance on MATH500 and GPQA-Diamond. One possible explanation is that reconstruction changes which latent regions are emphasized as the model's representation evolves, benefiting some evaluation distributions while

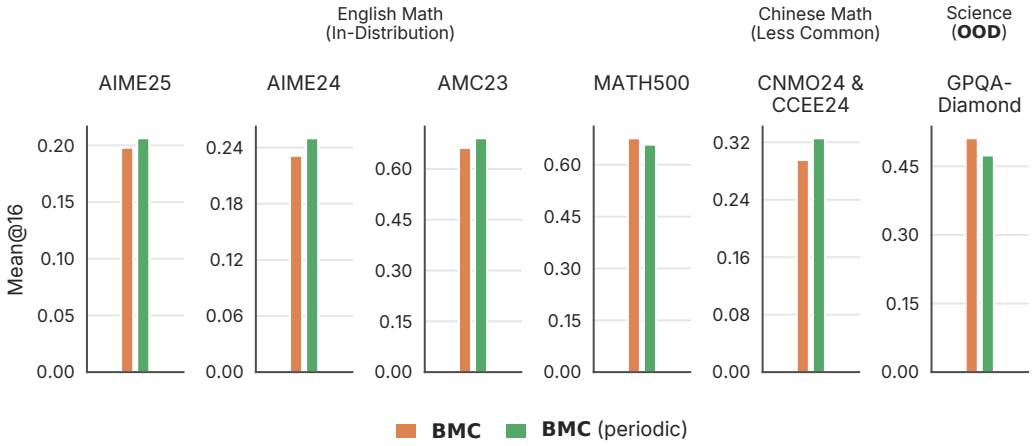

Figure 8. **Periodic tree reconstruction.** We compare standard BMC, which keeps the Latent Task Tree fixed throughout training, against a variant that reconstructs the tree every 100 steps after round-robin initialization.

reducing coverage or transfer to others. We treat this interpretation as speculative, but the small magnitude of the changes suggests that the static tree remains a reasonable approximation over the training horizon. We therefore keep the tree fixed in the main experiments for simplicity and stability.

D THE STANDARD BANDIT PATTERN

The purpose of this section is to clarify why we treat the **Difficulty Only** ablation as a representative adaptive curriculum learning baseline, rather than attempting to exhaustively compare against every recent adaptive prompt-selection method. We view our work as part of a growing line of adaptive curriculum learning methods for RL, many of which differ in motivation and implementation but share a common productivity-oriented abstraction: *estimate which individual prompts are likely to be informative and sample those prompts more often*. Our goal is not to claim that all instantiations are identical, but to identify the common mechanism they share and evaluate a strong implementation of that mechanism under the same training stack.

This abstraction arises from a common bottleneck in group-relative policy optimization: prompts whose sampled rollouts receive identical rewards provide little or no advantage signal. Dynamic sampling mitigates this issue by resampling until informative groups are found, but can incur substantial wall-clock overhead. Adaptive curriculum learning methods typically try to increase the *effective ratio* (the fraction of sampled prompts that contribute to the gradient) *without* the increase in wall-clock time by predicting useful prompts before sampling.

Across this literature, we refer to the recurring abstraction as the *standard bandit pattern*. We use the term "bandit pattern" descriptively rather than literally: a method need not implement a formal UCB or Thompson-sampling algorithm to follow this pattern, as long as it repeatedly assigns scalar values to prompt-level arms and adapts sampling according to those values.

In this pattern, each prompt is treated as an arm whose value estimates how useful the prompt is expected to be for learning. In this work, we refer to this value as the *learning signal*. As discussed in Appendix M.1.1, several common choices are closely related in binary-reward RLVR: reward variance is maximized near a 50% success rate, while expected absolute advantage and related information measures also prioritize prompts with nontrivial rollout-level contrast. Methods may differ in their statistical representation and sampling rule, for example by using Beta or Gaussian distributions, UCB or Thompson Sampling, and value estimates derived from running statistics or critic-style predictors. A scheduler then repeatedly samples prompts according to these values and updates the corresponding estimates from observed rewards.

More concretely, the pattern can be summarized as:

1. assign each prompt x_i an estimated learning value v_i;
2. initialize $\{v_i\}_{i=1}^{N}$ uniformly or from a prior model;
3. for each training step t:
   (a) sample a batch of individual prompts according to their estimated values, often without replacement;
   (b) generate rollouts and observe rewards for the sampled prompts;
   (c) update the values of the sampled prompts using the observed rollout rewards;
4. repeat as the policy evolves.

The standard bandit pattern is therefore a strong mechanism for improving sampling productivity, but it does not explicitly control several dimensions that can be important when training LLMs on broad, heterogeneous mixtures. In particular, it does not directly control *diversity* over problem types or *utility* relative to a target distribution. If prompts are prioritized only by expected productivity (usually defined in terms of difficulty), then coverage over problem types becomes a byproduct of the productivity objective rather than a controlled property of the scheduler. As a result, a productivity-only sampler can repeatedly allocate effort to a narrow set of high-signal regions even when other, less-represented regions also provide useful learning signal. This becomes especially subtle when the policy-dependent frontier is imbalanced differently from the raw dataset; see Appendix H. Similarly, prompts that are highly productive for training may not be the prompts most aligned with a particular target distribution.

This does not mean that difficulty-only methods are unimportant or solved: better predictors of problem difficulty can still improve the productivity axis. Rather, our point is that productivity remains only one axis of curriculum design, and improving it does not by itself guarantee diversity over problem types or benchmark superiority.

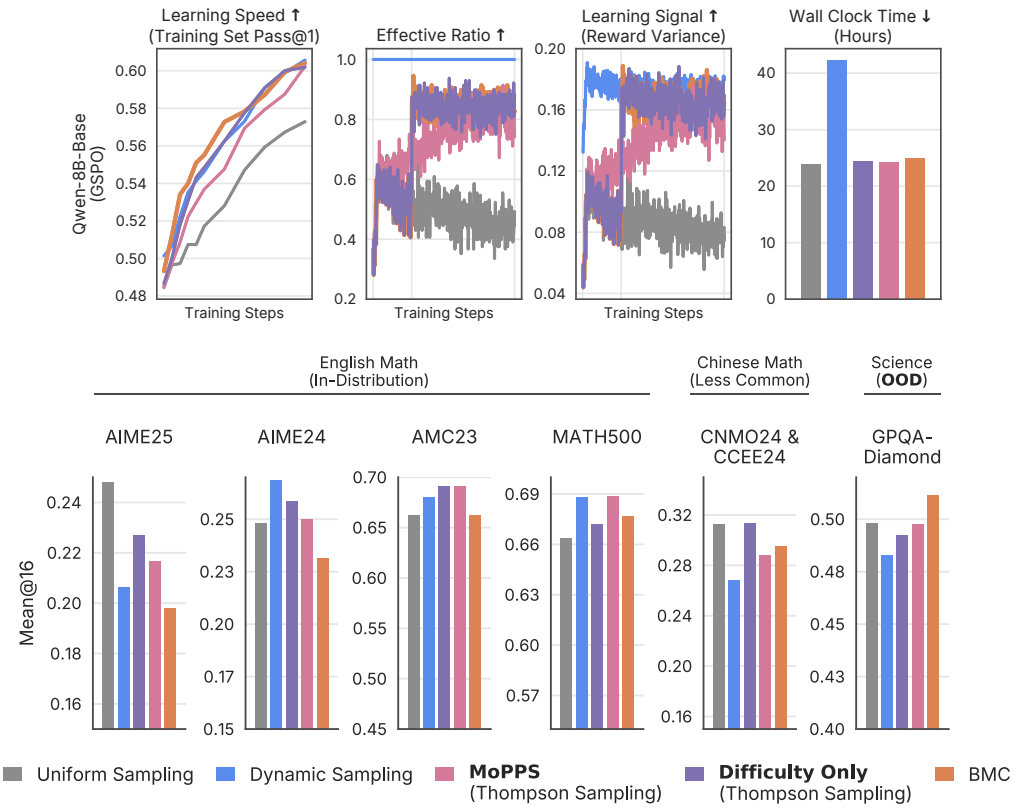

Figure 9. **Productivity and evaluation behavior under the standard bandit pattern. Top:** Training productivity metrics on `DAPO-Math-17K`. MoPPS and the Difficulty Only ablation both improve learning speed and effective ratio relative to uniform sampling while avoiding the wall-clock overhead of dynamic sampling. **Bottom:** Evaluation performance across English mathematics, Chinese mathematics, and out-of-distribution science benchmarks. MoPPS and the Difficulty Only ablation exhibit similar qualitative behavior, supporting their use as representative instantiations of the standard bandit pattern. See Figure 10 for per-step trajectories.

We design Difficulty Only to serve as a strong internal instantiation of the standard bandit pattern under our training stack. Rather than constructing an artificially weak baseline, Difficulty Only preserves the same policy optimization setup as BMC, but removes the latent-task structure and associated structure-aware sampling components. To validate that its behavior is representative of the broader pattern, we compare it against **MoPPS** (Qu et al., 2025), an external adaptive curriculum method for RLVR. Like Difficulty Only, MoPPS uses Thompson-style posterior sampling, but with a different belief model and selection rule. Specifically, MoPPS models each prompt's success rate using a Beta posterior,

$$p_i \sim \mathrm{Beta}(\alpha_i, \beta_i),$$

samples from each posterior, and selects the top-B prompts whose sampled success probabilities are closest to 0.5:

$$\mathcal{B}_t = \mathrm{TopB}_i \left(-|p_i - 0.5|\right).$$

This corresponds to prioritizing prompts near maximum reward variance under binary rewards. After sampling, the posterior parameters are updated from observed rewards. Unlike our Difficulty Only ablation, MoPPS decays posterior statistics only for sampled prompts; unsampled prompts do not become more uncertain over time. Despite these differences, MoPPS follows the same broad standard bandit pattern: it maintains prompt-level value estimates and adaptively samples prompts expected to be informative.

We evaluate MoPPS on `DAPO-Math-17K` and compare it with uniform sampling, dynamic sampling, our Difficulty Only ablation, and BMC. Since MoPPS does not use round-robin initialization by default, we preserve its original sampling behavior rather than modifying it to match our ablations.

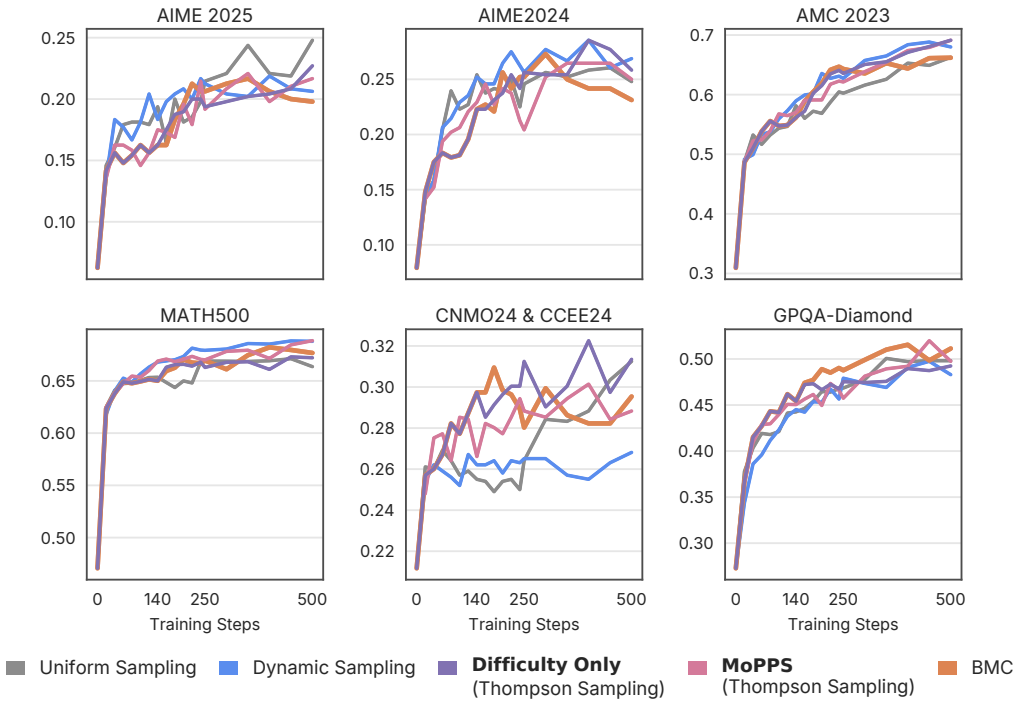

Figure 10. **Evaluation curves.** Performance across evaluations for `Qwen3-8B-base` while training on `DAPO-Math-17k` using **GSPO**. These evaluation trajectories correspond to the bar graphs visualized in Figure 9.

Figure 9 supports the role of Difficulty Only as a representative baseline for the standard bandit pattern. Although MoPPS and Difficulty Only differ in belief models, update rules, and initialization, both improve training-set learning speed and effective ratio relative to uniform sampling without incurring the wall-clock cost of dynamic sampling. Both also achieve their strongest relative performance on in-distribution English mathematics evaluations, where productivity-oriented sampling is well aligned with the evaluation distribution.

At the same time, both methods illustrate the limitations of productivity-only curricula. They perform better than dynamic sampling on the less-represented Chinese mathematics evaluations, consistent with the concern that generate-then-discard strategies can skip problem regions where the current policy receives little immediate learning signal. However, both standard-bandit methods perform lower on average than BMC on the out-of-distribution GPQA-Diamond benchmark[1]. Moreover, despite achieving higher effective ratios and absorbing more of the training data, they fall behind uniform sampling on AIME2025, suggesting that brute-force coverage can sometimes expose the model to high-utility regions that productivity-oriented sampling underweights.

We therefore view the standard bandit pattern as an important and effective baseline for adaptive curriculum learning in RLVR. It captures the dominant productivity-oriented mechanism in this line of work, and different instantiations can produce similar qualitative behavior. However, these results reinforce the central claim of this work: productivity, or "difficulty awareness," is not a complete model of curriculum design. When most of the training distribution has high utility for the target evaluations, methods that follow the standard bandit pattern may be sufficient. However, as the training mixture becomes broader and more heterogeneous, and as target evaluations emphasize

[1]MoPPS briefly exceeds BMC on one intermediate GPQA-Diamond evaluation. We emphasize the overall trajectory, rather than isolated pointwise crossings, when interpreting this result.

different regions of the task manifold, productivity-oriented curricula become increasingly vulnerable to train-test mismatch unless diversity and utility are explicitly controlled.

BMC generalizes the standard bandit pattern by changing the action space over which curriculum decisions are made: from independent prompt-level arms to structured regions of a latent task manifold. This allows the scheduler to exploit structural correlations between prompts, share evidence across related regions, and reason about sampling decisions at multiple levels of abstraction. As a result, *diversity* over latent problem types becomes an explicit part of the sampling process rather than an incidental byproduct of prompt-level exploration. BMC-T further extends this view by incorporating *utility* (evaluation relevance), using latent proximity to bias sampling toward regions expected to support a target evaluation distribution. Together, these extensions move beyond asking only how much learning signal a prompt provides, and instead ask which regions of the task manifold should receive training effort, how evidence should propagate across related tasks, and which downstream evaluations that effort is likely to support.

E MEDICAL DATA EXPERIMENTS

To further evaluate the domain generality of Latent Task Trees and BMC, we run an additional experiment on the medical question-answering dataset `AlphaMed19K` (Liu et al., 2025). We use `Qwen3-8B` with the same training hyperparameters as in the main experiments; see Appendix L.1. This experiment is intended as a domain-transfer stress test rather than a fully tuned medical-training recipe.

