What LLMs Think When You Don't Tell Them What to Think About?

Anonymous ACL submission

Abstract

Characterizing the behavior of large language models (LLMs) across diverse settings is critical for reliable monitoring and AI safety. However, most existing analyses rely on topic- or task-specific prompts, which can substantially limit what can be observed. This work studies what LLMs generate from minimal and topic-neutral inputs, probing their near-unconstrained generative behavior. Despite the absence of explicit topics, model outputs cover a broad semantic space, and surprisingly, each model family exhibits strong and systematic topical preferences. GPT-OSS predominantly generates programming (27.1%) and mathematical content (24.6%), whereas Llama most frequently generates literary content (9.1%). DeepSeek often generates religious content, while Qwen frequently generates multiple-choice questions. Beyond topical preferences, we also observe differences in content specialization and depth: GPT-OSS often generates more technically advanced content (e.g., dynamic programming) compared with other models (e.g., basic Python). Furthermore, we find that the near-unconstrained generation often degenerates into repetitive phrases, revealing intriguing behaviors unique to each model family. For instance, degenerate outputs from Llama include multiple URLs pointing to personal Facebook and Instagram accounts. Our method can provide valuable and practical insights for monitoring model behavior in unexpected or less-controlled environments and for identifying potential private information leakage. We release the complete dataset of 256,000 samples from 16 LLMs, along with a reproducible codebase.

1 Introduction

Analyzing the behavior of large language models (LLMs) across diverse inputs and prompting conditions has become increasingly important for the safe and reliable deployment of AI systems (Liu et al., 2023; Röttger et al., 2025). Behavioral auditing enables model developers to identify risks that are challenging to capture with standard benchmarks, such as the unintended generation of harmful content, biased outputs, or private information (Albuquerque et al., 2024). At the same time, it helps end users make more informed decisions about which models to use. Accordingly, numerous studies have systematically investigated LLM behaviors and biases (Tamkin et al., 2024; Hu et al., 2025). For instance, prior studies have examined inter-model homogeneity in open-ended settings (Jiang et al., 2025) and biased advice associated with names commonly linked to racial minorities and women (Salinas et al., 2024).

However, much of the existing literature relies on narrowly scoped, topic- or task-specific prompts. While effective for probing specific targeted behaviors, this approach explicitly constrains LLM outputs, limiting our ability to observe their behavior in minimally constrained settings (Hida et al., 2024; Yun et al., 2025). Studying minimally constrained settings is essential for understanding model behaviors in less-controlled environments, and for establishing safe usage guidelines, as such settings can reveal patterns that remain latent under structured prompts (Carlini et al., 2021; Hoscilowicz et al., 2024). This importance is even greater for open-source models, as their virtually limitless flexibility enables a wide range of experimentation across diverse generation scenarios, thereby necessitating deeper understanding and stricter safety guidelines. In particular, our findings demonstrate that open-source models can, through unconstrained behaviors, expose personal information (Figure 4), highlighting the need for comprehensive monitoring, evaluation, and risk assessment throughout the model development lifecycle.

Motivated by these perspectives, we study open-source models' behavior under minimal, topic-neutral prompts, where external constraints are

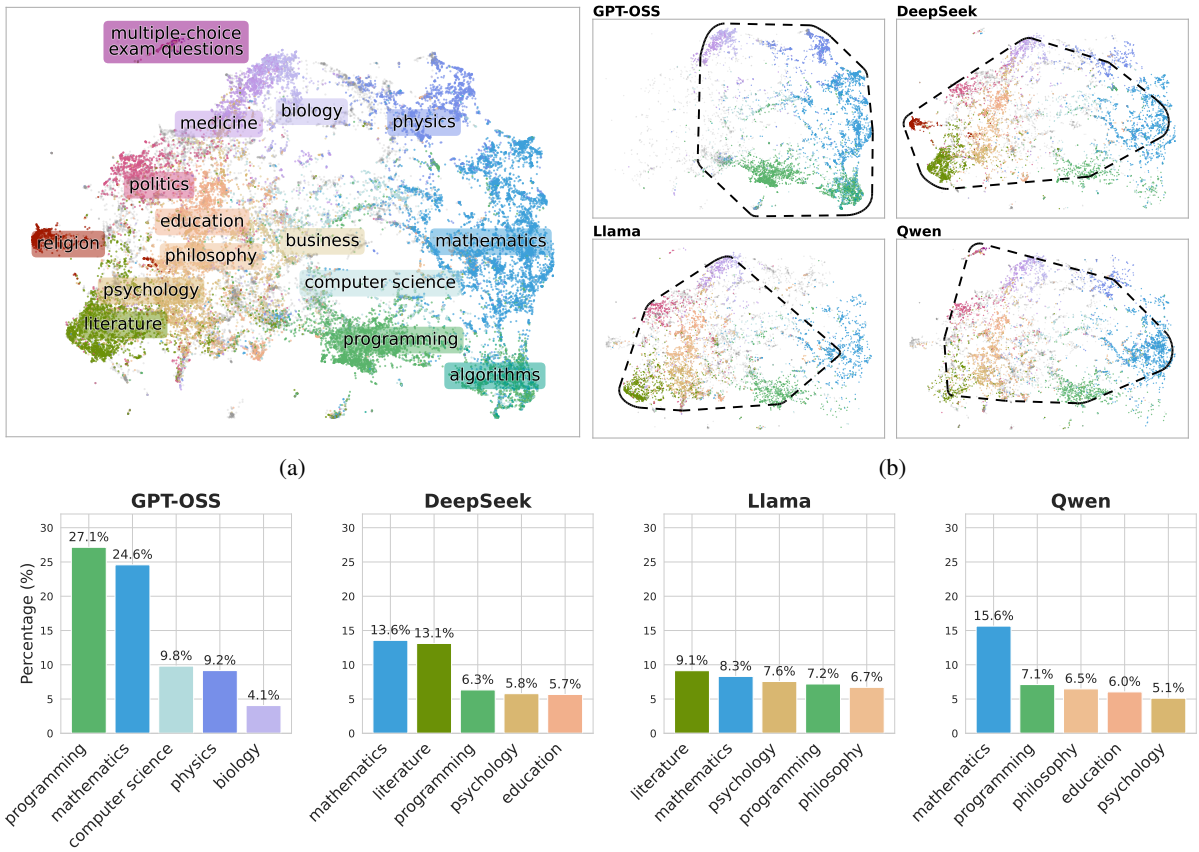


Figure 1: **LLM's top-of-mind behaviors.** UMAP visualizations of model outputs generated by the GPT-OSS, DeepSeek, Llama, and Qwen model families. Figure (a) visualizes outputs from all four model families, with each dot representing a generated output and class labels positioned at the centroid of their clusters. Except for 'multiple-choice exam questions' and 'algorithms,' cluster labels correspond to the 13 most frequent categories in our dataset. The two labels are included to better highlight clusters that are particularly prominent in Qwen and GPT-OSS. Figure (b) illustrates an individual model family, with the black dotted line indicating the convex hull of the high-density region. Figure (c) shows the top five categories within each model family and their corresponding percentages. The color scheme is consistent across all figures. **Despite the use of topic-neutral seed prompts, the semantic distributions of model outputs show a broad and diverse range of topics, and each model family exhibits distinctive distributional patterns.** Methodological details are provided in Section 2, and a high-quality interactive figure with fine-grained labels is available in Supplementary Material.

reduced as much as possible; we refer to this as the model's *top-of-mind* behavior. By minimizing the use of prompts and fixed chat templates that are typically applied by default, we focus on a more fundamental question: what types of content do models naturally generate when unconstrained, how systematically do these tendencies vary across model families, and more broadly, what insights can we gain from unconstrained settings? This is the central scientific question of this paper.

Main Contributions In this paper, we systematically explore what LLMs generate under minimally constrained settings across sixteen modern open-source LLMs. Through large-scale generation, we find that model outputs span a broad semantic space, even in the absence of explicit topics (Figure 1a). Interestingly, each model family exhibits distinctive and systematic topical preferences (Figure 1b). GPT-OSS mainly generates programming and math content, whereas Llama most frequently produces literary text. DeepSeek frequently produces religious content, while Qwen often generates multiple-choice questions. Our analysis also reveals substantial differences in subcategory specialization (Figure 2) and the technical depth across model families (Figure 3). For instance, GPT-OSS produces more technically advanced content than the other models do. In addition, we treat degenerate text as a behavioral signal and find that its characteristic patterns vary significantly across

model families. In particular, GPT-OSS frequently produces formatting-related phrases (e.g., a code block "\n\n```\n\n```"), while Qwen often generates conversational artifacts (e.g., "let me know" or "thank") as well as Chinese text. Llama generates accessible URLs referencing personal social accounts (Figure 4).

Together, our findings show that minimally constrained generation not only reveals underlying model tendencies, but also facilitates the identification and mitigation of private information leakage, highlighting its practical value for monitoring model behavior. We emphasize that our method is not intended to represent or optimize for typical end-user interaction patterns, as removing chat templates is typically infeasible in closed AI services. Rather, it serves as a systematic probe of a model's latent generative priors in less controlled settings, while also providing a useful diagnostic framework for auditing potential information leakage, developing safeguards, and understanding model behavior, especially for open-source model developers and researchers. We release the complete dataset of 256,000 samples from sixteen modern LLMs[1], along with a reproducible codebase and an interactive figure.

2 Method for Probing LLMs' Top-of-Mind Behaviors

Our method consists of three stages: (i) text generation, (ii) removal of degenerate text, and (iii) semantic labeling and embedding extraction for downstream data analysis.

Text Generation Our generation setting has two key components: (C1) the use of *topic-neutral, open-ended* seed prompts and (C2) the absence of chat templates.

For (C1), we carefully design a set of 36 *topic-neutral, open-ended* seed prompts. Here, the term *topic-neutral* indicates that the prompts do not explicitly specify any particular subject matter, and the term *open-ended* indicates that the prompts are not targeted at a specific task and do not imply a unique correct answer, allowing models to continue freely. In addition, our prompt design emphasizes the following aspects: brevity to ensure minimal constraints, no named entities, no domain-specific content words, and diversity. Our prompt set covers six different styles—conversational softeners, chain of thoughts, declarative prompts, rhetorical inquiries, informative expository prompts, and punctuation-only prompts—with six seed prompts for each style. It allows us to systematically probe a wide range of generative behaviors and ensures broad coverage across different scenarios. A full list of seed prompts is provided in Appendix C.