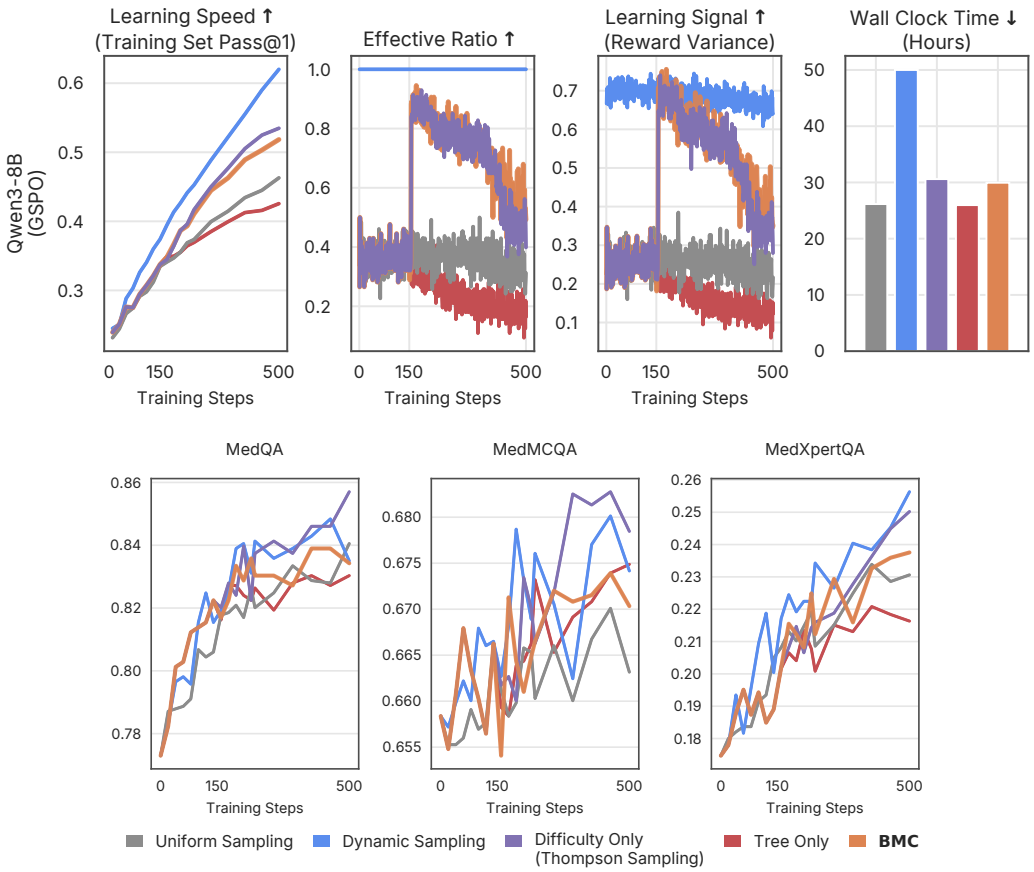

Figure 11. **Training and evaluation dynamics on AlphaMed19K. Top:** Training productivity metrics for Uniform Sampling, Dynamic Sampling, Difficulty Only, Tree Only, and BMC. Dynamic Sampling maintains the highest effective ratio and reward variance, but incurs substantially higher wall-clock time due to repeated filtering. BMC and Difficulty Only improve learning speed (training-set pass@1) relative to Uniform Sampling, even though their effective ratio and observed learning signal decline after adaptive sampling begins. **Bottom:** Medical evaluation performance on MedQA, MedMCQA, and MedXpertQA. Evaluation performance is strongest for Dynamic Sampling and Difficulty Only, while BMC's structure-aware sampling does not translate into the strongest downstream accuracy in this setting. This reinforces the tradeoffs between productivity, diversity, and utility.

Productivity. Figure 11 shows that `AlphaMed19K` exhibits different training dynamics from the main `DAPO-Math-17K` experiments. After the round-robin initialization phase, the effective ratio and observed learning signal decline over training for both BMC and Difficulty Only. Nevertheless, both methods improve training-set pass@1 relative to Uniform Sampling, indicating that adaptive prompt selection still identifies useful regions of the dataset early in training.

Dynamic Sampling maintains the highest effective ratio and reward variance by construction, since it filters candidate prompts until it forms an effective composite batch. However, this comes at

a substantial wall-clock cost. In the main mathematics experiments, Dynamic Sampling required roughly two filtering iterations on average to construct an effective batch; on `AlphaMed19K`, it required roughly three. This suggests that, for this model–dataset pair, productive prompts are sparser or become depleted more quickly, forcing generate-then-discard sampling to spend more computation searching for informative examples.

Utility. For evaluation, we partially follow the original medical question-answering setup from Liu et al. (2025) and evaluate on MedQA (Jin et al., 2020), MedMCQA (Pal et al., 2022), and MedXpertQA (Zuo et al., 2025). Due to the larger size of these evaluation sets, we report pass@1 rather than mean@16.

The evaluation results show that adaptive sampling remains effective in the medical domain and exhibits a pattern consistent with the main `DAPO-Math-17K` experiments. Dynamic Sampling and Difficulty Only are strongest on several medical evaluations, while BMC remains competitive but is not uniformly best. This mirrors the broader theme of the paper: productivity, diversity, and utility are distinct axes of curriculum design, and optimizing for one does not guarantee optimality along the others. As in the main experiments, this suggests that downstream evaluation performance may depend on how well structure-aware coverage aligns with the particular capabilities measured by the evaluation suite.

One possible explanation is that the Latent Task Tree derived from `AlphaMed19K` exhibits a more fine-grained structure than the tree derived from `DAPO-Math-17K` (see Figure 14). In such settings, structure-aware sampling may allocate effort across semantically distinct regions whose relevance to a fixed evaluation suite is uneven or difficult to infer from productivity alone. This interpretation is suggestive rather than causal, but it reinforces the motivation for target-aware variants such as BMC-T, where latent structure can be combined with an explicit utility signal.

Overall, these experiments further support the domain generality of Latent Task Tree construction and BMC, while reinforcing the central point that curriculum design should consider more than difficulty alone. On `AlphaMed19K`, BMC improves over Uniform Sampling in training productivity and approaches Dynamic Sampling without incurring the same wall-clock overhead, but Difficulty Only and Dynamic Sampling achieve stronger performance on several medical evaluations. This should not be interpreted as evidence that structure-aware sampling is domain-specific or ineffective in medical question answering. Rather, it provides another example in which productivity, diversity, and utility need not align automatically. This makes `AlphaMed19K` a useful case for target-aware variants such as BMC-T, where structure-aware coverage can be biased toward regions more directly connected to a target evaluation distribution or desired capability.

Base Model Attempt. We also attempted to run the same experiment with the base `Qwen3-8B-Base` model. However, despite testing multiple prompt templates, we found that the base model often converged to producing only the multiple-choice answer without the reasoning format required by our RL setup. The instruction-tuned model did not exhibit this issue. As a result, we use the instruction-tuned model for the reported `AlphaMed19K` experiment. This leaves open whether the observed dynamics are primarily due to the medical domain, the strength of the instruction-tuned initialization, or the interaction between the model and dataset.

F TREE-CONSTRUCTION STABILITY ANALYSIS & VISUALIZATIONS

To demonstrate the generality of Latent Task Tree construction, we provide full tree visualizations and construction statistics for the following datasets:

- **DAPO-Math-17K** (Yu et al., 2025): a multilingual math dataset used in our main experiments.
- **DeepCoder-Preview** (Luo et al., 2025): a coding dataset.
- **AlphaMed19K** (Liu et al., 2025): a medical reasoning dataset.
- **BarExamQA** (Zheng et al., 2025c): a legal reasoning dataset.
- **Agentar-DeepFinance-100k** (Zhao et al., 2025): a financial reasoning dataset.
- **GURU-92k** (Cheng et al., 2026): a multi-domain dataset spanning math, code, science, logic, simulation, and tabular reasoning.
- **IF-RLVR** (Pyatkin et al., 2025): a verifiable instruction-following dataset spanning mathematics, coding, NLP tasks, multilingual QA, writing, safety-sensitive requests, and other open-ended user requests, with prompts combining base tasks and automatically checkable output constraints.
- **Geometry3K** (Lu et al., 2021): a multimodal vision-language math dataset.

All trees were constructed using the same hyperparameters reported in Appendix L.1.

The tree structures are determined algorithmically, but the displayed node labels are interpretive annotations. To assign labels, we inspected 10 representative prompts and word clouds for each cluster, using ChatGPT to assist with summarization. These labels are necessarily coarse due to space constraints, and clusters without a clear dominant theme are denoted by "???". Each visualization is paired with example word clouds and recursion statistics, which summarize why recursive calls terminated: either because a node passed the chart test or because it failed to meet the required clustering criteria ("insufficient clusters").

Although latent task trees can be useful for interpreting datasets, node labeling is not required to use BMC. The algorithm operates directly on the unlabeled tree structure induced from the model's latent representations. The manual labels are included only to make the induced semantic organization visible to the reader, demonstrating that coherent task groupings emerge from raw LLM embeddings without requiring external embedding models or hand-designed taxonomies.

Finally, while fixing the random seed makes tree construction reproducible on a given machine, exact trees may vary across machines due to implementation- and hardware-level nondeterminism. We expect the broad semantic organization to remain similar, but the exact hierarchy may not be identical.

The tree visualizations are shown on the following pages.

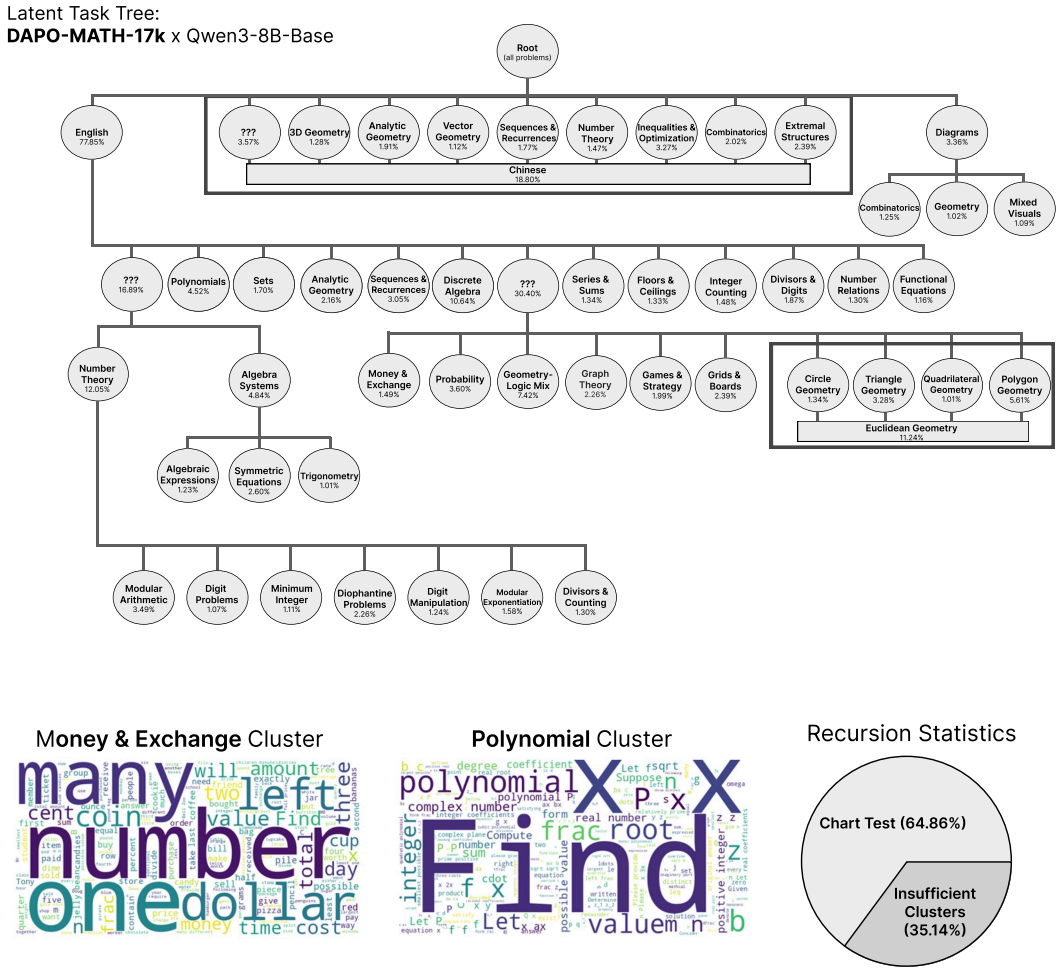

Figure 12. **Latent Task Tree for `DAPO-Math-17k` using Qwen3-8B-Base's latent space.** The dataset contains mathematical reasoning problems in both Chinese and English. Themes such as *Chinese* and *Euclidean Geometry* are not tree nodes, but descriptive annotations used to summarize the condensed view of the full tree shown in Figure 2. The induced tree contains 50 nodes. On one node with 8×H100 GPUs, loading the latent representations took 5:35 minutes and recursive tree construction took 6:29 minutes, for a total construction time of 12:04 minutes. Since the corresponding training run took 25.06 hours (*including* the tree construction time), the tree construction accounted for only 0.802% of the total training time.

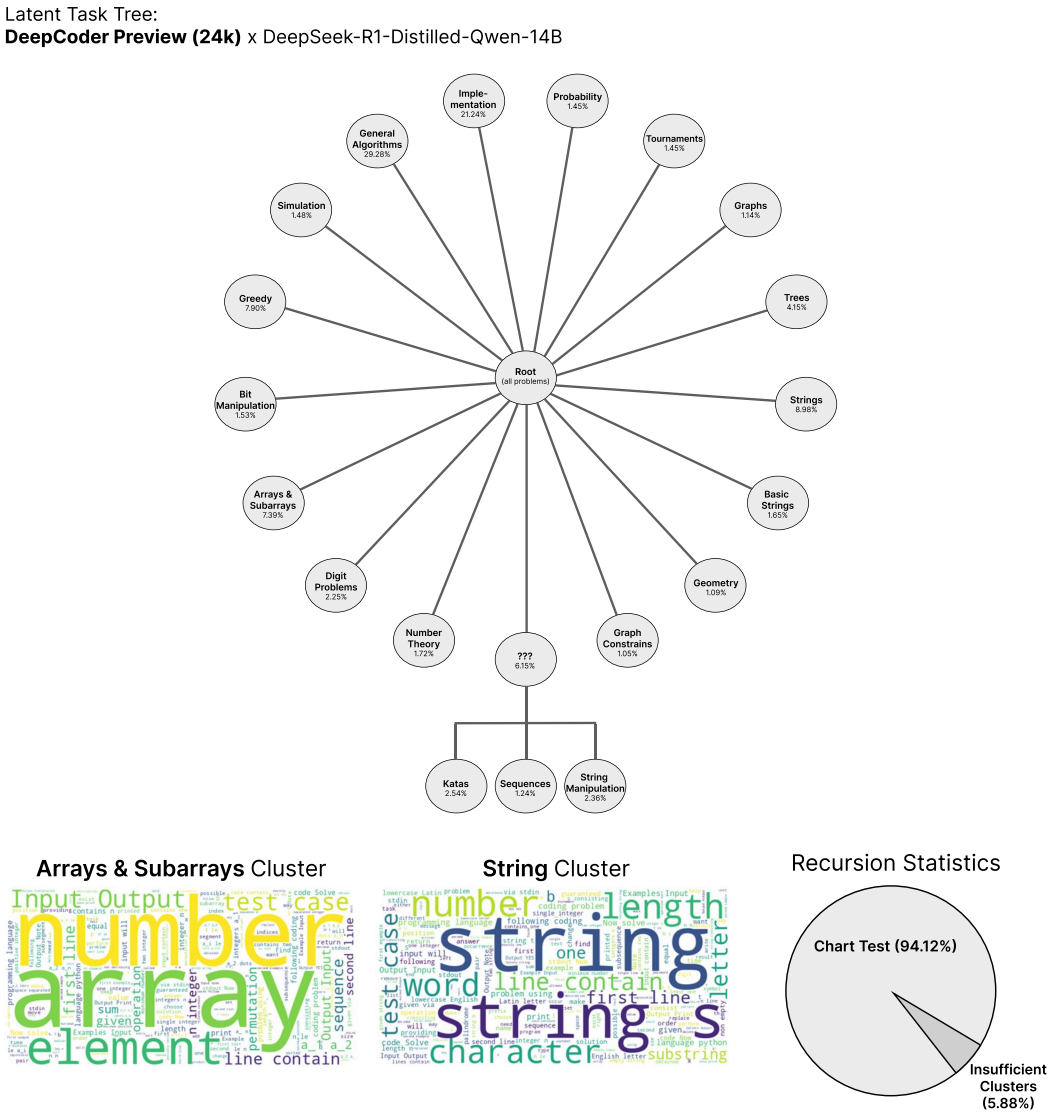

Figure 13. **Latent Task Tree for *DeepCoder-Preview-Dataset* (Luo et al., 2025) using DeepSeek-R1-Distilled-Qwen-14B's latent space.** The dataset consists of 24K coding problems. The induced tree contains 49 nodes. On one node with 8×H100 GPUs, loading the latent representations took 12:56 minutes and recursive tree construction took 7:22 minutes, for a total construction time of 20:18 minutes.