For (C2), we do not use a chat template, which is commonly employed in closed AI chat systems to improve user experience. Instead, the model is given only a randomly sampled seed prompt. This design choice is not meant to reflect their absence in real-world deployment, but rather to avoid a strong confound when the goal is to explore and assess model behaviors in diverse generation settings. To support this choice, Appendix A.1 shows that fixed chat templates can significantly influence generation behavior, reducing output diversity and restricting the behaviors that can be observed.

With (C1) and (C2), we generate text autoregressively from a randomly selected seed prompt, continuing until either the maximum token limit is reached or an end-of-sequence token is sampled. For generation, we use standard hyperparameters: a temperature of 1.0 and a top-p of 0.9 throughout the paper.

Remark 2.1 (The impact of chat templates). What happens when our seed prompt (e.g., "Let's think step by step.") is used in conjunction with a standard chat template? In this setting, we observe that LLMs often attempt to clarify the user's intent with a short answer (e.g., "What would you like to think through step by step?"), which stems from standard post-training procedures designed to accurately understand user intent. This suggests that standard generation settings based on fixed chat templates can substantially shape what models generate, underscoring the need for new approaches and frameworks to uncover behaviors that remain hidden under conventional prompting.

Removal of degenerate text While most text generated in the previous stage is coherent and meaningful, we observe that the removal of chat templates can lead LLM outputs to contain degenerate text—a consecutive substring that exhibits repetitive and semantically less coherent patterns (Holtzman et al., 2019). To be more precise, we define a segment of model output as degenerate text if it meets all of the following criteria. First, it contains a consecutive sequence of characters, which

[1] Our resulting dataset exceeds 6 GB, and due to its large size, it is not included in Supplementary Material. We will make this dataset publicly available after the review process.

we refer to in this paper as *a repeated phrase*, that is at least 10 characters long. Second, this sequence is consecutively repeated at least five times. Third, the repeated portion constitutes at least 5% of the entire output. Because degenerate text generally contains little semantic information and, in our experiments, continues until it reaches the maximum number of tokens allowed, which accounts for more than 96% of cases, we remove it by truncating the output from its first occurrence.

Semantic labeling and embedding extraction In the last stage of our method, we annotate semantic labels of the cleaned text from the previous stage and extract its embeddings for downstream data analysis. For labeling, following the practice in the literature (Gilardi et al., 2023; Tan et al., 2024), we adopt LLM-based open-vocabulary annotation. To improve consistency, we use a structured prompting format and require the model to return standardized JSON outputs. We additionally perform post-processing to normalize semantically equivalent labels and remove malformed responses. Further details on the labeling prompts, output schema, and normalization pipeline are provided in Appendix C. For embedding extraction, we use `Qwen3-Embeddings-8B`, as it is among the most powerful open-source embedding models (Muennighoff et al., 2022; Enevoldsen et al., 2025).

We additionally conduct extensive sensitivity analyses to examine the robustness of our results to the choice of LLM labelers (Appendix A.2) and embedding models (Appendix A.3). Specifically, we perform manual inspection of sampled annotations to evaluate the reasonableness of LLM-generated labels and the variability across different LLM models (Appendix A.2). These analyses show a human–LLM agreement rate of 87% and agreement between LLMs can be 75%, indicating that our findings are not driven by any particular labeling model. Moreover, the main qualitative and quantitative patterns remain consistent across a wide range of experimental settings, providing strong evidence that our conclusions are robust.

3 Analysis of Model Behaviors under Topic-Neutral, Open-Ended Prompts

3.1 Experimental settings

We evaluate sixteen language models from four families, GPT-OSS, DeepSeek, Llama, and Qwen, for which the complete list of models is provided in Appendix B. Our selection covers a diverse range of model providers, reasoning styles (e.g., reasoning, hybrid, and standard instruct), and parameter scales from 3B to 671B parameters. We focus on open-source models to ensure that text generation is not influenced by any external system prompts, hidden instruction tuning, or post-processing potentially applied in API-based deployments (Chen et al., 2024; Gao et al., 2024). For each model, we initially generate 16,000 samples with a maximum length of 4,096 tokens, yielding a total of 256,000 samples. We exclude 9,682 samples (3.8% of the raw dataset) labeled as 'unknown' or 'failed' by the LLM-based annotator. This filtered dataset is used for all analyses in the paper. For visualization, we create a balanced subset by stratified sampling across model families, with 10,000 randomly selected samples per family.

3.2 Main findings

Broad and diverse topics even under topic-neutral prompts Figure 1a visualizes the semantic space of generated texts from all models (McInnes et al., 2018). Despite the use of topic-neutral seed prompts, the generated outputs exhibit broad semantic coverage. We identify 123 well-populated representative categories, each containing at least 50 samples, which together account for 98.6% of the entire dataset. These categories span a wide range of domains, including the liberal arts (e.g., literature, philosophy, and education), science and engineering (e.g., physics, mathematics, and programming), as well as areas such as law, finance, music, sports, cooking, agriculture, archaeology, military, and fashion. See Appendix A.8 for representative examples.

Topical discrepancies across model families Interestingly, we observe that each model family exhibits distinct generative behaviors (Figure 1b). In particular, as shown in Figure 1c, GPT-OSS mostly generates content related to programming (27.1%) and mathematics (24.6%), with these two categories alone accounting for over 50% of its output. This combined proportion is substantially larger than that of any other model family; Qwen accounts for 22.7% of the outputs, followed by DeepSeek at 19.9%. In contrast, Llama generates the least technical content, but it produces diverse outputs in liberal arts domains, including literature (9.1%), psychology (7.6%), and philosophy (6.7%). This concentration of GPT-OSS in scientific domains and Llama in liberal arts is further illustrated

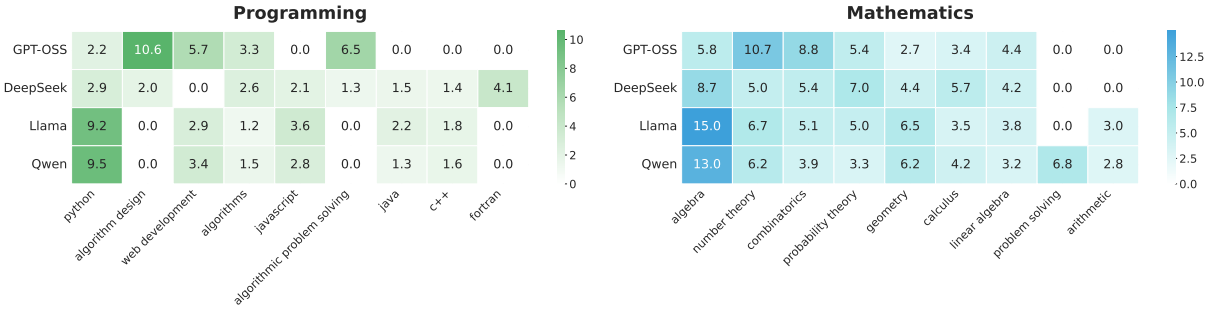


Figure 2: **Distribution of subcategories for programming and mathematics**. Numbers in each cell indicate the percentage of outputs that fall into a given subcategory among all outputs generated by the corresponding model family within a specific category. For each category, we select the nine subcategories with the highest average percentages and order them based on their size. **Each model family demonstrates distinct specialization across subcategories**.

by their high-density regions in Figure 1b, where kernel density estimation is used to estimate the embeddings' density values.

DeepSeek and Qwen exhibit more balanced semantic distributions than the other two model families. Still, as shown in Figure 1b, each exhibits distinct content preferences. DeepSeek generates religious content at a substantially higher rate (4.6%) than other model families: Qwen (2.6%), Llama (1.9%), and GPT-OSS (less than 0.05%). Similarly, Qwen forms a distinctive cluster dominated by multiple-choice exam questions, in which text often follows a structured format with a question and corresponding answer choices. We hypothesize that this prevalence reflects extensive use of multiple-choice exam data during Qwen's training. Our empirical results are shown to be consistent across different random splits of the prompts, supporting the robustness of our findings in Appendix A.6.

Remark 3.1 (Interpretation and implications). A natural question is *what drives the topic distributions observed* in this section. Given that the complexity of data mixtures, preprocessing pipelines, and training procedures of modern LLMs, providing a precise, rigorous characterization of the topic distributions is challenging. Nevertheless, we propose a plausible explanation for the dominant contributing factor: these results may primarily reflect pretraining data distributions optimized without any chat templates, rather than datasets structured with chat templates that impose stronger formatting and reasoning ability with supervised finetuning or reinforcement learning. See Appendix D. One direct implication is that the dominant prevalence of mathematics and programming datasets across different model families (Figure 1c) indicates substantial inclusion of mathematics- and programming-related content during pre-training, likely driven by their importance in benchmark evaluations and real-world AI applications (Cobbe et al., 2021; Jimenez et al., 2023).

Subcategory-level differences across model families We further conduct a fine-grained analysis at the subcategory level. Specifically, we select the top nine subcategories in programming and mathematics, and examine differences in subcategory preferences across model families. Additional analyses of ten categories, including literature, psychology, physics, and chemistry, are provided in Appendix A.4.

Figure 2 illustrates distributions of subcategories for programming and mathematics. In programming, GPT-OSS most frequently generates content related to 'algorithm design' and 'algorithmic problem solving,' while the other model families generate content in general programming languages: Llama and Qwen primarily focus on 'python,' and DeepSeek emphasizes 'fortran.' In addition, the distribution in Figure 2 includes many zero-valued entries, demonstrating that each model family specializes in distinct subcategories. In mathematics, GPT-OSS generates 'number theory' content, whereas the other three model families most frequently produce content related to 'algebra.' We observe that outputs categorized as 'algebra' frequently involve only basic arithmetic or polynomial operations, indicating that all models except GPT-OSS tend to generate relatively basic mathematical content.

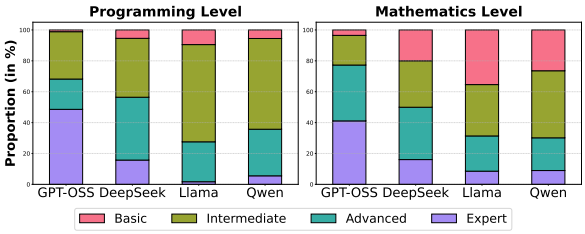


Figure 3: **Distribution of depth levels** in (left) programming and (right) mathematics. GPT-OSS mainly produces advanced or expert-level content in both programming and mathematics, while Llama and Qwen generate mostly basic or intermediate-level text.

Depth differences across model families Motivated by the preceding analysis, we analyze programming and mathematical text in terms of complexity. We focus on $n = 31,732$ mathematics and programming samples, assigning each to one of four difficulty tiers (namely, basic, intermediate, advanced, or expert) based on the detailed rubrics provided to `Claude-4.5-Opus`. These levels roughly correspond to elementary, middle, high school, and college-level content in mathematics, while in programming they cover a spectrum from basic concepts to competition- and interview-level questions. The exact prompt used in this experiment is provided in Appendix C.

As anticipated, GPT-OSS generates advanced and expert-level content more frequently than the other model families in both mathematics and programming (Figure 3). Specifically, 68.2% of GPT-OSS's programming outputs are classified as advanced or expert, primarily consisting of college-level algorithm questions (e.g., depth-first search, breadth-first search, or dynamic programming). In comparison, the corresponding percentages for DeepSeek, Qwen, and Llama are 56.4%, 35.7%, and 27.6%, respectively. These findings indicate that differences across model families are reflected not only in their subcategory specialization but also in the complexity of the content they produce.

Remark 3.2 (Topical preference and task competence). The topical preferences and output complexity analyzed in this section capture a model's knowledge space and its propensity to generate content related to particular domains. However, we find that these generative preferences do not reliably correspond to actual task competence. For example, GPT-OSS exhibits the lowest generation frequency in the law domain (less than 0.2%) among the model families, yet its performance on law benchmarks is not necessarily inferior to that of other models (Guha et al., 2023). We attribute this discrepancy to differences in generation settings, namely conditional versus near-unconditional generation. When prompted with explicit task context such as questions or instructions, model outputs are strongly guided by the prompt and the chat template, and they can express detailed, task-specific knowledge. In contrast, when prompted with topic-neutral inputs without chat templates, model outputs reflect their naive probabilistic distribution acquired during pre-training, which does not necessarily correspond to the model's actual competence (Nguyen, 2024).

4 Analysis of Degenerate Text

Prior work (Welleck et al., 2019; Holtzman et al., 2019) has shown that non-chat template settings frequently produce degenerate text, and we observe the same phenomenon in our generative settings. While degenerate text is typically discarded for low semantic coherence or repetitive content, we leverage it as a behavioral signal, analyzing how its structural and distributional patterns vary across model families. Below, we examine how these patterns vary structurally and distributionally across model families.

Artifacts in degenerate text We examine repeated phrases and find that degenerate text frequently exhibits conversational artifacts, question-answer artifacts, and Chinese text. For conversational artifacts, many models generate polite or helpful phrases commonly used in user interactions, such as "feel free to", "thank", "best regards", and "let me know". These four expressions appear in 27.8% of Qwen's degenerate samples, 26.9% for GPT-OSS, 14.6% for Llama, and only 0.6% for DeepSeek. Notably, although GPT-OSS and Qwen exhibit similar frequencies of such phrases, none of the GPT-OSS samples contain emojis, whereas Qwen frequently incorporates them (66.0% of degenerate text with the conversational artifacts). Figure 4 presents representative examples. For question-answer artifacts, we identify repeated phrases associated with answer formatting, such as "boxed", "final answer is", and "correct answer is". These patterns occur in 6.1% of Qwen's degenerate samples, 4.5% of Llama, 0.3% of DeepSeek, and less than 0.01% of GPT-OSS. For Chinese text, it is primarily observed in Qwen,

Model: GPT-OSS

\n\nYou're welcome! If there's anything else you need or want to discuss, just let me know. Have a fantastic day!\n\nThank you! I'm happy to help. If there's anything else you need or want to discuss, just let me know. Have a fantastic day!

Model: DeepSeek

[...] { **"from": "human"**, "value": "Let $f(x) = x^2 + 1$ and $g(x) = 2x - 1$. Find the value of $f(g(3))$." }, { **"from": "gpt"**, "value": "To find $f(g(3))$, first compute $g(3)$:\\n\\n$g(3) = 2(3) - 1 = 6 - 1 = 5$\\n\\nNow, plug this into $f(x)$:\\n\\n$f(g(3)) = f(5) = 5^2 + 1

Model: Llama

\nClick here to follow us on Facebook:\xa0 https://www.facebook.com/[redacted]/\nClick here to follow us on Twitter:\xa0 https://twitter.com/[redacted] 1\nClick here to follow us on Instagram:\xa0 https://www.instagram.com[redacted]/\nClick [...]

Model: Qwen

🌈💖\n\n如果有什么问题或者需要帮助，随时欢迎提问哦！😊💖✨\n\n希望你每天都能充满活力，保持积极乐观的态度！🌟🌈💖\n\n如果需要更多关于放松、减压的方法，可以回复我，我会分享更多实用技巧给你！💪🌟💕\n\n祝你今天充满活力，心情美好！🌞🌈✨💖🌞🌈✨💖🌞🌈✨💖[...]

Figure 4: **Examples of repeated phrases in degenerate text.** We observe that degenerate text often exhibits distinct qualitative patterns across model families, including conversational artifacts in GPT-OSS and Chinese text in Qwen. Degenerate text reveals sensitive or personal information. In the Llama example, we mask portions of the URLs, as they are accessible to personal Facebook and Instagram accounts at the time of inspection. English translation of the Qwen sample is provided in Appendix A.10.

representing 9.0% of its degenerate samples, compared to 1.5% for DeepSeek and less than 0.9% for the other models. Taken together, these patterns highlight systematic distributional differences across model families.

A qualitative analysis of degenerate text Beyond model-family discrepancies, our analysis also surfaces several noteworthy qualitative phenomena, illustrated by the examples in Figure 4. A sample from DeepSeek follows a structured JSON format, which is also known as ShareGPT format, with two entries. The first entry, labeled "from": "human", contains a mathematics question, while the second entry, labeled "from": "gpt", provides the corresponding solution. The presence of the second entry admits two possible explanations: (i) if the example was used during training, DeepSeek may be memorizing and reproducing it during generation. In this case, the example would constitute training data extraction from degenerate text, as studied by Carlini et al. (2021). (ii) If it was not used during training, the model would instead be fabricating a plausible output without grounding in an actual source—a behavior that could be concerning, as it produces outputs that appear to imply a specific provenance (e.g., ChatGPT).

Another example relates to the leakage of personal information. We find that degenerate text often includes specific URLs to personal Facebook, Instagram, or Twitter accounts. Specifically, using regular expression techniques to identify social accounts or email addresses, we estimate that Llama produces 0.78% and Qwen 0.35% of such instances. Fortunately, many of the generated links turned out to be invalid or hallucinated, but one sample generated by the Llama model included both Facebook and Instagram links that are actually accessible at the time of the investigation in January 2026. These examples show that monitoring degenerate text is crucial for model auditing, providing insights that are difficult to gain through conventional benchmark evaluations. We provide additional qualitative examples in Appendix E.

Remark 4.1. Our results demonstrate that monitoring degenerate text offers many practical insights on identifying training artifacts (e.g., conversational artifacts or question-answer patterns), and even personal information, which can be critical for identifying potential privacy risks. By systematically analyzing these patterns, model developers can identify unintended memorization or exposure of sensitive data, flag problematic behaviors, and implement safeguards. We believe minimally constrained generation has potential to serve not only as a behavioral probe but also as a proactive tool to reduce the risk of private information leakage in deployed models, especially for open-source models.

5 Related Work

Behavioral analyses under topic-specific prompts Numerous prior studies have analyzed LLM behavior under topic-specific prompts or conversational settings, examining different aspects of model behavior, such as the diversity of model outputs (Santurkar et al., 2023; Wu et al., 2024; Murthy et al., 2025; Jiang et al.,

2025; Xu et al., 2025; Hu et al., 2025; Lu et al., 2026), political viewpoints expressed by models (Hartmann et al., 2023; Westwood et al., 2025), social biases manifested in generated content (Salinas et al., 2024; Wang et al., 2025), and the identification or mitigation of harmful behaviors through red-teaming efforts (Ganguli et al., 2022; Perez et al., 2022). While these studies provide invaluable insights under structured and purpose-built prompts, such settings inevitably shape model behavior and may not capture models' near-unconstrained behaviors. Our work complements this line of research by analyzing models' underlying behavior using minimal, topic-neutral generation settings.

LLM Fingerprints One practical implication of our findings is the presence of systematic inter-model family discrepancies, which may be leveraged for LLM fingerprinting (Iourovitski et al., 2024; McGovern et al., 2025; Gao et al., 2024; Bitton et al., 2025). Existing approaches, often statistical or feature-based, compare distributions of linguistic, syntactic, or embedding-level features across models under targeted prompts (Hoscilowicz et al., 2024; Xu et al., 2024; Bansal and Sanghavi, 2025; Gao et al., 2024). Our results suggest that such distributional and stylistic patterns can naturally emerge under minimal, topic-neutral generation, indicating that they can complement traditional fingerprinting signals and provide a foundation for more robust and informative methods.

6 Discussions

Our minimally constrained generation provides a useful and new lens for interpreting model behavior, but we should acknowledge several limitations of this work, which suggest directions for future research.

Model-family level and individual-model level analysis While our analysis primarily focuses on differences at the model-family level, characterizing behavior at the level of individual models is also important for many practical applications (e.g., LLM fingerprinting). However, this is inherently more challenging to detect. To quantitatively illustrate these challenges, we measure the similarity between every pair of models used in our experiments, using one minus the Jensen–Shannon (JS) divergence between their semantic label distributions. Figure 11 in Appendix A.5 illustrates these similarities across sixteen models. We find that, with the exception of DeepSeek, models within the same family exhibit high mutual similarity, and differences in parameter size or reasoning style result in only minor variations. In particular, the similarity between `GPT-OSS-120B` and `GPT-OSS-20B` is 0.94, and between `Qwen3-235B-Instruct` and `Qwen3-235B-Thinking` is 0.93. This suggests that distinguishing individual models based on category information can be challenging and likely requires additional signals or finer-grained analyses, highlighting an important direction for future work. In Appendix A.5, we further discuss this challenge of distinguishing individual models using embeddings from the lens of LLM fingerprints.