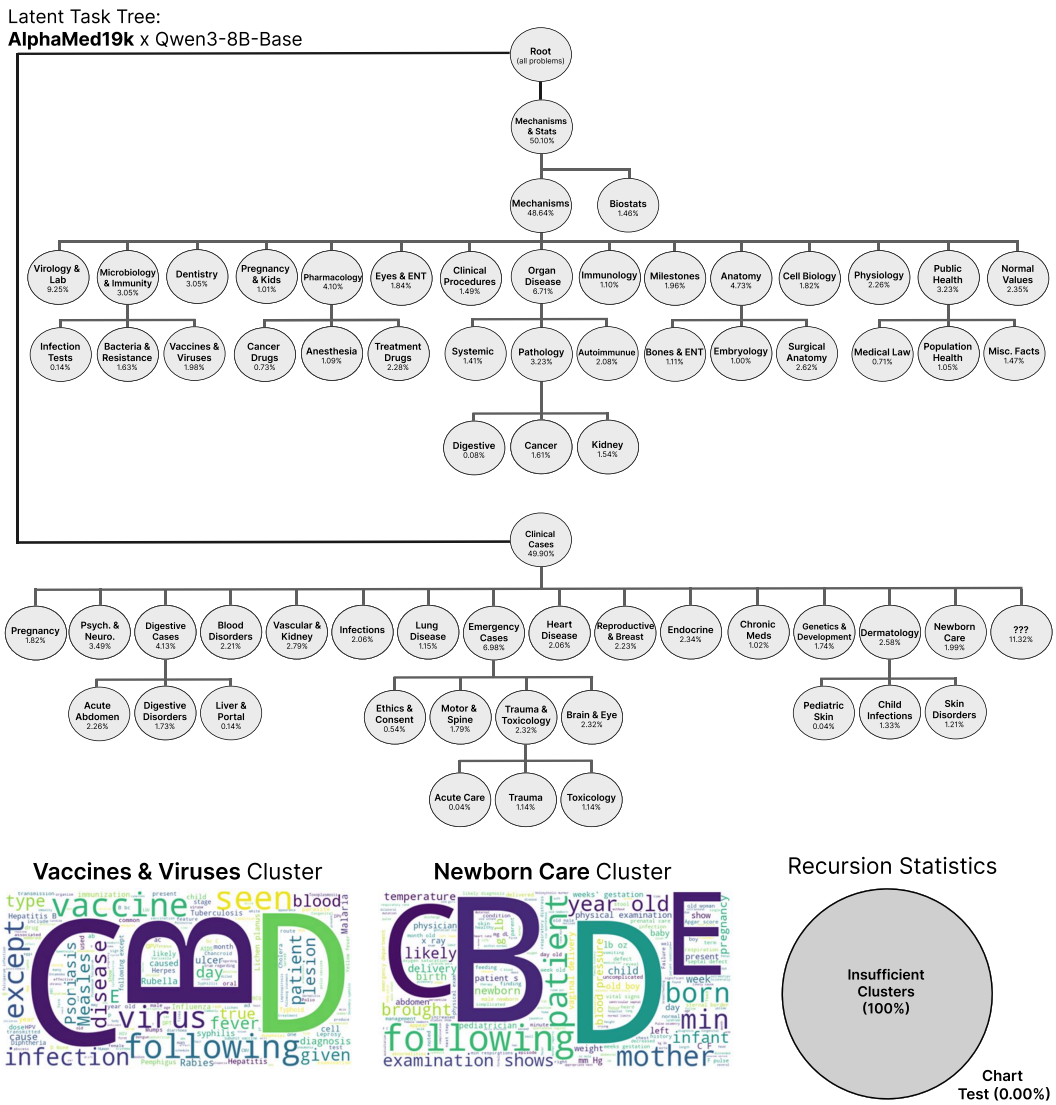

Figure 14. **Latent Task Tree for *AlphaMed19k* (Liu et al., 2025) using Qwen3-8B-Base's latent space.** The dataset contains medical Q&A multiple-choice problems across varying topics. The induced tree contains 67 nodes. On one node with 8×H100 GPUs, loading the latent representations took 10:07 minutes and recursive tree construction took 10:46 minutes, for a total construction time of 20:53 minutes. Since the corresponding training run (see Appendix E) took 30.30 hours (*including* tree construction time), tree construction accounted for only 1.15% of the total training time.

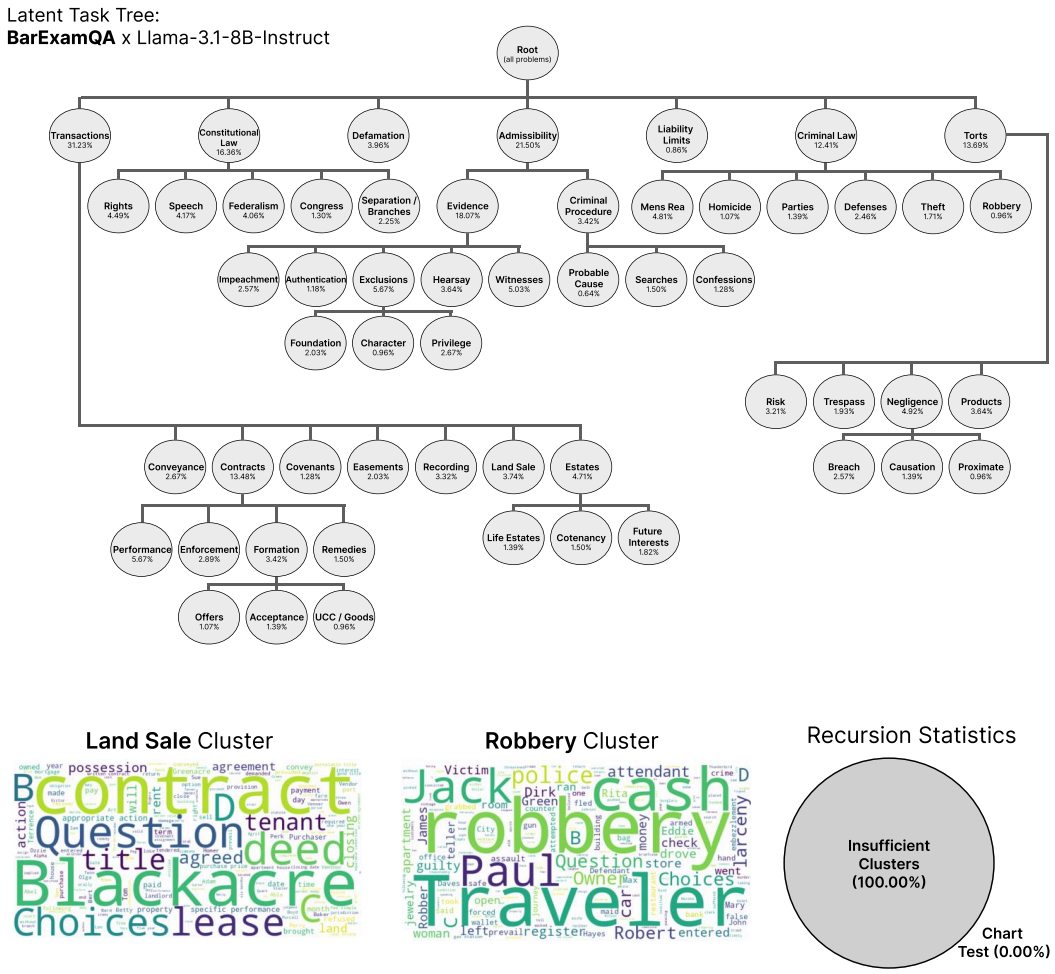

Figure 15. **Latent Task Tree for *BarExamQA* (Zheng et al., 2025c) using the latent space of Llama-3.1-8B-Instruct (Grattafiori et al., 2024).** The dataset consists of 1K multistate bar exam (MBE) questions. The induced tree contains 56 nodes. On one node with 8×H100 GPUs, loading the latent representations took 52 seconds and recursive tree construction took 51.5 seconds, for a total construction time of 1:44 minutes.

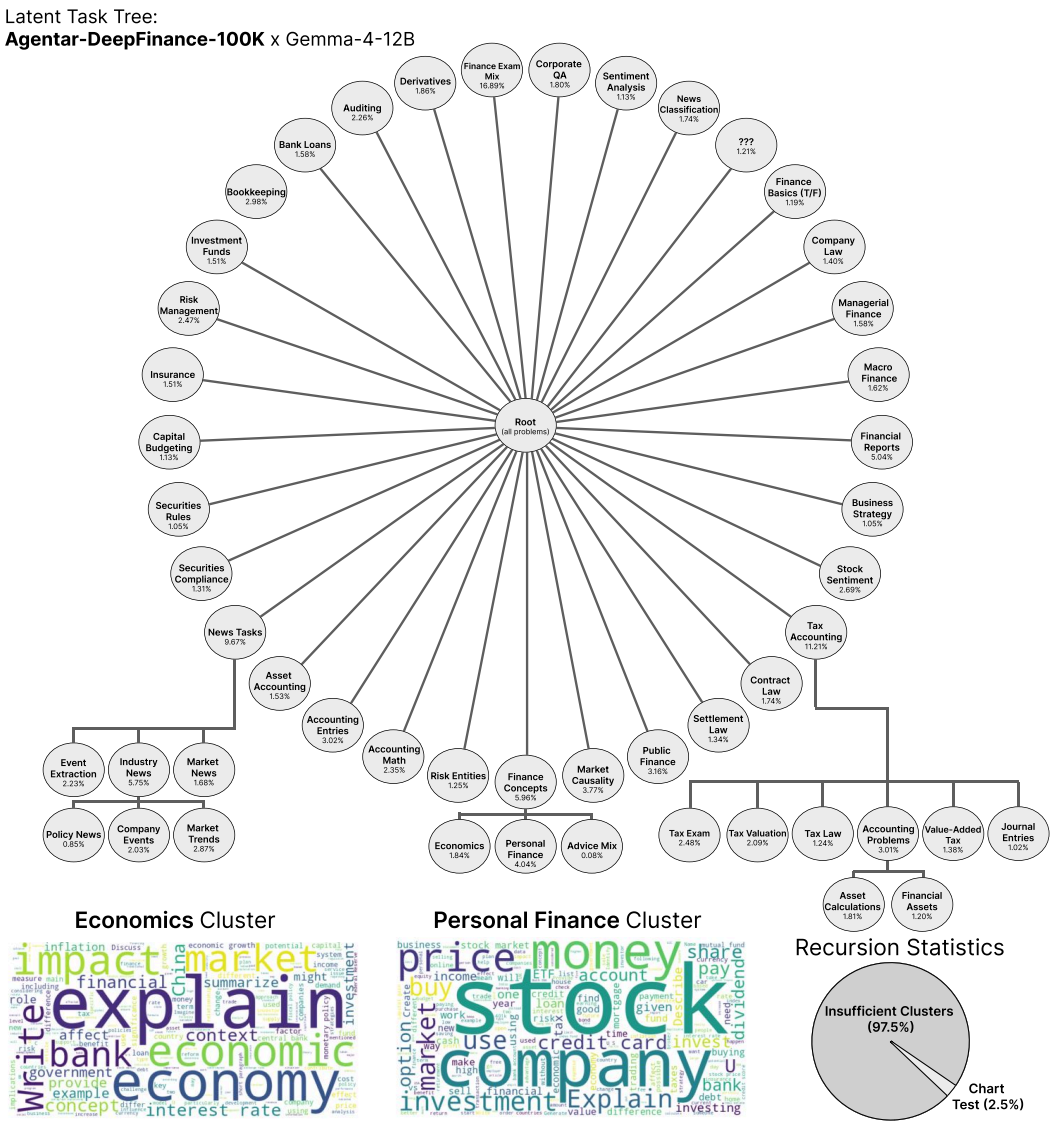

Figure 16. **Latent Task Tree for *Agentar-DeepFinance-100K* (Zhao et al., 2025) using the latent space of Gemma-4-12B (Team et al., 2024)** The dataset consists of 100K financial reasoning problems. The dataset is predominantly Chinese. A very small minority of prompts appear to be intended for SFT rather than RLVR. The induced tree contains 51 nodes. On one node with 8×H100 GPUs, loading the latent representations took 58:45 minutes and recursive tree construction took 48:19 minutes, for a total construction time of 1:47:04 hours.

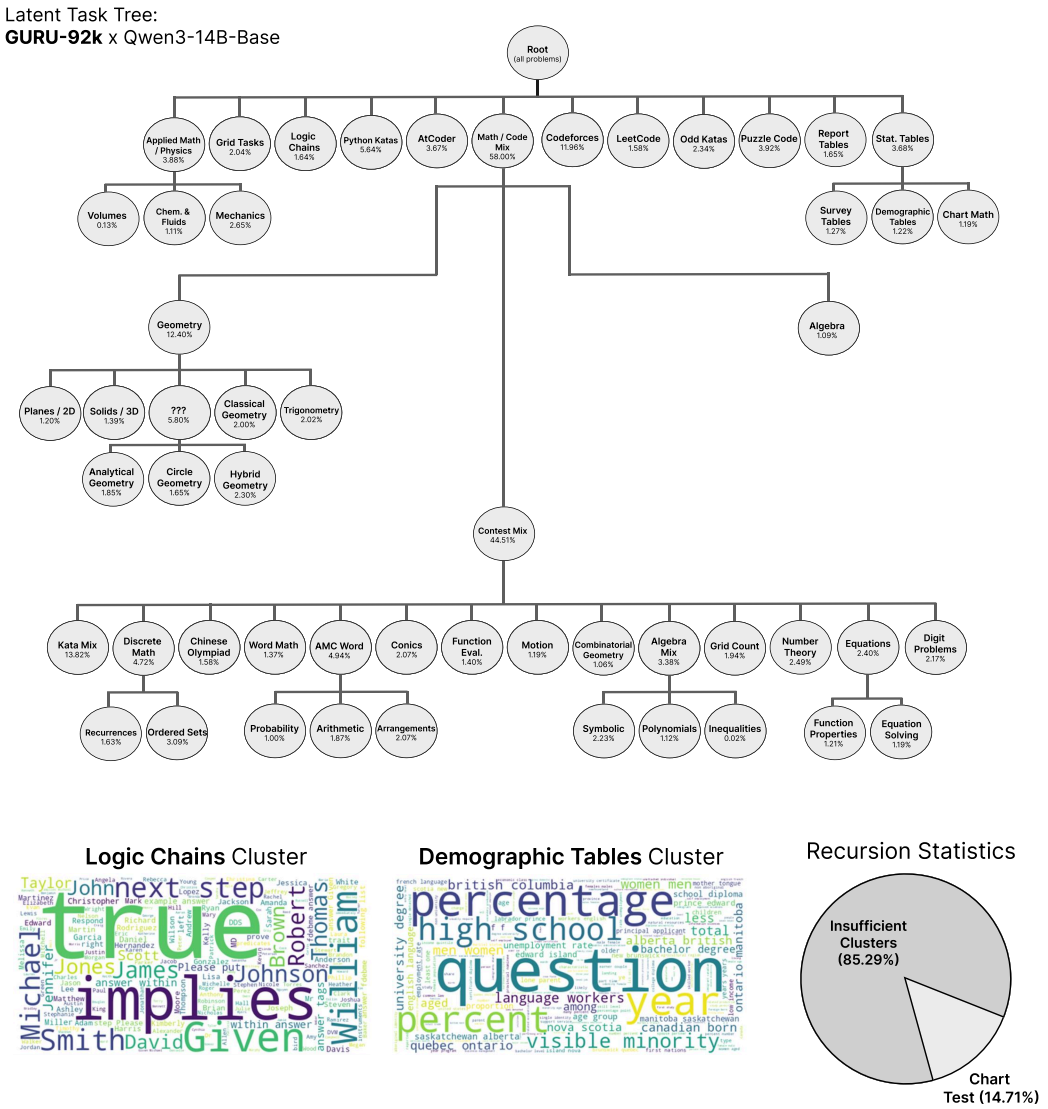

Figure 17. **Latent Task Tree for *GURU-92k* (Cheng et al., 2026) using Qwen3-14B-Base's latent space.** *GURU-92k* is a multi-domain reasoning dataset spanning math, code, science, logic, simulation, and tabular reasoning problems, allowing us to test whether latent task trees can organize heterogeneous training data beyond a single domain. The induced tree contains 54 nodes. On one node with 8×H100 GPUs, loading the latent representations took 1:25:16 hours and recursive tree construction took 52:31 minutes, for a total construction time of 2:17:53 hours. For scale, the original work reports a training wall-clock time of three days; thus, although this tree is more expensive to construct than those for smaller datasets, its construction time is still on the order of hours rather than days.

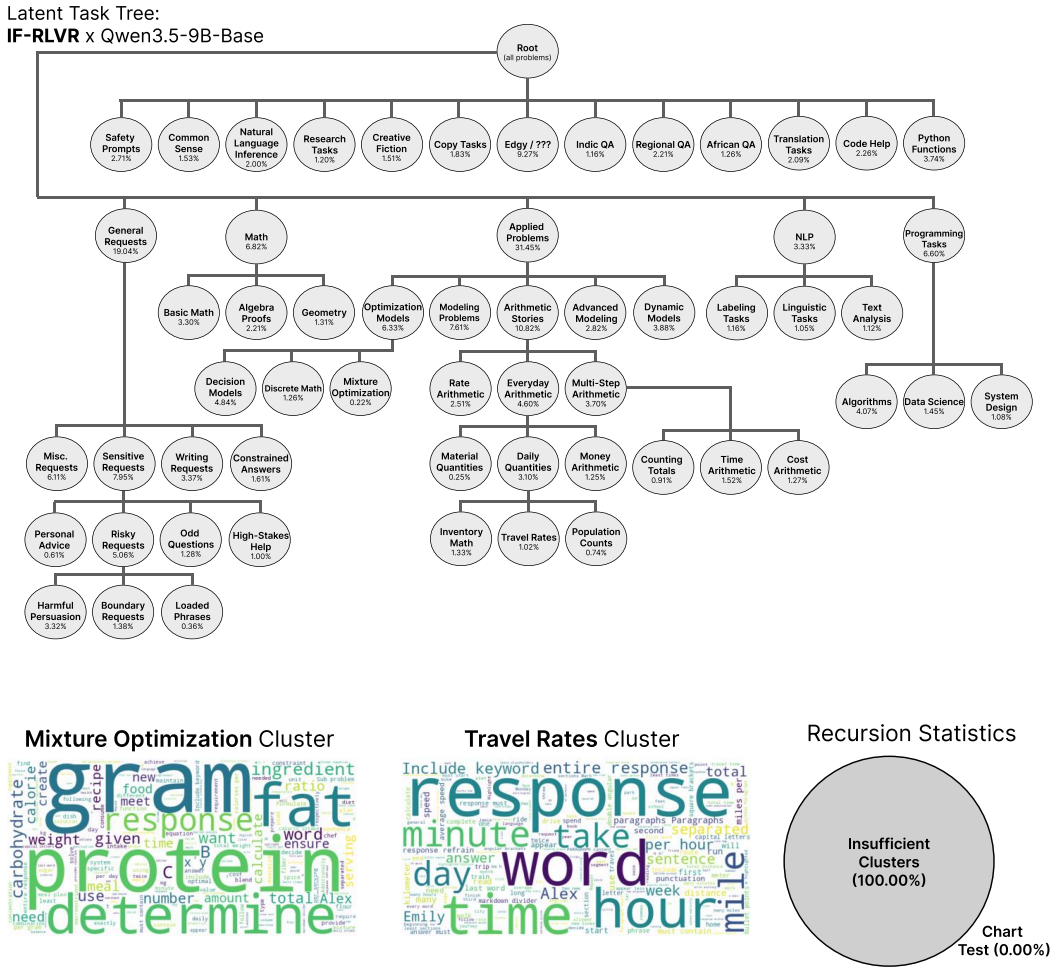

Figure 18. **Latent Task Tree for *IF-RLVR* (Pyatkin et al., 2025) using the latent space of Qwen3.5-9B-Base (Qwen Team, 2026)** The dataset consists of 95K instruction-following problems spanning mathematics, coding, NLP tasks, multilingual QA, writing, safety-sensitive requests, and other domains. Each prompt pairs a base task with automatically verifiable output constraints, so the induced hierarchy reflects both semantic topic and instruction-following form. The induced tree contains 59 nodes. On one node with (8×) H100 GPUs, loading the latent representations took 48:37 minutes and recursive tree construction took 48:32 minutes, for a total construction time of 1:37:09 hours.