Prompt design and its impact All prompts in our study are intentionally designed to be topic-neutral and open-ended. To avoid over-reliance on any single prompt, we use a diverse set of seed prompts spanning multiple styles and examine how prompt choice affects our main findings (Appendix A.6). While some variation emerges across different prompt styles, reflecting differences in phrasing and presentation (Appendix A.7), our use of diverse prompts reduces the likelihood that the patterns we report are driven by a narrow or idiosyncratic choice. Exploring a broader space of prompt designs remains an interesting direction for future work.

7 Conclusion

We systematically study how modern LLMs behave under unconstrained generation settings and show that, even under minimal prompting, model outputs span a broad semantic space while exhibiting distinct, family-specific topical preferences and levels of technical depth. We further identify significantly different patterns of degenerate text across model families, and our qualitative analysis reveals potential safety and privacy risks. Overall, our results demonstrate that minimally conditioned generation is an effective lens for monitoring underlying model tendencies, a perspective that is particularly relevant for open-source model developers.

Limitations

Our dataset is mainly produced by LLMs, it may include text content that is potentially harmful, offensive, or socially biased (e.g., stereotypes or derogatory language). While we manually inspected a

subset of the dataset and applied exact matching-based filters to remove sensitive words, not all samples were reviewed by human. Therefore, despite these efforts, the released data may still include undesirable content. To reduce risk, we will apply conservative redaction and content filtering before and after release, and we will provide clear documentation describing known limitations of the dataset and recommended safe-use guidelines (e.g., avoiding deployment in settings that may expose end users to unmoderated outputs).

References

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, and 1 others. 2025. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*.

Isabela Albuquerque, Jessica Schrouff, David Warde-Farley, Taylan Cemgil, Sven Gowal, and Olivia Wiles. 2024. Evaluating model bias requires characterizing its mistakes. *arXiv preprint arXiv:2407.10633*.

Yauhen Babakhin, Radek Osmulski, Ronay Ak, Gabriel Moreira, Mengyao Xu, Benedikt Schifferer, Bo Liu, and Even Oldridge. 2025. Llama-embed-nemotron-8b: A universal text embedding model for multilingual and cross-lingual tasks. *Preprint*, arXiv:2511.07025.

Parikshit Bansal and Sujay Sanghavi. 2025. Context-free synthetic data mitigates forgetting. *arXiv preprint arXiv:2505.13811*.

Yehonatan Bitton, Elad Bitton, and Shai Nisan. 2025. Detecting stylistic fingerprints of large language models. *arXiv preprint arXiv:2503.01659*.

Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Jorge, Célia Nunes, and Adam Jatowt. 2020. Yake! keyword extraction from single documents using multiple local features. *Information Sciences*, 509:257–289.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, and 1 others. 2021. Extracting training data from large language models. In *30th USENIX security symposium (USENIX Security 21)*, pages 2633–2650.

Lingjiao Chen, Matei Zaharia, and James Zou. 2024. How is chatgpt's behavior changing over time? *Harvard Data Science Review*, 6(2).

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

Kenneth Enevoldsen, Isaac Chung, Imene Kerboua, Márton Kardos, Ashwin Mathur, David Stap, Jay Gala, Wissam Siblini, Dominik Krzemiński, Genta Indra Winata, Saba Sturua, Saiteja Utpala, Mathieu Ciancone, Marion Schaeffer, Gabriel Sequeira, Diganta Misra, Shreeya Dhakal, Jonathan Rystrøm, Roman Solomatin, and 67 others. 2025. Mmteb: Massive multilingual text embedding benchmark. *arXiv preprint arXiv:2502.13595*.

Deep Ganguli, Liane Lovitt, Jackson Kernion, Amanda Askell, Yuntao Bai, Saurav Kadavath, Ben Mann, Ethan Perez, Nicholas Schiefer, Kamal Ndousse, and 1 others. 2022. Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned. *arXiv preprint arXiv:2209.07858*.

Irena Gao, Percy Liang, and Carlos Guestrin. 2024. Model equality testing: Which model is this api serving? *arXiv preprint arXiv:2410.20247*.

Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. 2023. Chatgpt outperforms crowd workers for text-annotation tasks. *Proceedings of the National Academy of Sciences*, 120(30):e2305016120.

Maarten Grootendorst. 2020. Keybert: Minimal keyword extraction with bert.

Neel Guha, Julian Nyarko, Daniel E. Ho, Christopher Ré, Adam Chilton, Aditya Narayana, Alex Chohlas-Wood, Austin Peters, Brandon Waldon, Daniel N. Rockmore, Diego Zambrano, Dmitry Talisman, Enam Hoque, Faiz Surani, Frank Fagan, Galit Sarfaty, Gregory M. Dickinson, Haggai Porat, Jason Hegland, and 21 others. 2023. Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models. *Preprint*, arXiv:2308.11462.

Jochen Hartmann, Jasper Schwenzow, and Maximilian Witte. 2023. The political ideology of conversational ai: Converging evidence on chatgpt's pro-environmental, left-libertarian orientation. *arXiv preprint arXiv:2301.01768*.

Rem Hida, Masahiro Kaneko, and Naoaki Okazaki. 2024. Social bias evaluation for large language models requires prompt variations. *arXiv preprint arXiv:2407.03129*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Jakub Hoscilowicz, Pawel Popiolek, Jan Rudkowski, Jedrzej Bieniasz, and Artur Janicki. 2024. Large language models as carriers of hidden messages. *arXiv preprint arXiv:2406.02481*.

Tiancheng Hu, Yara Kyrychenko, Steve Rathje, Nigel Collier, Sander van der Linden, and Jon Roozenbeek. 2025. Generative language models exhibit social identity biases. *Nature Computational Science*, 5(1):65–75.

J. D. Hunter. 2007. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95.

Dmitri Iourovitski, Sanat Sharma, and Rakshak Talwar. 2024. Hide and seek: Fingerprinting large language models with evolutionary learning. *arXiv preprint arXiv:2408.02871*.

Liwei Jiang, Yuanjun Chai, Margaret Li, Mickel Liu, Raymond Fok, Nouha Dziri, Yulia Tsvetkov, Maarten Sap, Alon Albalak, and Yejin Choi. 2025. Artificial hivemind: The open-ended homogeneity of language models (and beyond). *arXiv preprint arXiv:2510.22954*.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. 2023. Swe-bench: Can language models resolve real-world github issues? *arXiv preprint arXiv:2310.06770*.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Yang Liu, Yuanshun Yao, Jean-Francois Ton, Xiaoying Zhang, Ruocheng Guo, Hao Cheng, Yegor Klochkov, Muhammad Faaiz Taufiq, and Hang Li. 2023. Trustworthy llms: a survey and guideline for evaluating large language models' alignment. *arXiv preprint arXiv:2308.05374*.

Christina Lu, Jack Gallagher, Jonathan Michala, Kyle Fish, and Jack Lindsey. 2026. The assistant axis: Situating and stabilizing the default persona of language models. *arXiv preprint arXiv:2601.10387*.

Hope Elizabeth McGovern, Rickard Stureborg, Yoshi Suhara, and Dimitris Alikaniotis. 2025. Your large language models are leaving fingerprints. In *Proceedings of the 1stWorkshop on GenAI Content Detection (GenAIDetect)*, pages 85–95.

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

AI Meta. 2025. The llama 4 herd: The beginning of a new era of natively multimodal ai innovation. *https://ai. meta. com/blog/llama-4-multimodal-intelligence/, checked on*, 4(7):2025.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2022. Mteb: Massive text embedding benchmark. *arXiv preprint arXiv:2210.07316*.

Sonia Krishna Murthy, Tomer Ullman, and Jennifer Hu. 2025. One fish, two fish, but not the whole sea: Alignment reduces language models' conceptual diversity. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 11241–11258.

Timothy Nguyen. 2024. Understanding transformers via n-gram statistics. *Advances in neural information processing systems*, 37:98049–98082.

Dario Pasquini, Evgenios M Kornaropoulos, and Giuseppe Ateniese. 2025. {LLMmap}: Fingerprinting for large language models. In *34th USENIX Security Symposium (USENIX Security 25)*, pages 299–318.

Ethan Perez, Saffron Huang, Francis Song, Trevor Cai, Roman Ring, John Aslanides, Amelia Glaese, Nat McAleese, and Geoffrey Irving. 2022. Red teaming language models with language models. *arXiv preprint arXiv:2202.03286*.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Paul Röttger, Fabio Pernisi, Bertie Vidgen, and Dirk Hovy. 2025. Safetyprompts: a systematic review of open datasets for evaluating and improving large language model safety. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 27617–27627.

Alejandro Salinas, Amit Haim, and Julian Nyarko. 2024. What's in a name? auditing large language models for race and gender bias. *arXiv preprint arXiv:2402.14875*.

Shibani Santurkar, Esin Durmus, Faisal Ladhak, Cinoo Lee, Percy Liang, and Tatsunori Hashimoto. 2023. Whose opinions do language models reflect? In *International Conference on Machine Learning*, pages 29971–30004. PMLR.

Alex Tamkin, Miles McCain, Kunal Handa, Esin Durmus, Liane Lovitt, Ankur Rathi, Saffron Huang, Alfred Mountfield, Jerry Hong, Stuart Ritchie, and 1 others. 2024. Clio: Privacy-preserving insights into real-world ai use. *arXiv preprint arXiv:2412.13678*.

Zhen Tan, Dawei Li, Song Wang, Alimohammad Beigi, Bohan Jiang, Amrita Bhattacharjee, Mansooreh Karami, Jundong Li, Lu Cheng, and Huan Liu. 2024. Large language models for data annotation and synthesis: A survey. *arXiv preprint arXiv:2402.13446*.

TutteInstitute. 2025. Datamapplot: Creating beautiful plots of data maps. https://github.com/TutteInstitute/datamapplot. Version 0.6.4, MIT License.

Angelina Wang, Jamie Morgenstern, and John P Dickerson. 2025. Large language models that replace human participants can harmfully misportray and flatten identity groups. *Nature Machine Intelligence*, pages 1–12.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2019. Neural text generation with unlikelihood training. *arXiv preprint arXiv:1908.04319*.

Sean J Westwood, Justin Grimmer, and Andrew B Hall. 2025. Measuring perceived slant in large language models through user evaluations.

Fan Wu, Emily Black, and Varun Chandrasekaran. 2024. Generative monoculture in large language models. *arXiv preprint arXiv:2407.02209*.

Jiashu Xu, Fei Wang, Mingyu Ma, Pang Wei Koh, Chaowei Xiao, and Muhao Chen. 2024. Instructional fingerprinting of large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3277–3306, Mexico City, Mexico. Association for Computational Linguistics.