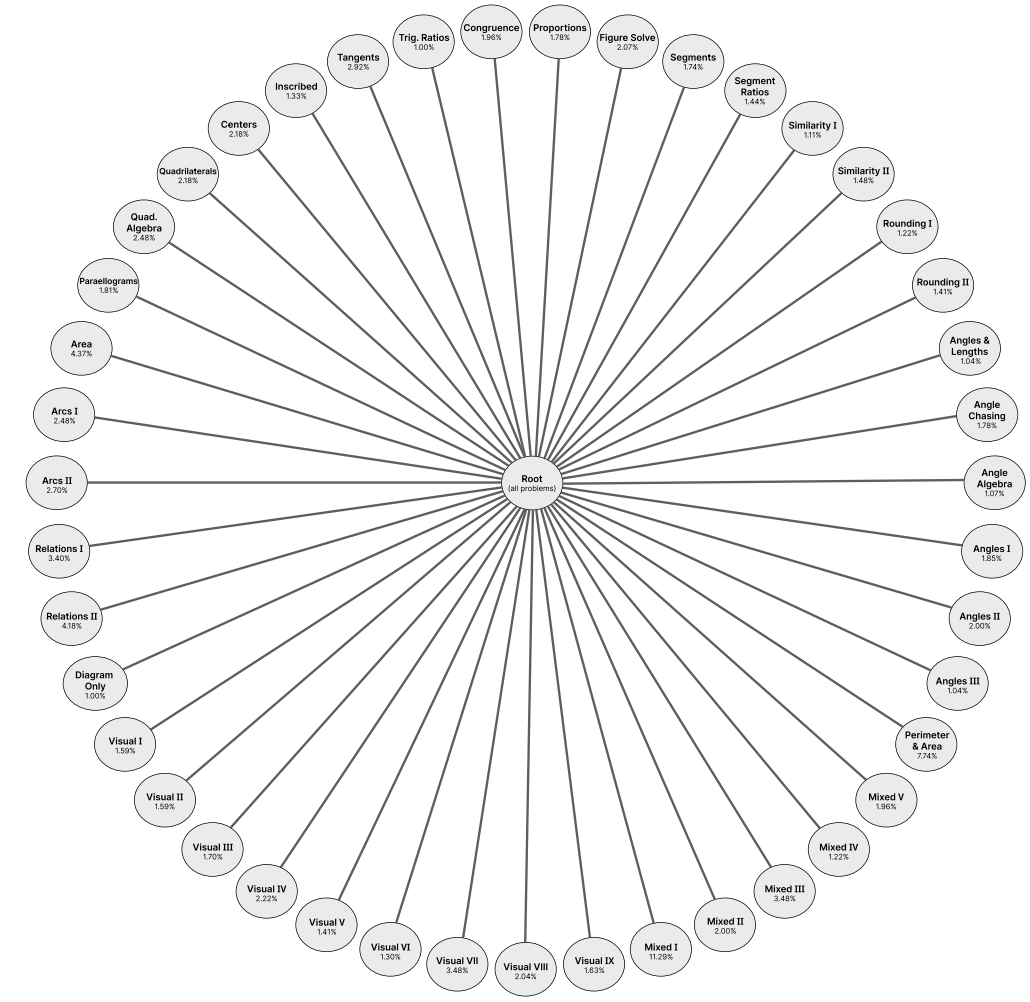

Figure 19. **Latent Task Tree for *Geometry3K* Lu et al. (2021) using Qwen3-VL-32B-Instruct's (Bai et al., 2025) latent space.** The dataset consists of multimodal geometry problems, and the induced tree contains 54 nodes. On one node with 8×H100 GPUs, loading the latent representations took 3:26 minutes, and recursive tree construction took 2:21 minutes, for a total construction time of 5:47 minutes. Because our cluster labels are assigned from example prompts and word clouds, several clusters with minimal or ambiguous text could not be cleanly differentiated without inspecting the associated diagrams. These cases suggest that the induced structure is not driven by language alone, but also reflects visual and multimodal properties of the problems. A finer-grained visual audit could further refine these labels, but the prompt-level analysis already shows that the latent tree captures structure beyond surface text.

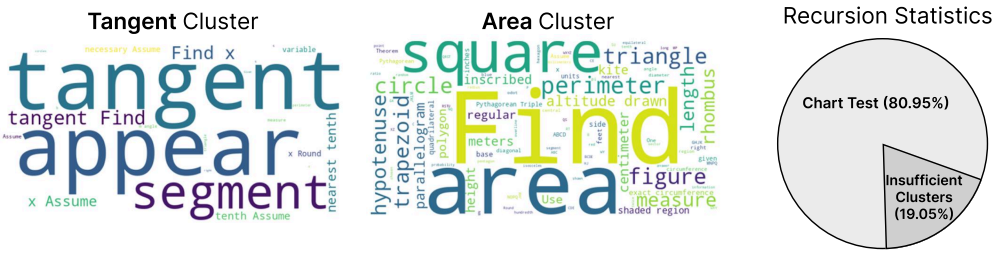

Figure 20. **Additional information for Figure 19**. Example word clouds and recursion statistics shown separately from the original figure to due space constraints.

G TREE-CONSTRUCTION SENSITIVITY ANALYSIS

We examine the sensitivity of Latent Task Tree construction to representative hyperparameters from each stage of the pipeline: PCA, UMAP, and HDBSCAN. We vary one hyperparameter at a time on `DAPO-Math-17K` using `Qwen3-8B-Base`, while holding all remaining settings fixed at their default values. We summarize each resulting tree using the number of **nodes** (N), **leaves** (L), **maximum depth** (D), **maximum width** (W), and **maximum branching factor** (B). Reported deltas follow the order (N, L, D, W, B) relative to the default tree.

PCA Variance Explained Ratio. PCA is applied locally at each recursive partition. Lower variance explained ratios (VER) impose stronger compression, whereas higher values retain additional directions that may contain either useful structure or noise.

PCA VER	N	L	D	W	B	$\Delta(\mathbf{N}, \mathbf{L}, \mathbf{D}, \mathbf{W}, \mathbf{B})$
0.85	49	42	5	11	11	$(+6, +4, +2, -13, -10)$
0.90	47	41	3	26	23	$(+4, +3, 0, +2, +2)$
0.95 (default)	**43**	**38**	**3**	**24**	**21**	—
0.99	47	41	4	25	22	$(+4, +3, +1, +1, +1)$

Table 4. **Sensitivity to the PCA variance explained ratio.** Tree summary statistics under one-at-a-time variation of the local PCA retention threshold.

The default produces the most compact hierarchy among these settings. VER $= 0.90$ and 0.99 produce only modest changes in node count, leaf count, width, and branching, with maximum depth changing by at most one. More aggressive compression at VER $= 0.85$ produces a deeper and substantially narrower hierarchy, indicating that stronger information removal redistributes substructure across levels rather than causing uncontrolled fragmentation.

UMAP Maximum Projection Dimension. We cap the UMAP output dimensionality to keep the local projection within a range suitable for density-based clustering. Lower maximum projection dimension (MPD) values impose stronger compression, while higher values retain more of the PCA representation.

UMAP MPD	N	L	D	W	B	$\Delta(\mathbf{N}, \mathbf{L}, \mathbf{D}, \mathbf{W}, \mathbf{B})$
25	42	37	3	24	21	$(-1, -1, 0, 0, 0)$
50 (default)	**43**	**38**	**3**	**24**	**21**	—
100	39	35	3	21	21	$(-4, -3, 0, -3, 0)$

Table 5. **Sensitivity to the UMAP maximum projection dimension.** Tree summary statistics remain similar across a fourfold range around the default configuration.

Tree-level statistics remain similar across the tested fourfold MPD range. MPD $= 25$ changes only one node and one leaf while preserving depth, width, and maximum branching exactly. MPD $= 100$ yields a somewhat more compact tree, but maximum depth and branching remain unchanged.

HDBSCAN Minimum Cluster Percentage. The minimum cluster percentage (MCP) specifies the smallest permitted cluster as a fraction of the *current recursively processed partition*; its absolute minimum size therefore adapts locally throughout the tree. Unlike the PCA and UMAP settings above, MCP directly controls the minimum granularity of the resulting partition.

As expected, MCP directly controls tree resolution. Increasing the threshold from 1% to 5% or 10% requires larger child regions and therefore produces substantially coarser hierarchies. The 5% configuration is narrower but slightly deeper, whereas the 10% configuration is both shallow and compact. These predictable changes motivate the lower default, which preserves the fine-grained heterogeneity that Latent Task Trees are intended to expose.

Summary. Overall, tree-level structure is relatively insensitive to moderate changes in the PCA retention threshold and UMAP projection cap around the default configuration, whereas HDBSCAN's

HDBSCAN MCP	N	L	D	W	B	$\Delta(\mathrm{N},\mathrm{L},\mathrm{D},\mathrm{W},\mathrm{B})$
0.01 (default)	**43**	**38**	**3**	**24**	**21**	—
0.05	17	12	4	7	4	$(-26,-26,+1,-17,-17)$
0.10	9	7	2	5	5	$(-34,-31,-1,-19,-16)$

Table 6. **Sensitivity to the HDBSCAN minimum cluster percentage.** Increasing the minimum permissible local cluster size predictably produces coarser Latent Task Trees.

minimum cluster percentage has the expected direct effect on resolution. This one-at-a-time sensitivity analysis complements our broader tree-construction stability analysis in Appendix F, where a single fixed configuration is applied across additional datasets, domains, model families, languages, and modalities without per-dataset tuning. Together, these results support the default configuration as a stable practical choice while also making no claim that it is uniquely optimal.

H FRONTIER IMBALANCE

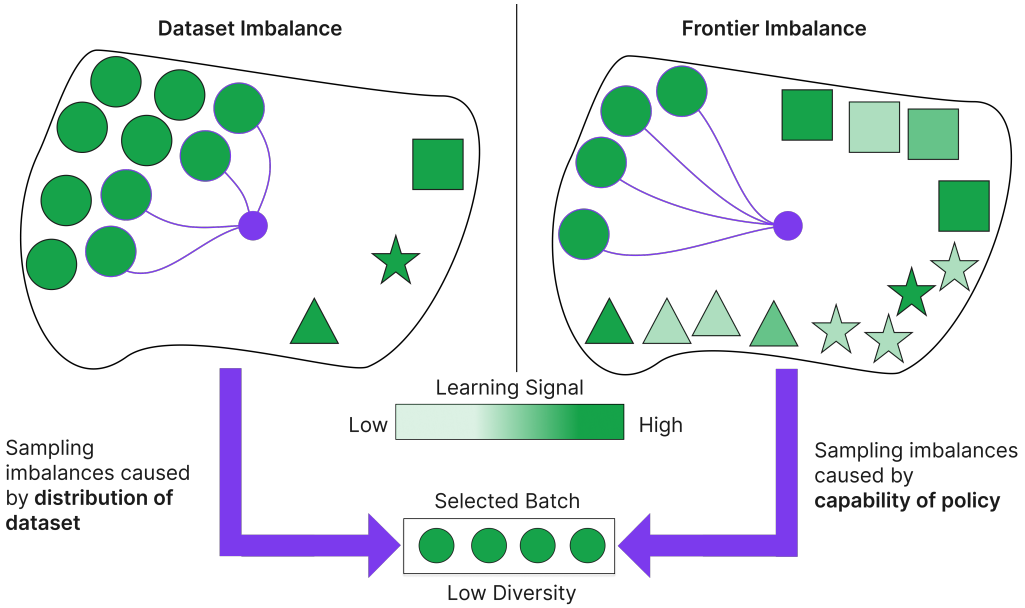

Figure 21. **Dataset imbalance and frontier imbalance.** Dataset imbalance arises from the static composition of the training set, while frontier imbalance arises from the policy-dependent distribution of problems that currently provide learning signal. A productivity-oriented sampler may therefore concentrate on a subset of problem types either because they are common in the dataset, overrepresented on the current frontier, or both.

Throughout this work, we emphasize that RL training datasets are heterogeneous and often conceptually imbalanced. If a dataset contains many more examples of one problem type than another, then a sampler that is not type-aware may allocate more training effort to the majority type simply because it appears more often. We refer to this as *dataset imbalance*.

However, adaptive sampling methods introduce a second and more subtle source of imbalance. At any point in training, only a subset of problems provides substantial learning signal under the current policy. Following the terminology of Foster et al. (2025), we refer to this subset informally as the model's *frontier of learnability*, or simply the *frontier*. Frontier imbalance occurs when this productive subset is itself unevenly distributed across problem types. Importantly, frontier imbalance is policy-dependent: as the model improves, the set of problems that are too easy, too hard, or productively learnable can change.

To illustrate the distinction, consider a dataset D consisting of two problem types, P_A and P_B. Suppose P_A makes up 80% of the dataset, while P_B makes up the remaining 20%. This is a dataset imbalance. Now consider two possible policies. For a first policy, the frontier may be balanced: among the problems that currently provide high learning signal, P_A and P_B may be represented equally. In this case, a productivity-oriented sampler may select useful examples from both types despite the underlying dataset imbalance.

For a second policy, the frontier may be imbalanced in the opposite direction: perhaps 80% of currently productive problems are of type P_B, even though P_B is only 20% of the dataset. In this case, a productivity-oriented sampler may concentrate training on P_B, not because P_B is more common, but because it is overrepresented on the current frontier. This is not necessarily undesirable: those problems may genuinely provide stronger learning signal at that stage of training. The issue is that productivity alone does not specify how training effort should be distributed across problem types, especially when the goal is broad improvement across a heterogeneous evaluation suite.

The most concerning case is when dataset imbalance and frontier imbalance reinforce each other. If P_A is both the majority type in the dataset and the majority type on the current frontier, then a

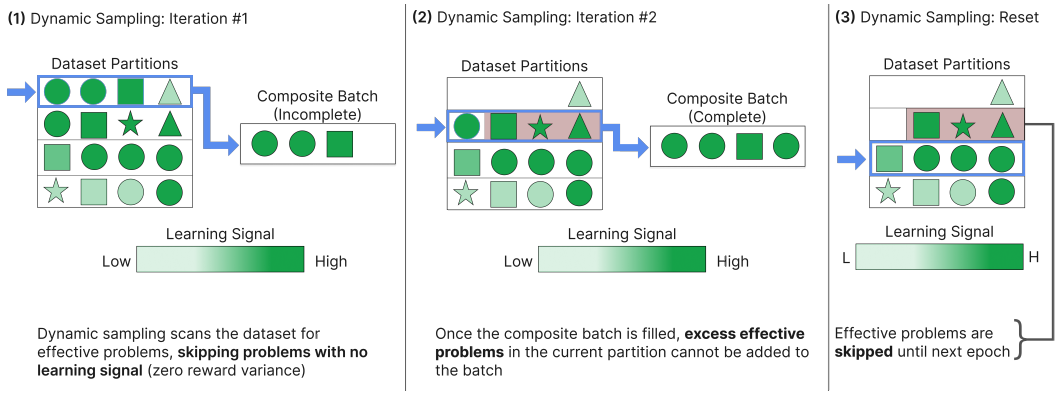

Figure 22. **Problem skipping in Dynamic Sampling.** Dynamic Sampling scans candidate prompts and adds prompts with nonzero reward variance to a composite batch. Once the composite batch is filled, later effective prompts in the current partition may be deferred until a future pass. When effective prompts are unevenly distributed across problem types or dataset partitions, this mechanism can interact with dataset and frontier imbalance. See Figure 23 for a consequence of this mechanism.

sampler that prioritizes learning signal may further concentrate training on P_A. Conversely, minority types may receive little exposure either because they are rare in the dataset, because they are rarely productive under the current policy, or because they are skipped by the mechanics of the sampling procedure. This can create a feedback loop in which some regions of the task space receive substantial training effort while others remain undertrained.

This distinction helps clarify the behavior of generate-then-discard methods such as Dynamic Sampling. Dynamic Sampling constructs effective batches by generating rollouts for candidate prompts and discarding prompts whose rollouts have zero reward variance. This ensures that the final composite batch contains prompts with nonzero learning signal. However, depending on implementation details, prompts that are effective but appear after the composite batch has already been filled may not be used until a later pass through the data. Figure 22 illustrates this mechanism. When productive prompts are unevenly distributed across dataset partitions or problem types, this kind of skipping can interact with dataset and frontier imbalance.

In our experiments, this issue is most visible in the English–Chinese mathematics split. As shown in Figure 23, the dataset is dominated by English mathematics problems, while Chinese mathematics problems form a smaller fraction of the training set. For the 8B model, Dynamic Sampling improves strongly on English mathematics evaluations but plateaus or underperforms on the Chinese mathematics evaluations relative to uniform sampling, despite constructing batches with high effective ratio. This pattern is consistent with the hypothesis that productivity-oriented filtering and problem skipping can underexpose minority regions of the dataset. However, the experiment does not isolate whether the effect is due to dataset imbalance, frontier imbalance, the skipping mechanism, or their interaction.

The behavior also differs across model sizes. For the 4B model, the same plateau is not observed. One possible explanation is that the smaller model's frontier contains a different mixture of productive prompts, giving Chinese mathematics problems more exposure under Dynamic Sampling. We treat this interpretation as suggestive rather than causal: the key point is that the frontier of learnability depends on the policy, and therefore sampling imbalances can change with model size and training stage.

Notably, this discussion is not meant to argue that Dynamic Sampling is inherently flawed. Alternative implementations could reduce or eliminate the skipping issue, for example by returning unused effective prompts to the active pool rather than deferring them to a later epoch. More broadly, the point is that adaptive sampling methods should account for the heterogeneous, multi-task nature of LLM training. A sampler can be highly effective at maximizing immediate learning signal while still allocating training effort unevenly across problem types. This motivates type-aware or structure-aware sampling methods such as BMC, which explicitly organize sampling over the latent task geometry rather than relying on productivity (difficulty) alone.

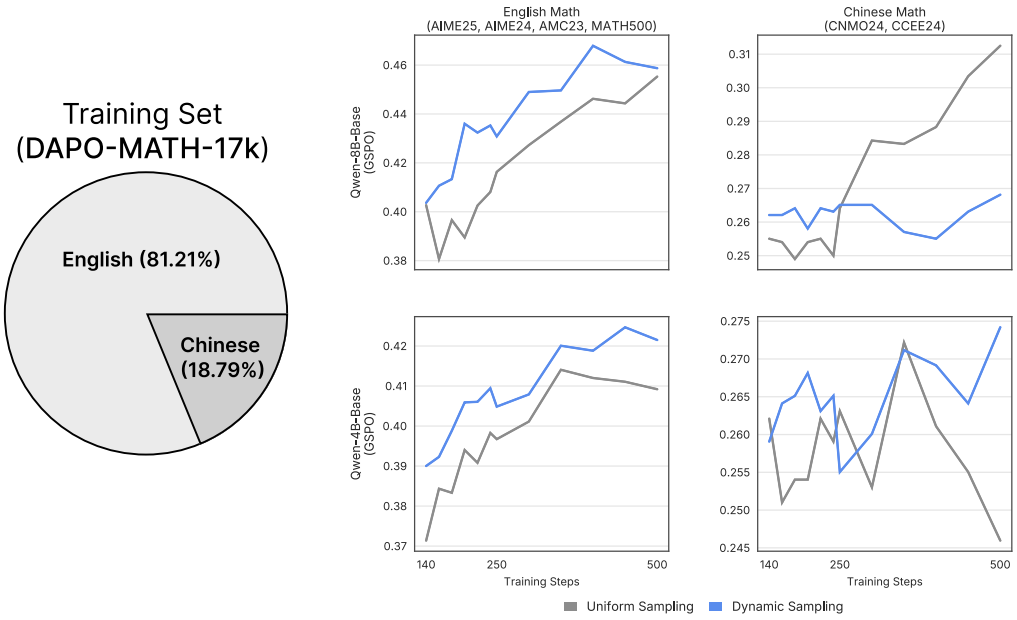

Figure 23. **Dynamic Sampling and minority-type exposure.** The training data is dominated by English mathematics problems, while Chinese mathematics problems form a smaller subset. For the 8B model, Dynamic Sampling improves strongly on English mathematics evaluations but plateaus on Chinese mathematics evaluations relative to uniform sampling. This behavior is consistent with an interaction between dataset imbalance, policy-dependent frontier imbalance, and problem skipping, though the experiment does not isolate these factors causally.

Finally, frontier imbalance is not necessarily a failure mode. In some stages of training, disproportionately sampling a subset of problem regions may be desirable if those regions genuinely provide stronger learning signal. BMC is therefore not designed to eliminate imbalance, but to make such imbalance structured and *explicitly controllable*. The concern is not that adaptive samplers focus training effort unevenly, but that this unevenness can arise implicitly, without direct awareness of whether it reflects dataset composition, policy-dependent frontier structure, sampling mechanics, or their interaction.

I UTILITY-AWARE SAMPLING AND THE EVALUATION-PROTOCOL TENSION

BMC-T uses a target distribution to estimate which parts of a heterogeneous training set are most relevant to a desired capability or evaluation. In our experiments, we instantiate this target distribution using held-out evaluation benchmarks in order to test whether different training problem types carry different evaluation relevance. Target examples are never used as training examples and cannot be sampled by the scheduler; they only influence the utility scores assigned to sampleable regions of the training distribution.

The broader motivation is that, in large RL training datasets, it is often unclear which problems are most important for inducing a desired capability. A dataset may contain many superficially relevant examples, but only some problem types may actually support transfer to the target behavior. For example, if the goal is to improve performance on a specific, extremely difficult class of competition mathematics, such as International Mathematical Olympiad (IMO)-style reasoning, one could construct a large heterogeneous math training set without training directly on IMO problems. A utility-aware scheduler could then use a representative target set to bias sampling toward training problems that the policy model views as nearby or relevant, allowing training effort to concentrate on problems that better support the desired capability.

This results-oriented use case is distinct from the role of target distributions in algorithmic evaluation. In algorithmic research, held-out benchmarks are typically intended to measure generalization independently of the training process, but each benchmark also emphasizes a particular distribution of capabilities. At the same time, benchmark performance is often treated as the primary evidence for a sampling method's effectiveness, even though heterogeneous training data may not contribute equally to those benchmarks. This creates an apparent paradox: the scheduler is judged by downstream evaluation performance, but evaluation relevance is not part of its objective unless target information is explicitly provided. Ignoring evaluation relevance can therefore obscure an important axis of curriculum design, while using an evaluation benchmark to guide sampling changes the role of that benchmark.

This motivates a protocol distinction. If a benchmark is used to define the target distribution for BMC-T, then it should be understood as part of the curriculum design process rather than as an untouched final test set. This is analogous to the role of a **validation set**: it may guide model or training decisions, but should not also serve as the final measure of generalization. Under a strict held-out protocol, one can instead designate a separate validation or development set as the target distribution, use it to construct the utility signal, and reserve separate benchmarks for final evaluation.

Constructing such a validation or target set is itself nontrivial. What appears relevant to humans may not be what is most relevant to the policy model. Human-defined data taxonomies, topic labels, or perceived similarity between problems may fail to capture how the model organizes the task space internally. As a result, a validation set that appears representative under human semantics may provide a weaker utility signal than expected if it is poorly aligned with the regions of the model's latent task manifold that contain useful training problems. Latent Task Trees can help diagnose this mismatch by showing how candidate target examples are distributed relative to the training data under the policy's own representation space. In this way, the same representation used for utility-aware sampling can also help design or audit validation sets intended to represent a desired capability.

We do not aim to resolve broader questions about benchmark awareness or train–test separation. These protocol choices depend on the goals of a given study: whether the objective is to build a target-aware training system, compare sampling algorithms under strict held-out evaluation, or diagnose which training problems support a particular benchmark. Our use of BMC-T is primarily diagnostic and methodological: it tests whether productivity and diversity fully explain downstream evaluation performance, or whether some problem types have greater influence on specific evaluation benchmarks. The results in Section 5.3 support the latter interpretation, showing that changing the target distribution can shift evaluation outcomes even when apparent training productivity remains similar.

J External Models

BMC constructs Latent Task Trees from the policy model's own hidden representations. This choice is somewhat different from many standard approaches to dataset organization, which often rely on semantic embedding models, frontier-model annotations, or manually defined task labels. We use the policy's internal representation space because our goal is not simply to recover human-interpretable task categories, but to approximate the structure of the problem space as perceived by the model being trained. We discuss this design choice below.

J.1 Semantic Embedding Models

A natural alternative is to cluster prompts using a pretrained embedding model. Such models are typically optimized to capture *human-facing* semantic similarity, for example by placing paraphrases, related documents, or retrieval-relevant texts close together in embedding space. This is useful for retrieval, ranking, and semantic search, but it is not necessarily the same notion of similarity needed for curriculum learning.

In BMC, the relevant geometry is the geometry of the training model's own behavior. For example, two prompts that are paraphrases, translations, or surface variants of the same problem may be semantically similar to a human or to a retrieval model, while still inducing different hidden representations, rollout behavior, reward variance, or learning dynamics in the policy model. Conversely, prompts that look different to humans may be treated similarly by the model if they activate similar internal reasoning patterns. Since the scheduler must decide which prompts are useful for training this particular policy, we use the policy model's hidden states rather than an external embedding model as the basis for task structure.

One possible concern is that generic embedding models may produce cleaner semantic clusters than raw hidden states from a non-embedding LLM. However, our Latent Task Tree visualizations suggest that the policy model's internal representations are sufficiently structured for curriculum construction: the induced clusters are often coherent and fine-grained, even though human-readable labels are added only post hoc for analysis.

J.2 Frontier Model Annotations & Prompt-based Clustering.

Another reasonable option is to ask a frontier model to label, summarize, or categorize each prompt, and then cluster prompts using those generated descriptions. This can produce highly interpretable taxonomies, and we use such annotations for analysis and visualization. However, we do not use them to construct the Latent Task Tree itself.

The main limitation is that a frontier model supplies an external semantic perspective. Its labels may reflect *human-aligned* categories or its own internal organization, rather than the representation geometry of the policy model being trained. For curriculum scheduling, this distinction matters: if the training model represents two problem types similarly, confuses them in similar ways, or generalizes between them, that relationship should be reflected in the scheduler even if an external annotator would label them differently.

There are also practical considerations. Prompt-based labeling can become expensive or cumbersome at large scale, especially for multimodal datasets where each example may require image, audio, or structured-context understanding. In contrast, if the policy model's hidden states are accessible, the same representation-extraction pipeline can be applied across modalities supported by the model.

A related alternative is to prompt the same model being trained to describe or cluster the dataset. This is closer in spirit to our goal, but it still replaces internal representation geometry with verbal self-description. Model-generated explanations or labels may be useful, but they need not faithfully expose the model's internal organization of prompts (Parcalabescu & Frank, 2024). We therefore treat prompt-based labeling as an analysis tool rather than as the source of task structure.

J.3 Sparse Autoencoders.

Sparse autoencoders (SAEs) provide another possible route for analyzing model representations. SAEs learn sparse dictionaries over activations, often decomposing hidden states into more in-

terpretable features (Cunningham et al., 2023; Kim et al., 2026). This makes them valuable for mechanistic interpretability and for identifying *human-interpretable* components of model computation.

However, human interpretability is not the primary goal of BMC. The Latent Task Tree is used as a curriculum structure, not as a mechanistic explanation of individual features. While SAE features could potentially provide a useful basis for task discovery, training and validating SAEs introduce substantial additional complexity, including choices of layer, dictionary size, sparsity penalty, feature aggregation, and feature-to-prompt representation. For this work, we instead use normalized hidden-state representations directly, which keeps the tree construction pipeline simpler while remaining tied to the policy model's own latent geometry.

J.4 Critic-Style Prompt Predictors

Another alternative is to train an additional model to estimate prompt difficulty, learning potential, or policy-improvement utility for curriculum selection (Gao et al., 2025; Mahrooghi et al., 2026; Gu et al., 2026). BMC takes a different approach to the same underlying problem of estimating which prompts should receive training effort. Rather than training a separate predictor to directly regress a policy-dependent target, BMC uses the policy model's own hidden representations to define a latent geometry over prompts, then maintains online Bayesian beliefs over learning signal within that geometry.

This shifts the modeling burden from learning a high-dimensional productivity function to maintaining local, uncertainty-aware estimates and sharing evidence between geometrically related prompts. Since prompt productivity changes as the policy evolves, we view this as a useful inductive bias: the tree supplies structure, while surprise-based forgetting and uncertainty growth allow beliefs to adapt to non-stationarity. This does not rule out learned productivity predictors, which may be valuable when sufficiently accurate and responsive. Rather, BMC provides a complementary alternative that avoids adding a separate critic or teacher model as the central mechanism. A secondary benefit is that once curriculum selection is organized over latent geometry, the same structure can also support explicit control over type coverage (diversity) and utility, as in BMC-T.

Summary. Overall, external models can be useful for labeling, validation, and analysis, but BMC deliberately avoids making them the source of curriculum structure. The Latent Task Tree is derived from the policy model's own hidden representations because, for curriculum scheduling, the relevant semantics are the semantics of *the model being trained* rather than an external model's taxonomy. This also preserves one of the practical advantages of group-relative RLVR: curriculum selection can be added without introducing a separate critic, teacher, annotator, or embedding model as a required component. BMC therefore remains a lightweight extension of the training loop while still providing structured control over productivity, diversity, and utility.

K LIMITATIONS

K.1 TREE CONSTRUCTION HEURISTICS

The recursive construction of the Latent Task Tree is heuristic. Although we find that the current pipeline produces coherent, fine-grained trees across different models, datasets, languages, domains, and modalities, there may be better choices for the latent representation, layer selection, dimensionality reduction, manifold approximation, stopping criterion, or clustering algorithm.

These choices also introduce hyperparameters whose effects are difficult to evaluate in isolation. Constructing trees is relatively inexpensive compared to RL training, but determining whether one tree is better than another for *curriculum scheduling* ultimately requires running downstream RL experiments. This makes comprehensive tree-construction ablations substantially more expensive than tree generation itself. We therefore view the current construction pipeline as a practical instantiation rather than an optimized solution, and leave systematic exploration of alternative tree-building procedures to future work.

K.2 COMPUTATIONAL REQUIREMENTS

Latent Task Trees are broadly applicable across model sizes, domains, languages, and modalities, but they require extracting and processing hidden representations for the training prompts. If the combination of model size and dataset size is sufficiently large, the default tree-construction pipeline may become computationally or memory intensive.

Our current implementation primarily uses CPU-based preprocessing for tree construction. This is not a fundamental requirement: GPU-accelerated variants of PCA, UMAP, and HDBSCAN could reduce construction time substantially (e.g., (Nolet et al., 2021)). However, such a GPU-based implementation is not included at the time of writing. Since LLM training is already heavily GPU-dependent, an optimized GPU pipeline would be a useful engineering improvement. For very large datasets or models, additional memory-efficient techniques, such as streaming PCA, approximate nearest-neighbor construction, batching, or subsampling-based initialization, may also be necessary.

K.3 STRUCTURE DRIFT

The Latent Task Tree is constructed once before training and then held fixed throughout RL. We analyze this issue in Appendix C. Our results suggest that a static tree can remain useful during training, but this remains a limitation: as the policy changes, the model's hidden representations may drift, and the latent geometry used to construct the original tree may become less aligned with the current policy.

One natural mitigation is to periodically reconstruct the tree from updated model representations. We explored this direction empirically, but online reconstruction introduces practical overhead. In our implementation, rebuilding the tree required interrupting and resuming training, since latent extraction and RL training could not be performed conveniently with all required models loaded simultaneously. This is primarily an engineering limitation rather than a conceptual one: prompt-level beliefs can be reassigned to the corresponding leaves after reconstruction using prompt identities, and the scheduler's action space is simply defined by the newly constructed tree.

The remaining question is whether the benefit of periodic reconstruction justifies its cost. In our experiments, periodic reconstruction did not substantially change the main conclusions, suggesting that the initial latent geometry is sufficiently stable for the training horizons considered here. However, representation drift may become more important for longer training runs, larger policy updates, or settings where the model's capabilities change more dramatically. We therefore leave more efficient online tree reconstruction and adaptive update schedules to future work.

K.4 BELIEF BOUNDS AND UNCERTAINTY SATURATION

BMC was designed to avoid introducing many additional tuning coefficients. Aside from tree-construction choices, the main scalar hyperparameters in the belief updates are the bounds on μ_i, λ_i, and σ_i^2, together with the sigma-rule constant used to determine the maximum uncertainty bound

(Appendix M.3). Other quantities, such as the coverage coefficient used for staleness accumulation, are determined by the data and batch size rather than manually tuned.

Nevertheless, the resulting uncertainty dynamics may still require further study. In particular, we observed that many prompt-level beliefs reached the maximum uncertainty bound by the end of training, with 80% of prompts reaching maximum uncertainty by the end of the main training runs. This may indicate that the current uncertainty-growth rule re-inflates stale prompts more aggressively than necessary, or that the maximum variance bound determined by the sigma rule is not well matched to the observed learning-signal dynamics. If too many prompts saturate at maximum uncertainty, the posterior may become less informative for distinguishing genuinely uncertain prompts from merely stale ones.