Weijia Xu, Nebojsa Jojic, Sudha Rao, Chris Brockett, and Bill Dolan. 2025. Echoes in ai: Quantifying lack of plot diversity in llm outputs. *Proceedings of the National Academy of Sciences*, 122(35):e2504966122.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Longfei Yun, Chenyang An, Zilong Wang, Letian Peng, and Jingbo Shang. 2025. The price of format: Diversity collapse in llms. *arXiv preprint arXiv:2505.18949*.

Appendix

In this Appendix, we present additional experiments (Appendix A), implementation details (Appendix B), prompts (Appendix C), theoretical explanation (Appendix D), and illustrative examples (Appendix E).

A Additional Experiments

A.1 Impact of Chat Templates

Jiang et al. (2025) show inter-model homogeneity, where different models converge on similar ideas in open-ended settings. Here, we reproduce their experimental setup and additionally evaluate an ablated variant in which chat templates are removed. Given that the two settings differ only in the use of chat templates, their inputs in terms of context information are essentially the same; therefore, we would anticipate comparable behavior. However, our results reveal a qualitatively opposite trend in inter-model diversity.

Figure 5 illustrates how fixed chat templates constrain model behavior. When prompted with "Write a metaphor about time." the standard chat template yields outputs that largely collapse into two dominant themes: "time as a river" and "time is a weaver.", as reported in the previous work (Jiang et al., 2025). However, the figure shows that removing the chat template produces more widely dispersed PCA representations and substantially more diverse metaphors (e.g., time as a "mirror", "architect", "currency", or "shadow"). This suggests that removing chat templates is more effective for exploring the diverse behaviors of models, echoing a previous work by Yun et al. (2025). Accordingly, we do not apply chat templates in our experiments and investigate LLMs' behaviors under near-unconditional settings.

A.2 Sensitivity Analysis on LLM-based Labeling

Motivation for Selecting the LLM Labeler Throughout our experiments, we used `GPT-OSS-120B` to assign category labels to model outputs. This choice was motivated by three observations. First, we initially experimented with n-gram–based keyword extraction methods (Campos et al., 2020; Grootendorst, 2020), but found them ineffective for labeling mathematical proofs and programming code. Second, based on randomly selected 100 samples, we observed that human annotation is both costly and insufficiently precise. Because LLM outputs span a highly diverse and open-ended range of topics, annotators frequently encounter ambiguous cases requiring subjective interpretation, which leads to low inter-annotator agreement. We emphasize that this issue likely stems not from poor annotator quality, but from the inherently open-ended and ambiguous nature of the task (e.g., a single generation often includes multiple topics). Third, we found that `GPT-OSS-120B` provides strong efficiency and instruction-following capabilities, which are critical for producing large-scale, well-formatted JSON

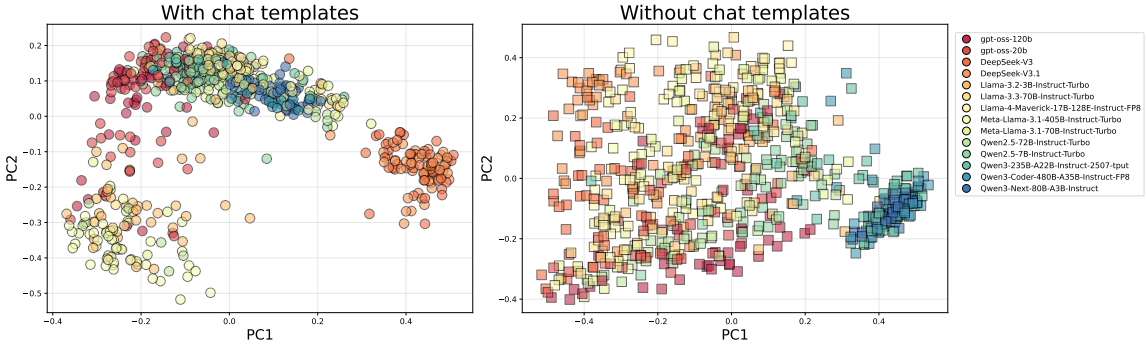


Figure 5: **Effect of removing chat templates on generative distributions**. The first two principal components of text generations from fourteen models in the legend, evaluated (left) with and (right) without chat templates. For each model, we generate 50 independent samples using the prompt "*Write a metaphor about time.*" with a temperature of 1.0 and a top-p of 0.9, following Jiang et al. (2025). Removing chat templates leads to more widely dispersed PCA representations and substantially more diverse metaphors. This indicates a template-induced shift in the models' generative distributions.

outputs. Additionally, we manually inspected 100 labels generated by `GPT-OSS-120B` and found them to be reasonably reliable, achieving an 87% agreement rate with human annotations[2].

Analysis of Potential Dual Bias A natural concern is whether the dual use of `GPT-OSS-120B` for both text generation and semantic labeling could introduce bias into our conclusions. To examine this possibility, we conduct a sensitivity analysis. Specifically, we uniformly sample $32,768$ model generations from the full dataset, which accounts for 12.8% of the full dataset, and compare the original labels produced by `GPT-OSS-120B` with those generated by `Gemini-2.5-Pro-Flash` and `Claude-4.5-Opus`. Figure 6 reproduces the category distributions for this subset across the three LLM labelers.

Across all three LLM labelers, the top five categories are largely consistent across labelers, suggesting the robustness of LLM-based semantic annotation. In particular, we observe consistent patterns in the proportion of math- and programming-related outputs (Figure 6). All three labelers indicate that GPT-OSS frequently generates mathematical or programming-related content, with the combined proportion exceeding 50% across labelers, which is observed in Figure 1c. Moreover, Llama places relatively less emphasis on science and technology, which is also observed in Figure 1c, as evidenced by having the lowest combined proportion of mathematics and programming outputs across all LLM labelers. Although absolute proportions vary slightly due to annotation differences, the relative ordering of categories and overall trends in major categories remain stable. This consistency suggests that any potential bias arising from the dual use of `GPT-OSS-120B` is not strong enough to affect our main findings. Our conclusions based on `GPT-OSS-120B` are robust to the choice of LLM used for semantic labeling; accordingly, we use `GPT-OSS-120B` to label the entire dataset.

Table 1: Agreement between LLM-based labelers.

Comparison	Exact Match
`GPT-OSS-120B` vs. `Claude-4.5-Opus`	74.97%
`GPT-OSS-120B` vs. `Gemini-2.5-Flash`	71.72%
`Claude-4.5-Opus` vs. `Gemini-2.5-Flash`	71.54%
All Three Match	63.29%

Table 1 further reports the agreement across labelers, measured using exact match[3]. Overall, we observe substantial consistency across labelers, suggesting again that our results are not overly sensitive to the choice of labeling model. The main disagreement pairs are (medicine, biology) and (computer science, mathematics), where these are also reasonable because a model generation can contain multiple topics (e.g., writing a Python code for mathematical problems). We expect that incorporating majority voting from multiple independent

[2]We ask the human annotator to check whether any part of the text generation includes the topic labeled by `GPT-OSS-120B`.

[3]We find that strict exact matching does not fully capture agreement between models, as they often express the same concept using slightly different forms (e.g., ("sport", "sports") and ("social science", "social sciences")). We therefore manually merge such cases and report a relaxed exact match metric. With the strict exact match, the agreement was approximately 3% lower.

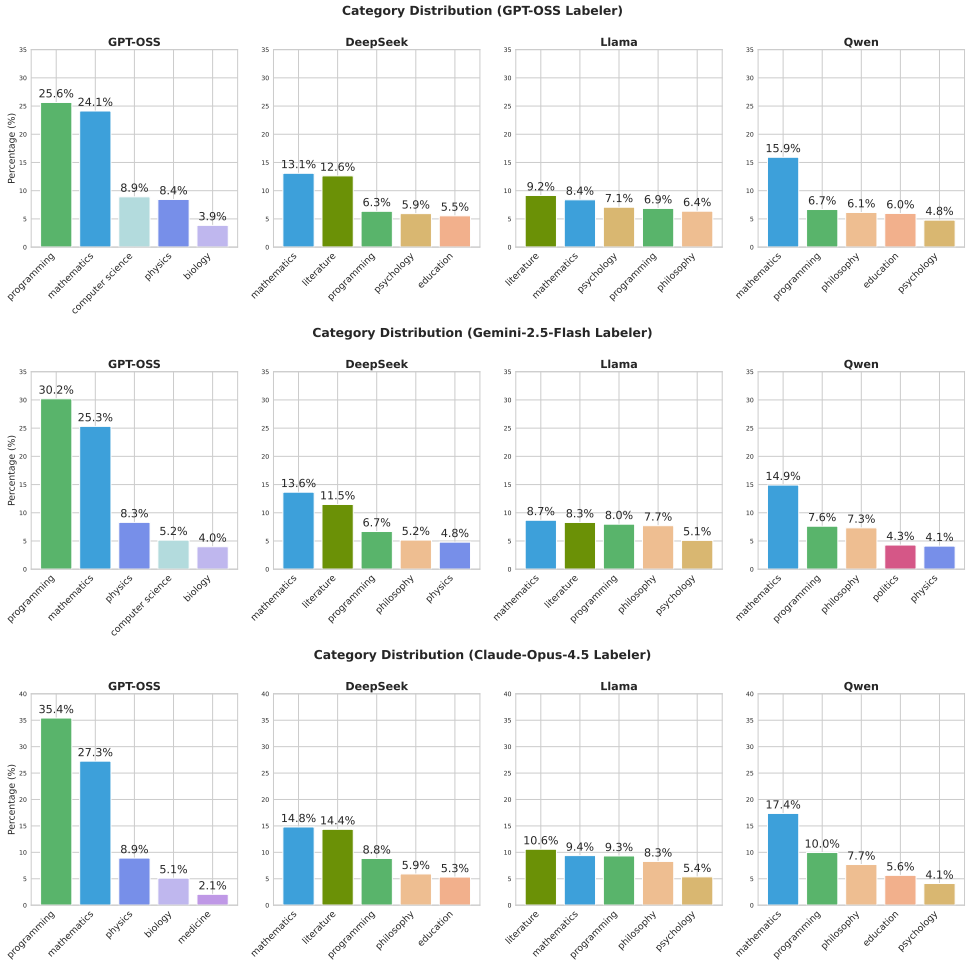


Figure 6: Sensitivity analysis of the category distributions. (Top) `GPT-OSS-120B`, (Middle) `Gemini-2.5-Pro-Flash`, and (Bottom) `Claude-4.5-Opus`. The analysis is conducted on a subset of 32, 768 model generations; consequently, the counts for `GPT-OSS-120B` differ slightly from those in Figure 1c. Overall, consistent with our main findings, each model family exhibits distinctive distributional patterns, and the general trends are stable regardless of the LLM used for labeling.

annotations could further reduce labeling bias; however, we also believe our main findings would be consistent as the general tendency is similar across model families in Figure 13.

Lastly, we examine the robustness of our results across different LLMs in Figure 3. Specifically, we consider `Gemini-2.5-Flash` and relabel the complexity of outputs categorized as mathematics and programming. Figure 7 shows the proportions across the four depth levels. While the differences are less pronounced than those in Figure 3, GPT-OSS still demonstrates a concentration at the Advanced and Expert levels, indicating that our findings are robust.

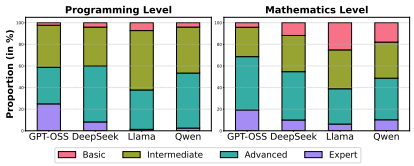


Figure 7: Distribution of depth levels for `Gemini-2.5-Flash`. Consistent with the results for `Claude-4.5-Opus` shown in Figure 3, GPT-OSS generates a larger proportion of text at the Advanced or Expert level.

A.3 Sensitivity Analysis on Embedding Model

Following the sensitivity analysis in Appendix A.2, a natural next question concerns the robustness of our visualization to the choice of embedding model. In this subsection, we examine whether our visualization remains stable across different embedding models. Specifically, we consider an alternative open-source, competitive embedding model, `Llama-embed-nemotron-8b` (Babakhin et al., 2025) and reproduce UMAP vi-

sualization plots in Figure 1. The implementation settings are identical to those used in the previous analysis with `Qwen3-Embedding-8B`, except that the UMAP hyperparameter "n_neighbors" is set to 8 instead of 5 to improve visual comparability.

We find that the resulting visualizations are strikingly similar. In particular, the high-density regions corresponding to different model families convey consistent patterns across embedding models: GPT-OSS concentrates on science and technology topics, Llama is more focused on the liberal arts, DeepSeek shows a strong concentration on religion, and Qwen frequently generates multiple-choice exam questions, which has been shown in the previous analysis with `Qwen3-Embedding-8B`. The relative spatial arrangement of these clusters remains highly consistent regardless of the embedding model used.

A.4 Additional subcategory analysis

This subsection presents additional subcategory analyses across ten more categories (Figures 9 and 10). Consistent with our main findings in Figure 2, we observe distinct distributional behaviors across model families on specific subcategories across these domains. For instance, in literature, GPT-OSS focuses largely on creative writing, generating little to no 'short story', 'fantasy', 'science fiction,' and 'romance' content. In contrast, Llama exhibits the opposite pattern, producing a more diverse range of literary content, with a focus on 'fantasy'. We also observe that all model families except DeepSeek favor 'creative writing' topic, whereas DeepSeek generates little to none of this content. In education, we find that Qwen has a strong concentration in 'multiple-choice exam questions', which may reflect its training on large amounts of textbooks or exams with multiple-choice questions.

A.5 Classification of individual model

Given that model behaviors differ across model families, one natural question is to what extent we can distinguish model families and further individual models. This question is central for LLM fingerprinting, where the goal is to identify the specific source LLMs from models (Pasquini et al., 2025; Gao et al., 2024). To assess the extent to which generated text reveals its underlying source, we construct a logistic regression model that predicts individual model source from embedding representations of individual outputs. We split the entire embedding dataset used in Section 3 into 80% training and 20% test.

Figure 12 shows a contingency table for this classifier. We find that model-family classification achieves 80.4% accuracy, whereas individual-model identification remains substantially more challenging, with accuracy dropping to 49.6%. This gap indicates that while model families exhibit distinguishable population-level behaviors, these signals are often too weak or inconsistent at the level of single generations, echoing our result in Figure 11.

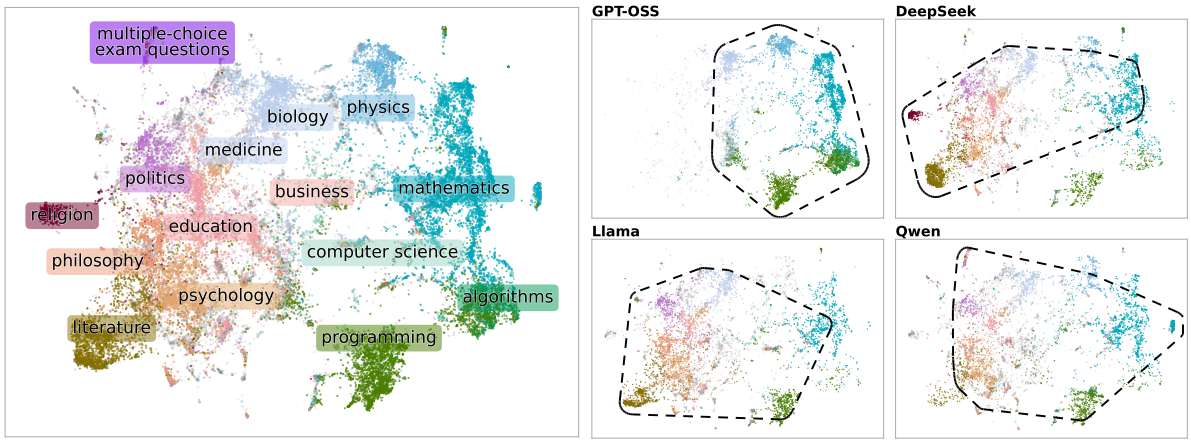
A.6 Robustness of the choice of the prompts

Effect of seed prompt choice on the main findings We further examine the robustness of our results with respect to prompt choice. Specifically, we split the prompt set into two subsets using stratified sampling over prompt styles, denoted as Set A and Set B. Set A contains three prompts per style, while Set B consists of the remaining three prompts for each style. The two sets are non-overlapping yet contain an equal number of seed prompts for each prompt style, thereby matching their prompt distributions. Using Sets A and B, we reproduce the category distributions shown in Figure 1c. If the results were sensitive to the choice of prompts, we would expect to see differences between the two visualizations.

Figure 13 illustrates no substantial variation across different random splits or relative to the original results in Figure 1c. This already suggests that our results are relatively insensitive to the choice of seed prompts. To further mitigate potential bias from a single random split, we evaluate the average JS divergence between label distributions across two random splits, using 10 independent random splits as described above. Table 2 shows that the distributions are highly similar across different splits, with very low standard deviation, indicating that our main findings are robust to the choice of prompts within each prompt style.

A.7 Effect of prompt style on the main findings

We conduct an additional experiment to examine whether different prompt styles affect our main findings. Following the previous setup in Appendix A.6, we split the prompts into two disjoint subsets, Set A and Set B. For each prompt style and subset, we compute the corresponding label distributions.



(a) (b)

Figure 8: UMAP visualizations of model outputs generated by the GPT-OSS, DeepSeek, Llama, and Qwen model families with `Llama-embed-nemotron-8b`. Implementation details are the same as those in Figure 1, and the only difference is the embedding model (`Qwen3-Embedding-8B` vs. `Llama-embed-nemotron-8b`). It shows that our visualization is fairly robust across different embedding models.

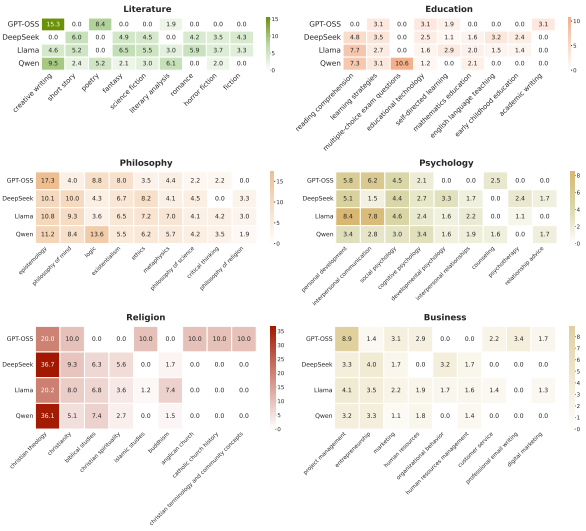


Figure 9: Additional subcategory analysis in literature, education, philosophy, psychology, religion, and business categories.

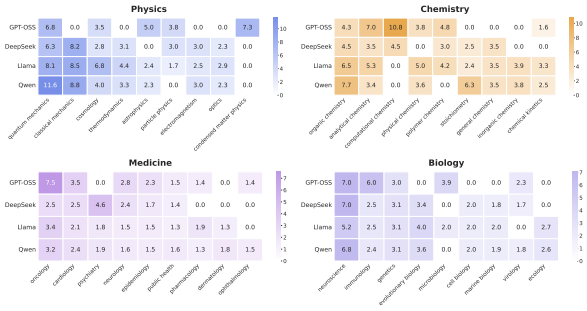


Figure 10: Additional subcategory analysis in physics, chemistry, medicine, and biology categories.

Table 2: JS divergence between label distributions derived from two random splits. We compute the average and standard deviation across 10 random splits. JS divergence ranges from 0 (identical distributions) to 1 (completely different distributions).

Model Family	Mean JS divergence	Standard deviation
GPT-OSS	0.0754	0.0020
DeepSeek	0.0708	0.0019
Llama	0.0530	0.0014
Qwen	0.0560	0.0011
Overall Mean	0.0638	0.0095

Figure 14 presents the label distributions for each of the six prompt styles and provides two interpretations. First, consistent with our earlier results, we observe low sensitivity within each prompt style, and thus no substantial differences between Set A and Set B. Second, we find that topic distributions can vary across prompt styles. Specifically, mathematics and programming consistently occupy top-ranked positions under conversational softeners, chain-of-thought, and punctuation-only prompts, whereas informative expository prompts exhibit a higher concentration in medicine and biology. These results demonstrate that prompt style systematically affects observed topic distributions, emphasizing the need for diverse prompt sets to mitigate dependence on a specific prompt style. Our use of multiple, qualitatively distinct prompt styles helps mitigate prompt-specific bias and supports the robustness of our main conclusions (Section 6). Nonetheless, incorporating an even broader range of prompt styles remains an important direction for future work.