This suggests several possible refinements. For example, staleness could accumulate sublinearly rather than through the current coverage-adjusted counter, or the uncertainty injection term could use a different saturating transform. Similarly, the sigma-rule bound could be adjusted to better match the empirical range of transformed learning signals. We leave these variants to future work.

Despite these limitations, the current bounded update rule was sufficient to improve learning signal and effective ratio in the main experiments. The behavior was less consistent in the medical-domain experiments in Appendix E, where learning signal decreased over time for both BMC and the Difficulty Only ablation. This may reflect the relative difficulty or reward structure of the dataset, overly aggressive uncertainty growth, or a more fundamental mismatch between the learning-signal statistic and that domain.

L Additional Experimental Details

L.1 Hyperparameters

Table 7. **Hyperparameters.** We detail hyperparameters central to our experimental setup below.

Hyperparameter	Value
Model Generation	
Train Temperature	1.0
Train Top-P	1.0
Evaluation Temperature	0.6
Evaluation Top-P	0.95
Rollouts-per-prompt (k)	8
Maximum Input Length	3000
Maximum Response Length	4096
Inference Engine	vLLM
Optimization	
Optimizer	AdamW
Learning Rate	0.000001
Learning Rate Warmup Steps	0
Grad Clip	1.0
Weight Decay	0.0
Batch Size	128
Mini-Batch Size	32
Training Engine	FSDP2
GSPO	
Clip Ratio (High)	0.0004
Clip Ratio (Low)	0.0003
Use KL	No
Entropy Coef	0.0
GRPO	
Clip Ratio (High)	0.2
Clip Ratio (Low)	0.2
Use KL	Yes
KL Coefficient	0.001
Entropy Coef	0.0
Tree Construction	
Seed	0
Layer Depth	0.75
Variance Explained Ratio (PCA)	0.95
KNN Min Points for Connectivity (Chart Test)	500
KNN Connectivity Threshold (Chart Test)	10
TwoNN Dimension Threshold (Chart Test)	1.5
Min. Distance (UMAP)	0.0
Max Projection Dimension (UMAP)	50
N Neighbors (UMAP)	30
Min. Cluster Percentage (HDBSCAN)	0.01
Min. Global Cluster Size (HDBSCAN)	5
BMC	
Initial μ	0.0
Initial λ	1.0
Minimum λ	0.2
Maximum λ	10
Minimum σ^2	0.05
σ^2 Rule (determines maximum σ^2)	3
γ (BMC-T only)	0.25
ρ (BMC-T only)	0.05

L.2 Compute

Across all experiments, training was performed on a single H100 GPU node. Due to machine availability, the node interconnect varied across runs, with both Ethernet- and InfiniBand-connected machines used. Latent task tree construction was performed on CPU, using machines with approximately 32 CPU cores. All methods were run under the same hardware constraints, and comparisons are made within matched experimental settings.

L.3 Round-Robin Initialization

At the start of training, before adaptive sampling is activated, we perform a round-robin initialization pass for BMC and its ablations. During this phase, each training prompt is sampled once using the same rollout allocation used throughout training, yielding an initial estimate of its learning signal. This avoids a cold-start regime in which the scheduler must compare observed prompts against prompts that have not yet been sampled, and provides initial empirical statistics for information sharing across the Latent Task Tree. We use this initialization for experimental stability and compute efficiency; it is not a requirement of the method, and alternative cold-start strategies could be used in settings where a full round-robin pass is impractical.

BMC and Difficulty Only are initialized from the same post-round-robin policy checkpoint and the same initial prompt-level belief statistics. For consistency in model exposure, Tree Only also starts from the same post-round-robin policy checkpoint. However, because Tree Only is designed to isolate the effect of the Latent Task Tree without adaptive belief learning, it uses fixed uniform beliefs rather than the round-robin learning-signal estimates. Thus, BMC, Difficulty Only, and Tree Only share the same policy state when adaptive sampling begins, while differing only in how sampling decisions are made thereafter.

L.4 Evaluation Datasets

We evaluate on a mixture of competition mathematics and scientific reasoning benchmarks. The mathematics benchmarks test exact-answer problem solving across varying levels of difficulty, while GPQA evaluates out-of-domain scientific reasoning beyond the mathematical training distribution.

AIME. AIME consists of problems from the American Invitational Mathematics Examination, a challenging high-school mathematics competition. We evaluate on AIME 2024 and AIME 2025 using public dataset releases (Maxwell-Jia, 2024; yentinglin, 2025). These benchmarks test olympiad-style mathematical reasoning across topics such as algebra, geometry, number theory, and combinatorics, with answers given as integers between 000 and 999.

AMC 2023. AMC 2023 consists of problems from the American Mathematics Competition, drawn from the AI-MO validation set (AI-MO, 2024). Compared with AIME, AMC problems are generally shorter and more accessible, but still require nontrivial mathematical reasoning across algebra, geometry, counting, probability, and number theory.

MATH500. MATH500 is a 500-problem subset of the MATH benchmark (Hendrycks et al., 2021), commonly used to evaluate mathematical reasoning in large language models. The dataset covers a broad range of competition-style mathematics topics, including algebra, geometry, number theory, counting and probability, intermediate algebra, prealgebra, and precalculus.

CNMO 2024. CNMO 2024 consists of problems from the China Mathematical Olympiad, drawn from LiveMathBench (OpenCompass, 2024). The benchmark evaluates olympiad-style mathematical reasoning in Chinese, emphasizing non-routine problem solving, multi-step derivations, and advanced mathematical concepts.

CCEE 2024. CCEE 2024 consists of mathematics problems from China's College Entrance Examination, also drawn from LiveMathBench (OpenCompass, 2024). Compared with olympiad-style benchmarks such as AIME and CNMO, CCEE reflects a standardized national examination setting and tests mathematical reasoning across the Chinese high-school curriculum.

GPQA. GPQA (Rein et al., 2023) is a graduate-level multiple-choice question-answering benchmark designed to evaluate scientific reasoning in domains such as biology, physics, and chemistry. Questions are written by domain experts and are intended to be difficult for non-experts even with access to search. We report results on GPQA-Diamond, a more selective subset containing especially high-quality and challenging questions.

L.5 Dense-to-Sparse Evaluation Schedule

Evaluation can be a substantial source of cost during RL training, especially when tracking both training-set absorption and downstream benchmark performance across long runs. In this work, we evaluate two complementary quantities: training-set pass@1, which measures how quickly the policy absorbs the training distribution, and downstream mean@16 evaluation performance, which measures generalization to held-out benchmarks. Running both evaluations densely throughout training would substantially increase the financial cost of the experiments.

We therefore use a simple dense-to-sparse evaluation schedule. During the early phase of training, when policy performance often changes rapidly, we evaluate every 10 optimizer steps. To reduce cost, these early evaluations are interleaved: one checkpoint evaluates training-set pass@1, while the next evaluates downstream mean@16 performance. After step 250, we switch to a sparser schedule, evaluating every 50 optimizer steps. At these later checkpoints, we evaluate both training-set pass@1 and downstream mean@16 performance together.

This schedule can be viewed as a simple form of *annealed evaluation*: evaluation frequency is high when the model is changing quickly and lower once training dynamics have slowed. This allows us to measure early learning dynamics while still tracking longer-term behavior at substantially lower cost than fully dense evaluation. The trade-off is that the later part of each curve is sparser, so it may miss short-lived fluctuations or brief regressions. We therefore interpret late-stage evaluation results primarily as coarse trends rather than as fine-grained measurements of every training transition.

L.6 Evaluation Curves

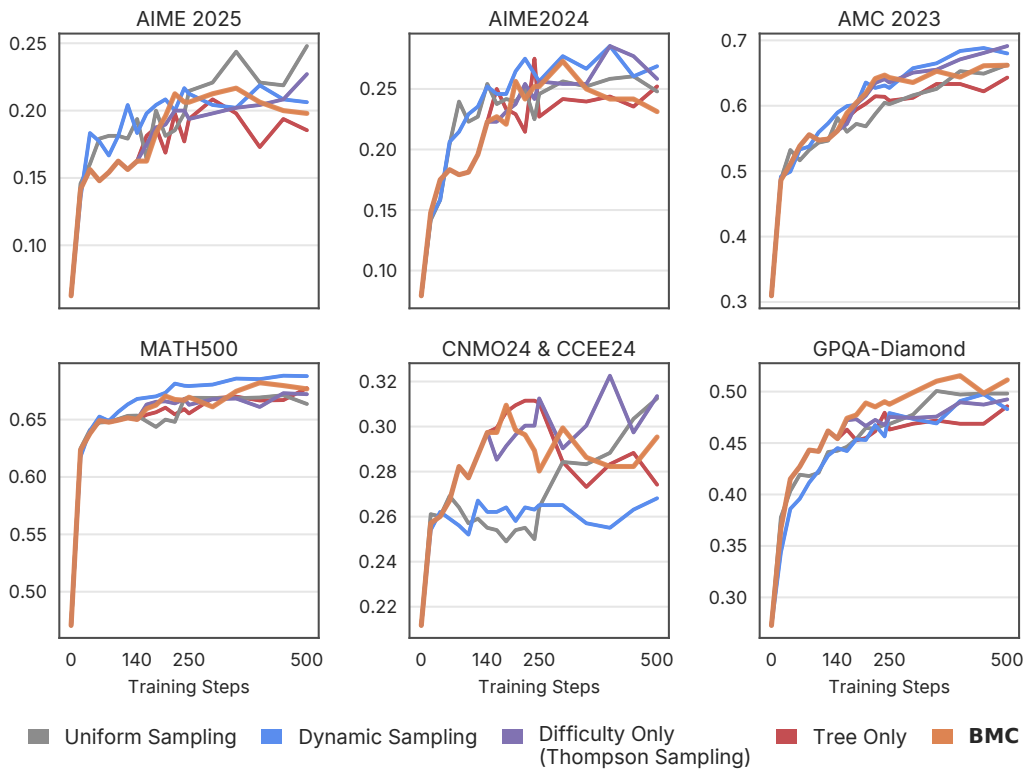

Figure 24. **Evaluation curves.** Performance across evaluations for `Qwen3-8B-base` while training on `DAPO-Math-17k` using **GSPO**. This corresponds to the bar graphs visualized in Figure 5. We refer to Appendix H for additional intuition and discussion on why the performances between the 8B and 4B models can vary on the same dataset.

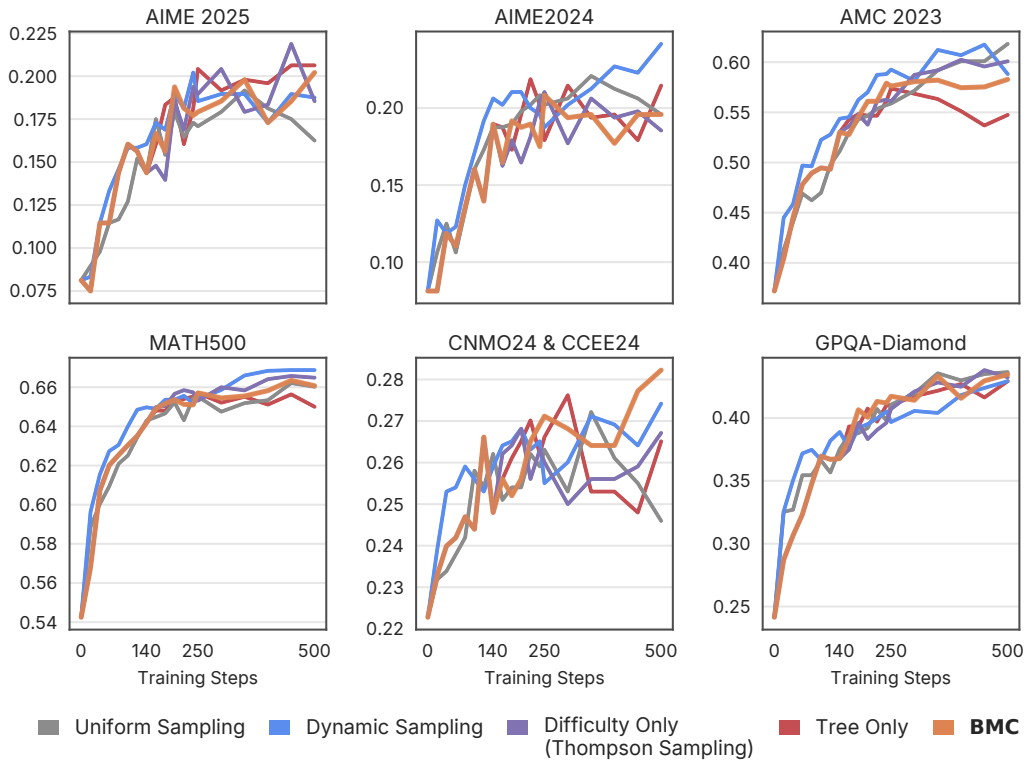

Figure 25. **Evaluation curves.** Performance across evaluations for `Qwen3-4B-base` while training on `DAPO-Math-17k` using **GSPO**. We refer to Appendix H for additional intuition and discussion on why the performances between the 8B and 4B models can vary on the same dataset.



Figure 26. **Evaluation curves.** Performance across evaluations for `Qwen3-4B-base` while training on `DAPO-Math-17k` using **GRPO**. We refer to Appendix H for additional intuition and discussion on why the performances between the algorithms can vary on the same model-dataset combination.

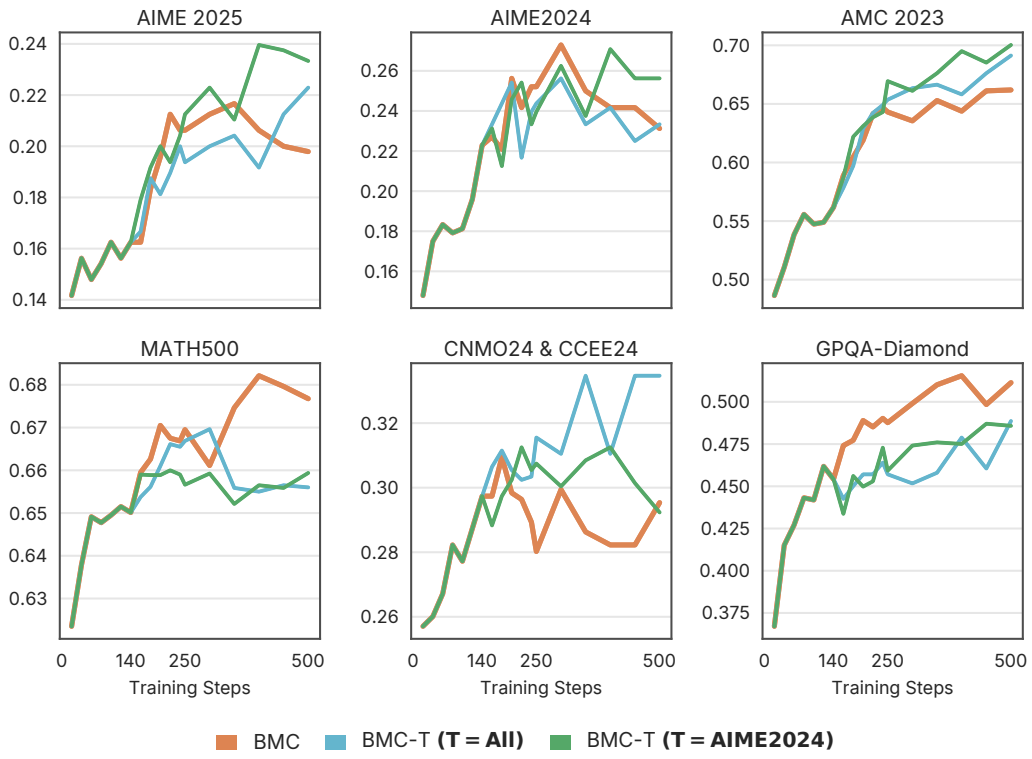

Figure 27. **Evaluation curves.** Performance across evaluations for `Qwen3-8B-base` while training on `DAPO-Math-17k` using variations of BMC-T. These evaluation trajectories correspond to the bar graphs visualized in Figure 6.

M Additional Algorithmic Details

M.1 Learning Signal

In our framework, the reward maximized by the multi-arm bandit scheduler is referred to as the *learning signal*. It is not the same reward that is maximized by the policy. Here we provide more details regarding this signal not given in the main body of work.

M.1.1 Quantifying the Learning Signal

Applying a bandit scheduler to RL training requires a scalar feedback signal for each sampled prompt. We refer to this quantity as the *learning signal*: a prompt-level statistic that should be inexpensive to compute and correlated with expected policy improvement. In RLVR, a natural choice is the variance of rewards across multiple rollouts from the same prompt. Reward variance appears frequently in adaptive curriculum learning methods for RLVR because it captures whether a prompt induces meaningful within-group contrast: if all sampled rollouts receive the same reward, then the prompt provides little or no relative signal for assigning credit among sampled responses.