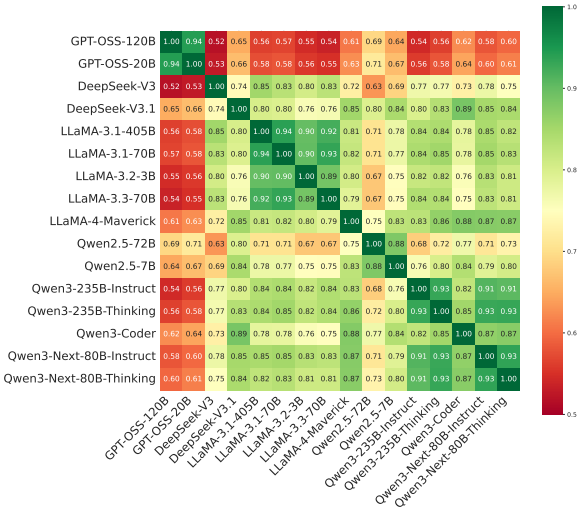


Figure 11: Similarity among the sixteen LLMs used in this paper. Similarity is measured as $1-$(JS divergence) and ranges from 0 to 1, with higher values indicating greater similarity. Models from the same series but with different parameter sizes or reasoning styles generally exhibit high similarity, suggesting that distinguishing individual models within these groups is yet more challenging.

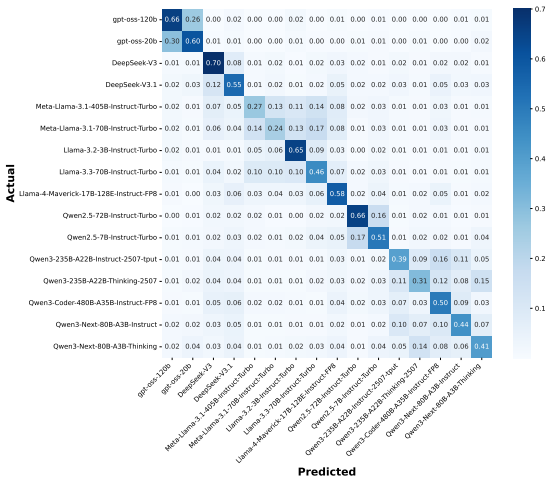


Figure 12: Contingency table for individual model classification. Overall accuracy reaches 80.4% at the model-family level but drops to 49.6% for individual models, indicating that fine-grained model identification remains challenging. Most misclassifications occur among closely related models that differ only in parameter size.

Table 3: Overview of language models used in this paper, including model family, reasoning style, and parameter scale.

Family	Model	Type	Parameters
GPT-OSS	GPT-OSS-20B (Agarwal et al., 2025)	Reasoning	20B
	GPT-OSS-120B (Agarwal et al., 2025)	Reasoning	120B
DeepSeek	DeepSeek-V3 (Liu et al., 2024)	Instruct	671B
	DeepSeek-V3.1 (Liu et al., 2024)	Hybrid	671B
Llama	LLaMA-3.1-70B (Dubey et al., 2024)	Instruct	70B
	LLaMA-3.1-405B (Dubey et al., 2024)	Instruct	405B
	LLaMA-3.2-3B (Dubey et al., 2024)	Instruct	3B
	LLaMA-3.3-70B (Dubey et al., 2024)	Instruct	70B
	LLaMA-4-Maverick (Meta, 2025)	Instruct	17B
Qwen	Qwen2.5-7B (Qwen et al., 2025)	Instruct	7B
	Qwen2.5-72B (Qwen et al., 2025)	Instruct	72B
	Qwen3-235B-Instruct (Yang et al., 2025)	Instruct	235B
	Qwen3-235B-Thinking (Yang et al., 2025)	Reasoning	235B
	Qwen3-Coder (Yang et al., 2025)	Instruct	480B
	Qwen3-Next-80B-Instruct (Yang et al., 2025)	Instruct	80B
	Qwen3-Next-80B-Thinking (Yang et al., 2025)	Reasoning	80B

A.8 Representative examples

Figure 15 provides some representative examples.

A.9 Degenerate text distribution

Figure 16 provides distinct degenerate text behavior across model families.

A.10 English Translation of Qwen sample

Figure 17 provides the English translation of the Qwen sample in Figure 4. More Chinese examples are available in Appendix E.

B Implementation Details

B.1 Model list

Table 3 provides a complete list of LLMs used in this paper.

B.2 Visualization

For UMAP, we set the number of neighbors to 5, the minimum distance to 0.01, and use the Euclidean metric for measuring distances between continuous features. For density estimation, we apply kernel density estimation with a bandwidth of 0.5. We use the Python library DataMapPlots (TutteInstitute, 2025) for embedding visualization and Matplotlib to visualize the convex hull of the embedded points (Hunter, 2007).

C Prompts

In Table 4, we provide a complete list of seed prompts. We also include the prompts used for semantic labeling as well as for the level tests in mathematics and programming.

Prompt for Semantic Labeling

[h] You are an expert in hierarchical semantic labeling. Analyze the text in section "## Text to analyze" and provide labels at two different levels of granularity:

- category: Choose a broad class of discipline that best captures the overall domain of the

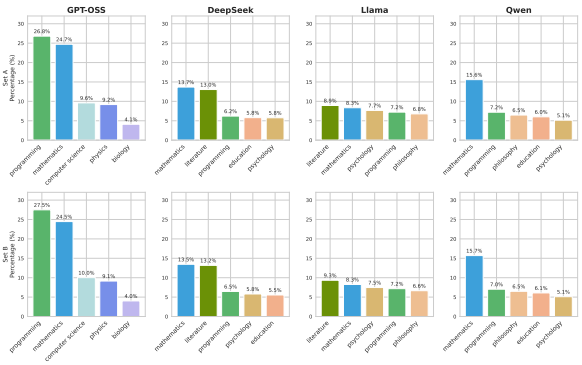


Figure 13: Replication of the experiments in Figure 1c using two random splits of the prompts. We perform stratified sampling across prompt styles: (Top) Set A contains three prompts for each style. (Bottom) Set B consists of the remaining three prompts not included in Set A. Each row in the figure corresponds to one split. The results show no substantial variation across different random splits or relative to the original results in Figure 1c. This suggests that our main findings are robust to the choice of prompts within each prompt style.

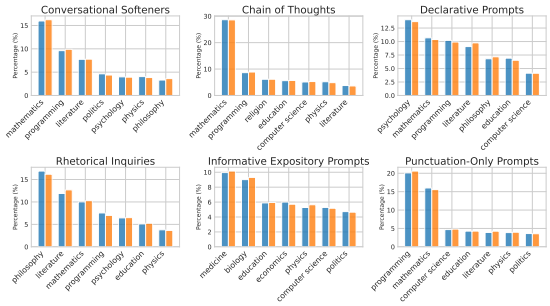


Figure 14: Label distributions across different seed prompt styles. Blue and orange denote different random splits. While there is no substantial variation within the same prompt style, label distributions vary across prompt styles.

text (e.g., mathematics, programming, physics, chemistry, biology, sport, art, music, literature, cooking, politics, philosophy, etc.).

- subcategory: Select a more precise topic that accurately reflects the main focus within the chosen discipline (e.g., algebra, web development, quantum mechanics, organic chemistry, biochemistry, swimming, modern art, classical music, Russian literature, eclipse, democracy, Plato, etc.).

Output format:
- Provide only the JSON output; no additional text or explanations.
- Provide labels in this exact JSON format:
{{
“category": “Broad topic of the text",
“subcategory": “Specific sub-topic within the category class",
}}

Text to analyze:
{text}

Again, provide only the JSON output; no additional text or explanations.

Prompt for Math Level Test

[h] You are an expert in mathematics education. Analyze the mathematical content in the section “## Text to analyze" and classify its difficulty level.

Classification Categories (choose exactly one):

1. **basic**: Basic arithmetic and numeracy, counting, simple word problems, fractions and decimals, ratios and proportions, basic statistics (mean, median), fundamental geometry (shapes, area, perimeter), pre-algebra, basic algebraic expressions, and introduction to negative numbers. Typically middle school level or lower (grades K–8).

2. **intermediate**: Standard secondary-school mathematics, including Algebra I and II, coordinate geometry, geometry with formal proofs, trigonometry, pre-calculus, sequences and series, and introductory probability and combinatorics. Typically high school level (grades 9–12).

3. **advanced**: Undergraduate-level university mathematics, including single-variable and multivariable calculus, linear algebra, differential equations, discrete mathematics, probability theory with formal definitions, and introductory real analysis (proof-based but foundational). Typically college level with emphasis on mathematical rigor and abstraction beyond high school.

Model / Category: GPT-OSS / programming

In fact, it's a known problem from the **USACO** (maybe). [...] \"the minimum cost to reach node 1 from node N in a graph where edges exist between i and its multiples/factors\". [...] Alternatively, think **DP with Dijkstra-like approach**. [...] Implementation plan:\n\n- For each node i (1..N), we need to generate its neighbors j where:\n\n1). [...]

Model / Category: DeepSeek / religion

[...] Well, you start by taking it back to the Word. Jesus says in **John 6:63** that his words are “full of the Spirit and life.” You read your **Bible**. [...] Knock, and the door will be opened to us (**Matthew 7:7**). [...] **Hebrews 10:24-25** says, “Let us think of ways to motivate one another to acts of love and good works. [...[

Model / Category: Llama / literature

[...] **Part 3: Into the Dark**\nKerry was skeptical, but Jimmy could see the desperation in Aisling's eyes. He knew that look, the look of someone who was willing to cling to any shred of hope.\n'What kind of dreams?' Jimmy asked, his voice low.\nAisling swallowed hard. 'I see them, the children. [...]

Model / Category: Qwen / education

Evidence indicates that **the most effective way to develop students’ advanced thinking skills** is to teach them directly. In this module, you will find tools to guide your students [...] **Integrating the arts and humanities into STEM education** (STEAM) can enhance students' critical thinking, creativity, and empathy, [...]

Figure 15: **Examples of model outputs and their category labels**. We find that most generated texts appear coherent and meaningful, and that the LLM-based labeler produces sensible label annotations. For instance, the GPT-OSS sample describes an attempt to solve a problem that appears to be from the USA Computing Olympiad using a dynamic programming algorithm and is categorized as programming. The DeepSeek sample references Bible verses from Matthew and Hebrews and is categorized as religion. Llama generates a segment of fictional narrative, and Qwen outlines an approach to developing critical thinking; these outputs are accordingly categorized as literature and education, respectively.