We can formalize this intuition under a standard group-relative policy optimization setting, such as GRPO, DAPO, or GSPO. Consider a prompt x_i with k sampled responses $\{y_{i,j}\}_{j=1}^{k}$ and corresponding rewards $\{r_{i,j}\}_{j=1}^{k}$. Let

$$\bar{r}_i = \frac{1}{k}\sum_{j=1}^{k} r_{i,j}$$

denote the within-prompt mean reward. A typical group-relative advantage estimator takes the form

$$A_{i,j} \propto r_{i,j} - \bar{r}_i,$$

or, in normalized variants,

$$A_{i,j} = \frac{r_{i,j} - \bar{r}_i}{s_i + \epsilon},$$

where s_i is the within-prompt reward standard deviation. The policy gradient contribution for a sampled response is weighted by this advantage:

$$\nabla_\theta J(\theta) \approx \mathbb{E}_{x_i, y_{i,j}}\left[A_{i,j} \nabla_\theta \log \pi_\theta(y_{i,j} \mid x_i)\right].$$

Thus, when all rollouts for a prompt receive the same reward, the unnormalized group-relative advantages are zero:

$$r_{i,1} = \cdots = r_{i,k} \quad \Longrightarrow \quad A_{i,j} = 0 \ \ \forall j.$$

Such prompts may be either too easy, with all rollouts succeeding, or too difficult, with all rollouts failing. In both cases, the prompt provides little relative information for improving the policy.

For binary rewards, this connects directly to the common intermediate-difficulty heuristic. If $r \in \{0, 1\}$ and the model solves a prompt with probability p, then the reward variance is

$$\mathrm{Var}(r) = p(1-p),$$

which is maximized at $p = 0.5$. Equivalently, for rewards in $\{-1, 1\}$,

$$\mathrm{Var}(r) = 4p(1-p),$$

which is also maximized at $p = 0.5$. Therefore, prioritizing prompts with high reward variance is equivalent, in the binary-reward case, to prioritizing prompts near a 50% success rate. These prompts lie between trivial and impossible cases, making them more likely to provide informative credit-assignment signal. This is analogous to curriculum-learning intuitions such as a "Goldilocks" level of difficulty or, more historically, the Zone of Proximal Development (Vygotsky, 1978).

More generally, to also account for *continuous* rewards, we note that reward variance is closely related to the magnitude of the group-relative advantage. In the unnormalized case,

$$\frac{1}{k}\sum_{j=1}^{k} A_{i,j}^2 \propto \frac{1}{k}\sum_{j=1}^{k} (r_{i,j} - \bar{r}_i)^2,$$

so the mean squared advantage is proportional to the empirical reward variance. Likewise, objectives that prioritize the expected absolute advantage are closely related to prioritizing the within-prompt reward standard deviation. This provides a simple explanation for why adaptive curriculum methods based on reward variance (Foster et al., 2025; Wang et al., 2025a), intermediate-difficulty heuristics (Qu et al., 2025; Shen et al., 2025; Hu et al., 2025), expected absolute advantage (Wang et al., 2025b; Chen et al., 2025), or related notions of information gain (Zhang et al., 2025a; Zeng et al., 2025) can often select similar prompts in practice. While these methods differ in their objectives and implementations, they share a common emphasis on prompts that produce nontrivial rollout-level contrast.

There also exists a more subtle reason for this choice of learning signal. Reward variance is not the full policy gradient: the gradient also depends on the score term $\nabla_\theta \log \pi_\theta(y \mid x)$, and prompts may differ in their gradient geometry. In principle, one could define a more direct learning signal using gradient norms, Fisher information, Hessian approximations, or other curvature-based quantities. However, such quantities are substantially more expensive to compute and may introduce task-dependent scale effects. For example, prompts from different domains may induce systematically different token-level likelihoods, response lengths, or gradient magnitudes, causing a gradient-norm-based scheduler to emphasize some task types for reasons unrelated to their usefulness for learning (Wu et al., 2025).

Reward variance provides a cheaper and more normalized alternative. It is computed entirely from rollout rewards, is naturally prompt-local, and places binary and continuous reward signals on a common scale when appropriately normalized. We therefore use reward variance as a practical proxy for the availability of learning signal: it identifies prompts whose sampled responses produce meaningful differences in outcome, and hence prompts for which group-relative policy optimization can assign nontrivial credit.

M.1.2 Normalizing the Learning Signal

In traditional multitask reinforcement learning, it is generally considered good practice to normalize rewards across heterogeneous tasks in order to maintain balanced learning dynamics and avoid overemphasizing tasks with naturally larger reward scales (Hafner et al., 2024; Hansen et al., 2024). We believe the same principle applies to RL training for LLMs, particularly when reward magnitudes differ across prompts, tasks, or datasets.

For this reason, while not strictly required, our framework assumes that the learning signal is normalized to the range $[0, 1]$ whenever possible. If the underlying rewards are bounded and the bounds are known, this normalization can be implemented with standard rescaling procedures. For example, given a learning signal s with known minimum and maximum values $s_{\min}$ and $s_{\max}$, one may apply:

$$s_{\text{norm}} = \frac{s - s_{\min}}{s_{\max} - s_{\min}}.$$

More generally, any monotonic transformation that maps heterogeneous reward scales to a shared range may be used. This ensures that the scheduler compares prompts using a consistent notion of learning signal magnitude, rather than inadvertently prioritizing prompts whose rewards are numerically larger for unrelated reasons.

Importantly, normalization is not fundamentally required by our framework. Since the scheduler models learning signals using Gaussian beliefs, it can in principle represent arbitrary continuous quantities, including unbounded signals. However, if learning signals from different prompts or tasks reside on substantially different scales and cannot be normalized appropriately, sampling imbalance issues may arise. In such settings, the scheduler may over-prioritize certain prompts due purely to scale differences rather than genuine informativeness. We leave a more detailed investigation of unbounded and heterogeneously scaled learning signals to future work.

M.2 Extended Descriptions of BMC Components

In the main paper, we presented the high-level components of Bayesian Manifold Curriculum (BMC). Here, we provide additional conceptual and implementation details regarding latent task tree construction, hierarchical Thompson sampling, belief updates, and practical considerations used in our experiments.

M.2.1 Extended Description of Top-Down Problem Selection (Hierarchical Thompson Sampling)

To construct a training batch, we treat each sample in the batch as an independent agent that begins at the root of the Latent Task Tree. Each agent independently descends the tree via hierarchical Thompson sampling. This induces a natural balance between exploration and exploitation, allowing different elements of the batch to explore distinct regions of the task space.

At a node with children $c \in \mathcal{C}$, the agent samples a reward estimate for each child:

$$\tilde{y}_c \sim \mathcal{N}(\mu_c, \sigma_c^2), \quad \forall c \in \mathcal{C}, \tag{6}$$

and selects $c^* = \arg\max_{c \in \mathcal{C}} \tilde{y}_c$. This procedure is applied recursively, guiding the agent from coarse, high-level regions of the tree toward increasingly fine-grained subsets of problems. The descent terminates at a leaf node, corresponding to an individual problem (prompt), which is then selected for inclusion in the batch.

This approach differs from prior bandit-based curricula (Wang et al., 2025b; Chen et al., 2025; Do et al., 2025) that select distributions of tasks for the *entire batch*. While such strategies can be effective, they couple all samples in a batch to the same region of the task space. In contrast, treating each batch element as an independent agent allows a single batch to span multiple regions of the latent manifold, promoting diversity in the resulting learning signals.

At the same time, the hierarchical structure enables focused sampling within regions estimated to be more productive. This interplay between *focus* (concentrating sampling within productive regions) and *diversity* (spreading across equally productive regions) allows BMC to adaptively allocate training effort, yielding gradients that are informative across a broader range of tasks and potentially improving generalization and learning efficiency.

M.2.2 Extended Description of Non-Stationary Belief Modeling (Bayesian Filtering)

Each leaf of the Latent Task Tree maintains a probabilistic belief over its learning signal, modeled as a Gaussian distribution $y_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$, where μ_i denotes the expected learning signal and σ_i^2 captures uncertainty. Unlike classical stationary settings, the learning signal associated with a problem can evolve as the model acquires new capabilities, requiring an update rule that both incorporates new evidence and discounts outdated beliefs.

Our update mechanism is inspired by Bayesian filtering for a Gaussian with unknown mean and variance, as captured by the Normal–Inverse–Gamma (NIG) conjugate prior. Rather than maintaining the full conjugate posterior, we track only the implied mean, variance, and an effective prior weight λ_i, which serves as a soft measure of accumulated evidence. Crucially, λ_i adapts dynamically in response to estimation error (surprise), enabling non-stationary behavior.

Let $y_i^{(t)}$ denote the observed learning signal at step t. We define the normalized surprise as

$$s_i^{(t)} = \frac{y_i^{(t)} - \mu_i^{(t-1)}}{\sigma_i^{(t-1)}}. \tag{7}$$

This normalization interprets deviations relative to the model's confidence: unexpected outcomes for highly certain leaves induce stronger belief updates than comparable deviations for uncertain leaves.

Surprise modulates the influence of prior evidence through an effective prior weight

$$\lambda_i^{\text{eff}} = \lambda_i^{(t-1)} \cdot \exp\Big(-(s_i^{(t)})^2\Big), \tag{8}$$

so that highly surprising observations reduce reliance on outdated statistics. The posterior mean is updated as

$$\mu_i^{(t)} = \frac{\lambda_i^{\text{eff}} \cdot \mu_i^{(t-1)} + y_i^{(t)}}{\lambda_i^{\text{eff}} + 1}. \tag{9}$$

The variance update balances confidence contraction with uncertainty injection:

$$\sigma_i^{2\,(t)} = \sigma_i^{2\,(t-1)} \frac{\lambda_i^{\text{eff}}}{\lambda_i^{\text{eff}} + 1} + \frac{\log\Big(1 + (s_i^{(t)})^2 + \text{staleness}_i\Big)}{\lambda_i^{\text{eff}} + 1}. \tag{10}$$

Intuitively, beliefs become more confident when observations align with expectations and are recent, while surprising outcomes or prolonged lack of feedback increase uncertainty. This adaptive mechanism allows the curriculum to remain responsive to model drift, revisit previously explored problems when necessary, and avoid premature convergence to stale regions of the task manifold.

Further intuition and justification for these update rules are provided in Appendix M.5

M.2.3 Extended Description of Bottom-Up Belief Propogation (Empirical Bayes)

We perform bottom-up belief propagation using an *empirical Bayes* approach, estimating the parameters of higher-level nodes directly from the posterior statistics of their children. After updating leaf-level beliefs, information is propagated upward through the Latent Task Tree to maintain consistent beliefs at all levels of the hierarchy. While only leaf nodes correspond to individual problems, higher-level nodes provide aggregated summaries that guide top-down problem selection by identifying productive regions of the task manifold. Crucially, this propagation induces an endogenous coupling between related tasks: observations at individual prompts immediately influence the beliefs of their ancestors, thereby affecting the sampling probabilities of other tasks within the same region of the tree.

For each non-leaf node, we aggregate the beliefs of its children using a precision-weighted estimator. Let $\{(\mu_i, \sigma_i^2)\}_{i=1}^K$ denote the posterior mean and variance of the K child nodes in a subtree. Intuitively, children with lower uncertainty should exert greater influence on the parent belief, while highly uncertain children contribute less.

However, child nodes may disagree substantially in their estimated learning signal, reflecting genuine heterogeneity within a region of the task manifold. To account for this, we introduce a subtree-level heterogeneity term (DerSimonian & Laird, 1986),

$$\tau_p = \max\big(0, \mathrm{Var}(\mu_i) - \mathbb{E}[\sigma_i^2]\big)\,, \tag{11}$$

which measures the extent to which variation between child means exceeds what would be expected from their internal uncertainty alone. When children agree, τ_p is near zero; when they disagree, τ_p inflates parent uncertainty.

Using this correction, we define precision weights

$$w_i = \frac{1}{\sigma_i^2 + \tau_p}, \tag{12}$$

and compute the parent belief as

$$\mu_p = \frac{\sum_i w_i \mu_i}{\sum_i w_i}, \qquad \sigma_p^2 = \frac{1}{\sum_i w_i}. \tag{13}$$

This aggregation procedure can be viewed as a random-effects hierarchical estimator that balances confidence in individual children with observed disagreement across the subtree. Rather than assuming stationarity or independence, it explicitly increases uncertainty in regions where learning signals are inconsistent, preventing overconfident selection during top-down sampling.

Belief propagation proceeds recursively from the leaves to the root after each batch update, allowing evidence collected at the level of individual problems to inform higher-level task representations. This enables coordinated exploration and exploitation across the hierarchy, where uncertainty and disagreement at lower levels appropriately influence decisions over broader regions of the task space.

M.3 Logit-Normal Beliefs and the Sigma Rule

The main text presents BMC's prompt-level belief model as Gaussian for clarity. In the implementation, however, the normalized learning signal satisfies $y_i \in [0, 1]$. An *unconstrained* Gaussian belief directly over this bounded signal can assign probability mass to impossible values, which makes uncertainty difficult to interpret when beliefs are propagated through the Latent Task Tree. A Beta distribution would naturally respect the interval $[0, 1]$, but is most directly suited to Bernoulli-style success probabilities, whereas our learning signal may be a continuous bounded quantity such as normalized reward variance or expected absolute advantage. We therefore use a Logit-Normal parameterization: BMC maintains Gaussian beliefs in logit space, where posterior updates, hierarchical

Thompson sampling, and empirical-Bayes aggregation remain convenient. Under the inverse sigmoid map, this Gaussian belief corresponds to a valid bounded belief over the original learning-signal scale.

Before applying the logit transform, we clamp the observed signal to avoid infinite values:

$$\tilde{y}_i = \mathrm{clip}(y_i, \epsilon, 1-\epsilon).$$

We then transform the signal as

$$r_i = \mathrm{logit}(\tilde{y}_i) = \log \frac{\tilde{y}_i}{1-\tilde{y}_i}.$$

BMC maintains a Gaussian belief over this transformed signal,

$$r_i \sim \mathcal{N}(\mu_i, \sigma_i^2).$$

Equivalently, the inverse sigmoid map induces a logit-normal belief over the original bounded learning signal:

$$y_i = \sigma(r_i) = \frac{1}{1+\exp(-r_i)}.$$

In practice, BMC performs belief updates, aggregation, and sampling directly in logit space. The inverse map is useful for interpretation and intuition: it shows that the underlying belief corresponds to valid learning-signal values in $[0, 1]$.

Importantly, because the sigmoid and logit transforms are monotone, operating in logit space does not change the ordering of point estimates used for comparative Thompson sampling. Thus, if one prompt or distribution has a larger estimated learning signal than another in the original space, it also has a larger transformed estimate in logit space. The purpose of the transform is therefore not to change what counts as productive, but to place uncertainty modeling and empirical-Bayes aggregation in a space where Gaussian beliefs are better aligned with the bounded geometry of the learning signal.

The clamping parameter ϵ is chosen based on the number of rollouts per prompt. In our implementation, with k rollouts, we set

$$\epsilon = \frac{1}{k^2}.$$

This avoids treating empirical signals of 0 or 1 as infinitely certain endpoints, which would be inappropriate under finite rollout estimates. The corresponding maximum logit magnitude is

$$r_{\max} = \left|\mathrm{logit}(1-\epsilon)\right|.$$

For example, with $k = 8$ rollouts, $\epsilon = 1/64$ and $r_{\max} = \log(63)$.

Let a denote the sigma-rule constant, set to $a = 3$ in our experiments.[2] We choose $\sigma_{\max}$ so that a standard deviations span the effective logit range:

$$a\sigma_{\max} = r_{\max}.$$

Thus,

$$\sigma_{\max}^2 = \left(\frac{r_{\max}}{a}\right)^2.$$

This prevents uncertainty from growing without bound while still allowing Thompson samples to cover the effective range of attainable learning signals. Conversely, we also impose a minimum variance to prevent beliefs from collapsing permanently, ensuring that prompts can be revisited if their learning signal changes as the policy evolves.

Overall, the logit-normal parameterization aligns the belief model with the bounded geometry of the learning signal. The sigma rule then provides a simple scale for uncertainty in the transformed space: large enough to represent substantial uncertainty over the full range of possible signals, but bounded enough to avoid unstable empirical-Bayes propagation or Thompson samples dominated by extreme logit-space values.

[2]This corresponds to the standard "three-sigma" heuristic for Gaussian distributions, where approximately 99.7% of the mass lies within three standard deviations of the mean; see the 68–95–99.7 rule.

M.4 Sampling Without Replacement

In LLM training, prompt batches are typically sampled without replacement so that the same prompt does not appear multiple times in a single optimization step. For prompt-level curriculum methods that follow the standard bandit pattern, this is straightforward: the scheduler can select the top-B prompts under the current acquisition score, or sample a batch of distinct prompt-level arms directly.

For BMC, without-replacement sampling is slightly more subtle because sampling decisions are made through hierarchical descent. Each element of the batch is selected by traversing the Latent Task Tree, rather than by independently scoring all prompt-level arms. A naive implementation is to let each batch element descend the tree independently, check for duplicate prompts after the batch is formed, and resample any duplicates from the root until all prompts are unique. We refer to this as the rejection-based implementation.

Although this approach is simple and often sufficient, it has two drawbacks. First, resampled prompts are selected using a tree whose beliefs still include prompts that have already been assigned to the current batch. Thus, the posterior quantities used during descent are not fully conditioned on the remaining available prompts. Second, when the sampler becomes highly concentrated, rejection sampling can lead to long duplicate-resolution loops, increasing wall-clock time. We observed this issue most clearly in BMC-T, where utility-aware sampling can concentrate probability mass on a small set of relevant regions.

To avoid this issue, we use a batch-aware variant of hierarchical descent. Batch elements are selected sequentially. After a prompt is selected, it is marked as unavailable for the remainder of the batch, and the tree statistics used for subsequent descents are temporarily recomputed to exclude unavailable prompts. Thus, each batch element descends a tree conditioned on the prompts already selected in the current batch. This preserves without-replacement sampling while avoiding repeated rejection loops.

Batch-aware descent also has a useful exploration effect. As high-value prompts in a region are exhausted within the current batch, subsequent descents are encouraged to select other available prompts or neighboring regions, rather than repeatedly attempting to select the same prompt. In this sense, batch awareness makes the within-batch sampling process better aligned with the hierarchical structure used by BMC.

Unless otherwise stated, we treat batch-aware descent as the default implementation of BMC. The main BMC experiments used the rejection-based implementation because duplicate conflicts were rare and the simpler procedure was sufficient. The two implementations differ only in how without-replacement constraints are enforced within a batch; the underlying posterior updates, tree structure, and acquisition rules are unchanged.

M.5 Saturating Estimation Errors

The non-stationary belief update functions mentioned in the main section are intended to make BMC responsive to non-stationarity without making belief updates overly reactive to individual noisy observations. Intuitively, we want the scheduler to revise its beliefs when observations contradict prior expectations, but not to "panic" in response to a single surprising estimation error.

The squared surprise term:

$$\lambda_i^{\text{eff}} = \lambda_i^{(t-1)} \cdot \exp\Big(-(s_i^{(t)})^2\Big)$$

provides a simple symmetric measure of mismatch. Since both unexpectedly high and unexpectedly low learning signals indicate that the current belief may be stale, the update should depend on the magnitude of the deviation rather than its sign. Squaring the normalized surprise accomplishes this while increasing sensitivity to larger deviations.

For the belief (posterior mean) update:

$$\mu_i^{(t)} = \frac{\lambda_i^{\text{eff}} \cdot \mu_i^{(t-1)} + y_i^{(t)}}{\lambda_i^{\text{eff}} + 1}$$

the exponential decay applied to the effective prior weight implements soft forgetting. For small surprises, the decay is mild, so accumulated evidence is mostly preserved. For large surprises,

the effective prior weight decreases rapidly, allowing the belief to adapt to potential model drift. This gives the update a controlled reset-like behavior without requiring a hard threshold or discrete change-point detector.

For the uncertainty (posterior variance) update:

$$\sigma_i^{2\,(t)} = \underbrace{\sigma_i^{2\,(t-1)} \frac{\lambda_i^{\text{eff}}}{\lambda_i^{\text{eff}} + 1}}_{\substack{\text{confidence} \\ \text{contraction}}} + \underbrace{\frac{\log\Big(1 + (s_i^{(t)})^2 + \text{staleness}_i\Big)}{\lambda_i^{\text{eff}} + 1}}_{\substack{\text{uncertainty} \\ \text{injection}}}$$

the logarithmic uncertainty term serves the complementary purpose of injecting uncertainty without allowing it to grow explosively. Large surprises and stale observations should increase uncertainty, since they suggest the current estimate may no longer be reliable. However, a linear uncertainty penalty could over-amplify noisy measurements and destabilize sampling. The logarithm provides diminishing growth: uncertainty increases when mismatch is detected, but the increase saturates for very large deviations.

Together, the exponential and logarithmic terms implement a conservative adaptation rule: surprising observations can reduce confidence in old evidence and encourage re-exploration, but their effect is bounded and smooth. These choices are practical heuristics rather than claims of Bayesian optimality. Alternative saturating functions could be used, but we found this parameterization to be simple, coefficient-light, and effective in our structured bandit setting.

M.6 Staleness & Uncertainty Growth

In addition to local surprise from newly observed rewards, BMC increases uncertainty for prompts that have not been sampled recently. This mechanism is intended to prevent uncertainty estimates from collapsing permanently for prompts whose learning signal may have changed due to policy drift.

Let B denote the batch size and N denote the number of available arms. We define the batch coverage coefficient as

$$c = \frac{B}{N}.$$

This coefficient measures the fraction of arms that can be sampled in a single batch. If c is small, then many arms will naturally remain unsampled for long periods of time, so staleness should accumulate more slowly. Conversely, if c is large, then failing to sample an arm for several steps is stronger evidence that its belief may be stale relative to the current policy.

For each arm i, we maintain an age counter a_i measuring the number of scheduler steps since the arm was last sampled. We define the coverage-adjusted staleness term as

$$\text{staleness}_i = c \cdot a_i.$$

The resulting uncertainty injection is then computed as

$$\Delta\sigma_i^2 = \frac{\log\Big(1 + (s_i^{(t)})^2 + \text{staleness}_i\Big)}{\lambda_i^{\text{new}}},$$

where λ_i^{new} denotes the updated effective evidence weight.

This design makes staleness contribute to uncertainty growth while accounting for the sampling budget. The coverage factor prevents uncertainty from saturating too quickly in settings with many arms and small batches, where long gaps between observations are expected. The logarithmic transformation further ensures diminishing growth, so uncertainty increases for stale arms without becoming dominated by extremely large staleness values.

M.7 Tree Diagnostics

To assess differences between sampling strategies (specifically, strategies that rely on non-uniform sampling), we devise two diagnostics: *rarity-weighted coverage* and *structure gain*.

M.7.1 RARITY-WEIGHTED EXPOSURE

To evaluate whether a scheduler explores underrepresented regions of the latent task tree, we compute a rarity-weighted exposure metric over penultimate clusters. Let $\mathcal{C} = \{1, \dots, K\}$ denote the set of penultimate clusters, and let n_c^{data} be the number of training prompts assigned to cluster c. The empirical dataset prior over clusters is

$$p_{\text{data}}(c) = \frac{n_c^{\text{data}}}{\sum_{c'=1}^{K} n_{c'}^{\text{data}}}.$$

We assign each cluster a rank-based rarity weight. Let $\text{rank}(c)$ be the rank of cluster c when clusters are sorted in ascending order by $p_{\text{data}}(c)$, so that the rarest cluster has rank 0 and the most common cluster has rank $K-1$. We define

$$w_c = 1 - \frac{\text{rank}(c)}{K-1},$$

for $K > 1$, and $w_c = 1$ when $K = 1$. Thus, rarer clusters receive larger weights, while the most common clusters receive smaller weights. Because the weights depend only on rank rather than raw inverse frequency, the metric is less sensitive to outlier cluster sizes.

At training step t, let $n_c^{(t)}$ denote the number of selected prompts in the batch that belong to cluster c. The scheduler-induced exposure distribution over clusters is

$$p_t(c) = \frac{n_c^{(t)}}{\sum_{c'=1}^{K} n_{c'}^{(t)}}.$$

The rarity-weighted exposure at step t is then

$$R_t = \sum_{c=1}^{K} p_t(c) w_c.$$

This metric is high when the scheduler allocates more batch mass to clusters that are rare under the dataset distribution.

For visualization, we normalize this quantity relative to the expected rarity exposure under the dataset prior. Let

$$R_{\text{data}} = \sum_{c=1}^{K} p_{\text{data}}(c) w_c, \qquad R_{\max} = \max_c w_c.$$

We report the normalized rarity-weighted exposure

$$\widetilde{R}_t = \text{clip}\left(\frac{R_t - R_{\text{data}}}{R_{\max} - R_{\text{data}}}, 0, 1 \right).$$

Thus, $\widetilde{R}_t = 0$ corresponds to exposure no greater than what would be expected under the dataset prior, while larger values indicate disproportionate exposure to rarer clusters.

We emphasize that rarity-weighted exposure is a diagnostic metric, not an objective that BMC directly optimizes. Directly maximizing this quantity would be degenerate: a scheduler could obtain a high score by allocating all samples to the rarest penultimate cluster, regardless of whether those prompts provide useful learning signal. Instead, we use rarity-weighted exposure to characterize sampling behavior. In particular, it helps distinguish BMC from unstructured Thompson Sampling by showing whether the scheduler can prioritize high-learning-signal prompts while still allocating exposure to underrepresented regions of the task space.

M.7.2 STRUCTURE GAIN

To quantify whether the latent task tree captures meaningful structure in the problem space, we measure how much variation in a prompt-level statistic (belief about learning signal) is explained by the penultimate clusters. Let $y_i^{(t)}$ denote a prompt-level scalar at training step t, such as the current estimated learning signal for prompt i. Let $c_i \in \{1, \dots, K\}$ denote the penultimate cluster assignment for prompt i.

We compute the fraction of variance in $y^{(t)}$ explained by the cluster partition:

$$R^2_{\text{tree}}(t) = 1 - \frac{\sum_{k=1}^{K} \sum_{i:c_i=k} \left(y_i^{(t)} - \bar{y}_k^{(t)}\right)^2}{\sum_{i=1}^{N} \left(y_i^{(t)} - \bar{y}^{(t)}\right)^2},$$

where

$$\bar{y}_k^{(t)} = \frac{1}{|\{i : c_i = k\}|} \sum_{i:c_i=k} y_i^{(t)}$$

is the mean value within cluster k, and $\bar{y}^{(t)}$ is the global mean across prompts. This quantity measures the proportion of total variance explained by the tree partition. Higher values indicate that prompts within the same penultimate cluster have more similar learning-signal estimates than prompts in different clusters.

Because larger or more imbalanced clusters can affect explained variance, we compare the tree partition against a size-matched random baseline. Specifically, we generate random partitions by permuting the cluster labels across prompts, preserving the exact cluster-size distribution while destroying the relationship between prompts and clusters. For each random partition m, we compute the same explained variance score $R^2_{\text{rand},m}(t)$.

We define the structure gain as the ratio between the explained variance of the true tree partition and the average explained variance of these size-matched random partitions:

$$G_{\text{struct}}(t) = \frac{R^2_{\text{tree}}(t)}{\frac{1}{M}\sum_{m=1}^{M} R^2_{\text{rand},m}(t)}.$$

Values greater than one indicate that the latent task tree explains more variation in the prompt-level statistic than expected from a random partition with the same cluster sizes. Because the denominator can be small, the ratio should be interpreted as a relative alignment diagnostic rather than as a claim that the tree fully explains the learning signal. In particular, a large structure gain indicates that the tree partition captures substantially more learning-signal variation than size-matched random groupings. In the 8B setting, the true partition explains an average of $R^2 = 0.00875$ of prompt-level learning-signal variation, indicating that the effect is modest in absolute terms but substantially enriched relative to the random baseline.

M.8 BMC-T

M.8.1 Implementation

BMC-T extends BMC by adding a target-utility term to hierarchical sampling. Let $\mathcal{D}_{\text{train}} = \{x_i\}_{i=1}^{N}$ denote the sampleable training problems, and let $\mathcal{D}_{\text{target}} = \{z_j\}_{j=1}^{M}$ denote an arbitrary target distribution, such as a held-out evaluation set, validation set, or collection of problems representing a desired capability. Target examples are used only to define utilities over regions of the Latent Task Tree; they are never included in the set of arms available to the bandit scheduler. Thus, BMC-T may use target examples to estimate which latent regions are relevant, but it can only sample problems from $\mathcal{D}_{\text{train}}$.

We construct a Latent Task Tree using both $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{target}}$. Let $\mathcal{T}$ denote the resulting tree. The leaves of $\mathcal{T}$ correspond to individual examples, while internal nodes correspond to latent task regions. For a node $v \in \mathcal{T}$, let $\text{child}(v)$ denote its children and let $\mathcal{L}(v)$ denote the set of examples contained in the subtree rooted at v.

For each node v, we compute

$$n_{\text{train}}(v) = \sum_{x \in \mathcal{L}(v)} \mathbf{1}\{x \in \mathcal{D}_{\text{train}}\},$$

$$n_{\text{target}}(v) = \sum_{x \in \mathcal{L}(v)} \mathbf{1}\{x \in \mathcal{D}_{\text{target}}\},$$

$$w_{\text{target}}(v) = \sum_{x \in \mathcal{L}(v)} \omega(x)\mathbf{1}\{x \in \mathcal{D}_{\text{target}}\},$$

where $\omega(x) \geq 0$ is an optional target-example weight. In our experiments, these weights can be used to normalize across target sources or to focus on a particular target subset. If no weighting is needed, we use uniform target weights.

A node is sampleable only if it contains at least one training example:

$$n_{\text{train}}(v) > 0.$$

Nodes with $n_{\text{train}}(v) = 0$ may still contain target examples, but they are excluded from the scheduler because there are no training problems available to sample from that region.

For each internal node v, utilities are defined locally over its sibling set $\text{child}(v)$. Let

$$N_{\text{train}}(v) = \sum_{c \in \text{child}(v)} n_{\text{train}}(c),$$
$$N_{\text{target}}(v) = \sum_{c \in \text{child}(v)} n_{\text{target}}(c),$$
$$W_{\text{target}}(v) = \sum_{c \in \text{child}(v)} w_{\text{target}}(c).$$

If $N_{\text{target}}(v) = 0$ or $W_{\text{target}}(v) = 0$, then the children of v receive zero utility:

$$u(c) = 0 \quad \forall c \in \text{child}(v).$$

Otherwise, we estimate the local target mass assigned to each child. Let $K_v = |\text{child}(v)|$ and define

$$\alpha_v = \rho \frac{W_{\text{target}}(v)}{K_v},$$

where $\rho > 0$ is a smoothing coefficient. The child-level target mass is

$$p_{\text{target}}(c \mid v) = \frac{w_{\text{target}}(c) + \alpha_v}{W_{\text{target}}(v) + K_v \alpha_v}.$$

Continuing the Bayesian interpretation used throughout BMC, this additive smoothing can be viewed as placing a symmetric Dirichlet prior with concentration α_v on the categorical distribution over children. Intuitively, $p_{\text{target}}(c \mid v)$ is not meant to represent a true data-generating probability; it is a normalized local estimate of how the observed target mass under parent v is distributed across its children. This makes target relevance *comparable* within each sibling set during top-down descent, while smoothing prevents finite target samples from assigning overly brittle zero-utility estimates to nearby regions.

We additionally apply a train-support gate to avoid assigning large utility to regions that contain target examples but little sampleable training support. Let

$$\lambda_v = \eta N_{\text{train}}(v),$$

where η is the same HDBSCAN minimum-cluster fraction hyperparameter used in constructing the Latent Task Tree (see Appendix L.1). We define

$$g(c \mid v) = \frac{n_{\text{train}}(c)}{n_{\text{train}}(c) + \lambda_v}.$$

Intuitively, this gate downweights children whose target overlap is not supported by enough sampleable training examples. Using the same minimum-cluster fraction η ties this notion of support to the scale used when constructing the Latent Task Tree: a child receives high utility only when it contains a nontrivial amount of trainable mass relative to the local parent region.

The final utility assigned to child c is then

$$u(c) = p_{\text{target}}(c \mid v) \cdot g(c \mid v).$$

During hierarchical Thompson Sampling, standard BMC samples a productivity score for each child using the Gaussian belief maintained at that node. For child c of parent v, the standard BMC score is

$$s_{\text{BMC}}(c) \sim \mathcal{N}\left(\mu_c,\, \sigma_c^2 + \tau_v\right),$$

where μ_c and σ_c^2 are the child belief parameters, and τ_v is the parent-level heterogeneity term used by BMC.

BMC-T modifies this score by adding the target-utility bonus. Since utilities are computed locally within each sibling set, we normalize by the maximum child utility under the current parent:

$$\bar{u}(c) = \frac{u(c)}{\max_{c' \in \text{child}(v)} u(c') + \epsilon},$$

where ϵ is a small constant for numerical stability. The utility-aware score is

$$s_{\text{BMC-T}}(c) = s_{\text{BMC}}(c) + \gamma r_{\max} \bar{u}(c).$$

Here, $\gamma \geq 0$ controls the strength of utility awareness, and $r_{\max}$ places the utility bonus on the same reward scale as the productivity model. Setting $\gamma = 0$ removes the utility bonus, but does not necessarily recover standard BMC because target examples may still affect the topology of the Latent Task Tree. Larger values of γ increasingly steer the scheduler toward regions with high target overlap.

At each internal node, BMC-T selects the child with the largest utility-aware score:

$$c^\star = \arg \max_{c \in \text{child}(v)} s_{\text{BMC-T}}(c),$$

subject to the constraint that c has at least one available training example in its subtree. The sampler then descends recursively until reaching a sampleable training problem.

M.8.2 Alternative Implementations

The BMC-T implementation above is one concrete way to incorporate utility into hierarchical curriculum learning. It estimates target relevance from overlap between training and target examples within the Latent Task Tree, then adds a normalized utility bonus to the productivity score used during top-down sampling. This design keeps the roles of productivity and utility separated: productivity estimates where the current policy can learn, while utility estimates which learnable regions are relevant to a target distribution.

However, the broader idea is not tied to this particular utility estimator or additive scoring rule. Utility could instead enter multiplicatively, modulating the productivity score so that target relevance acts as a gate rather than a bonus. Utility could also be estimated from other signals, such as gradient-based alignment (Yang et al., 2026) or optimizer-based data valuation (Wang et al., 2026). These alternatives may be useful when target examples are sparse, unavailable, or difficult to combine directly with the training set in a shared tree.

Thus, BMC-T should be interpreted as a proof of concept for utility-aware curriculum design rather than a unique prescription. Its purpose is to show that evaluation relevance can be explicitly represented alongside productivity and diversity, and that curriculum decisions can be made over target-relevant regions of the task manifold rather than over difficulty-based metrics alone.