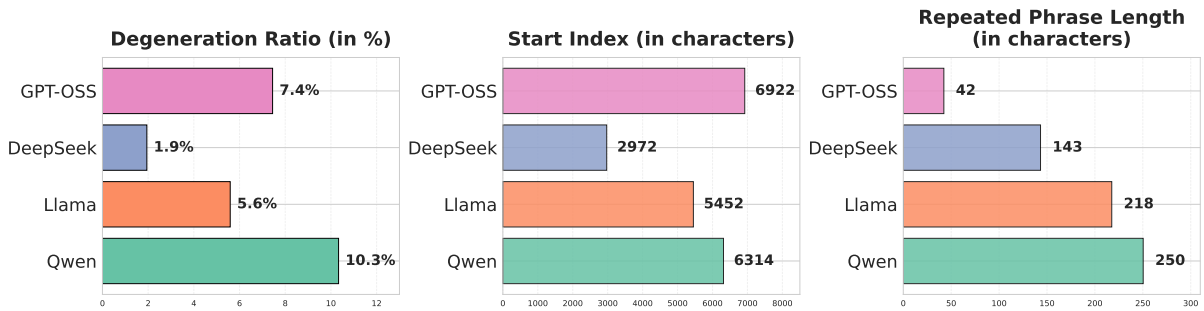


Figure 16: **Distinct degenerate text behavior across model families.** The left figure shows how likely model outputs include degenerate text, the middle shows the start index of degenerate text within the generated sequence, and the right figure shows the average length of repeated phrases. We observe substantial variation across model families in the frequency of degenerate text, the position at which it begins, and the length of repetitive phrases. In particular, degenerate text occurs in only 1.9% of DeepSeek generations, while it appears in 10.3% of Qwen generations, representing a 5.4-fold increase.

Figure 17: English translation of the Qwen sample in Figure 4. (Left) The original degenerate text; (Right) its translation. Translations are produced using Google Translate. It shows conversational artifacts in Chinese.

4. **expert**: Advanced or research-oriented mathematics requiring strong undergraduate foundations, such as advanced real analysis, abstract algebra, topology, measure theory, functional analysis, advanced number theory, algebraic geometry, and other graduate or PhD-level topics. This category also includes mathematical competition or olympiad-style problems (e.g., AMC, AIME, USAMO, IMO, Putnam) characterized by non-routine problem statements that require creative insight, clever constructions, or deep problem-solving techniques.

5. **unclassifiable**: The text cannot be properly classified due to one of the following reasons: too short to determine difficulty, not related to mathematics, contains no meaningful mathematical content, or is ambiguous/unclear in its mathematical intent.

Output format:
- Provide only the JSON output; no additional text or explanations.
- Provide the classification in this exact JSON format:
{{
“difficulty": “one of: basic, intermediate, advanced, expert, unclassifiable",
“reasoning": “Brief explanation of why this difficulty level was chosen"
}}

Text to analyze:
{text}

Again, provide only the JSON output; no additional text or explanations.

Prompt for Programming Level Test

You are an expert in computer science education. Analyze the programming content in the section “## Text to analyze" and classify its difficulty level.

Classification Categories (choose exactly one):

1. **basic**: Introductory programming concepts intended for complete beginners, including variables, basic data types, simple conditional statements (if/else), basic loops (for/while), simple input/output, very simple functions or procedures, and block-based or Scratch-style logic. Typically middle school level or lower. Explanations are highly guided and step-by-step. No use of arrays, recursion, or algorithmic reasoning.

2. **intermediate**: Intermediate programming concepts commonly taught in AP-level or equivalent courses, such as arrays, lists, and dictionaries (basic usage), functions with parameters and return values, nested loops, introductory object-oriented programming (classes and objects), basic file I/O, and simple algorithms (e.g., linear search, basic sorting). Typically high school level. The focus is on writing correct programs rather than optimization or formal analysis.

3. **advanced**: Undergraduate computer science material covering core theory and systems, including data structures (trees, heaps, hash tables, graphs), algorithm design and analysis (Big-O notation, basic correctness reasoning), recursion and divide-and-conquer, introductory to intermediate dynamic programming, database concepts, operating systems and networking basics, web development fundamentals, software engineering principles, design patterns, and introductory concurrency or multithreading. Typically college level. Explanatory or instructional texts without competitive constraints belong here.

4. **expert**: Advanced or research-oriented computer science content that assumes strong undergraduate foundations, such as advanced algorithms with formal proofs, distributed systems internals and consistency models, machine learning implementations from first principles, compiler design and optimization, operating system internals (e.g., scheduling, memory models), advanced database systems, formal verification, program analysis, and research-style system design or implementations. Typically graduate or PhD level. This category also includes competitive programming or interview-style problem-solving content (e.g., Codeforces, ICPC, IOI, LeetCode Hard–style problems), characterized by explicit problem statements with input/output specifications, tight time or space constraints, and an emphasis on optimization, edge cases, and performance tuning.

5. **unclassifiable**: The text cannot be properly classified due to one of the following reasons: too short to determine difficulty, not related to programming or computer science, contains no meaningful programming content, or is ambiguous/unclear in its programming intent.

Output format:
- Provide only the JSON output; no additional text or explanations.
- Provide the classification in this exact JSON format:

{{
"difficulty": "one of: basic, intermediate, advanced, expert, unclassifiable",
"reasoning": "Brief explanation of why this difficulty level was chosen"
}}

Text to analyze:
{text}

Again, provide only the JSON output; no additional text or explanations.

D Theoretical explanation

This section provides a theoretical explanation on how reinforcement learning (RL) influences model inference, which answers why our generation setting is more relevant to the pre-training dataset. To be more specific, let $\mathcal{V}$ be a finite vocabulary, and let $x \in \mathcal{V}^L$ denote a context consisting of L tokens. Consider an LLM policy that is parameterized by θ, takes x as input and generates a fixed-length sequence fixed-length sequence $y_{1:T} \in \mathcal{V}^T$. Then, we can factorize the policy distribution as

$$\pi_\theta(y_{1:T} \mid x) = \prod_{t=1}^{T} \pi_\theta(y_t \mid x, y_{<t}), \qquad (1)$$

where $y_{<t} := (y_1, \ldots, y_{t-1})$. Let $R : \mathcal{V}^{L+T} \to \mathbb{R}$ be a scalar reward assigned to the full sequence. Then the following theorem shows how the Kullback-Leibler (KL)-regularized proximal algorithm in reinforcement learning (RL) updates a conditional policy.

Theorem D.1 (Re-weighting style closed-form solution)**.** *Consider the KL-regularized proximal update over distributions on $\mathcal{V}^T$:*

$$\pi^{(k+1)} = \arg\max_{\pi} \left\{ \mathbb{E}_x \mathbb{E}_{y\sim\pi(\cdot|x)}[R(x,y)] - \frac{1}{\eta}\mathrm{KL}\Big(\pi \,\|\, \pi^{(k)}\Big) \right\}, \qquad (2)$$

where $\eta > 0$ is a regularization parameter and $KL(\cdot \mid \cdot)$ is the Kullback–Leibler divergence. Then, its solution has a closed form:

$$\begin{aligned} &\pi^{(k+1)}(y \mid x) \\ &= \frac{\pi^{(k)}(y \mid x)\exp(\eta R(x,y))}{\sum_{y'\in\mathcal{V}^T} \pi^{(k)}(y' \mid x)\exp(\eta R(x,y'))} \\ &\propto \pi^{(k)}(y \mid x)\exp(\eta R(x,y)). \end{aligned} \qquad (3)$$

Proof. For each fixed x, define

$$\begin{aligned} p_x(y) &:= \pi^{(k)}(y \mid x), \\ q_x(y) &:= \pi(y \mid x), \\ Z(x) &:= \sum_{y\in\mathcal{Y}} p_x(y) e^{\eta R(x,y)}. \end{aligned}$$

Because $\mathcal{Y}$ is finite and p_x is a probability distribution, we have $0 < Z(x) < \infty$ for every x. Now define

$$g_x(y) := \frac{p_x(y)e^{\eta R(x,y)}}{Z(x)}, \qquad y \in \mathcal{Y}.$$

Then $g_x(\cdot)$ is a probability distribution on $\mathcal{Y}$.

We claim that for every x and every distribution q_x,

$$\begin{aligned} &\sum_{y\in\mathcal{Y}} q_x(y)R(x,y) - \frac{1}{\eta}\mathrm{KL}(q_x\|p_x) \\ &\quad = \frac{1}{\eta}\log Z(x) - \frac{1}{\eta}\mathrm{KL}(q_x\|g_x). \end{aligned}$$

Indeed,

$$\begin{aligned} &\mathrm{KL}(q_x\|g_x) \\ &= \sum_{y\in\mathcal{Y}} q_x(y)\log\frac{q_x(y)}{g_x(y)} \\ &= \sum_{y\in\mathcal{Y}} q_x(y)\log\frac{q_x(y)Z(x)}{p_x(y)e^{\eta R(x,y)}} \\ &= \sum_{y\in\mathcal{Y}} q_x(y)\log\frac{q_x(y)}{p_x(y)} \\ &+ \sum_{y\in\mathcal{Y}} q_x(y)\log Z(x) \\ &- \eta\sum_{y\in\mathcal{Y}} q_x(y)R(x,y) \\ &= \mathrm{KL}(q_x\|p_x) + \log Z(x) - \eta\sum_{y\in\mathcal{Y}} q_x(y)R(x,y), \end{aligned}$$

where we used $\sum_y q_x(y) = 1$. Rearranging gives the claim.

Applying this identity with $q_x(\cdot) = \pi(\cdot \mid x)$ and $p_x(\cdot) = \pi^{(k)}(\cdot \mid x)$, we obtain

$$\begin{aligned} &\mathbb{E}_{y\sim\pi(\cdot|x)}[R(x,y)] - \frac{1}{\eta}\mathrm{KL}\big(\pi(\cdot \mid x) \,\|\, \pi^{(k)}(\cdot \mid x)\big) \\ &\qquad = \frac{1}{\eta}\log Z(x) - \frac{1}{\eta}\mathrm{KL}\big(\pi(\cdot \mid x) \,\|\, g_x\big). \end{aligned}$$

Taking expectation over $x \sim \mu$ yields

$$J(\pi) = \frac{1}{\eta}\,\mathbb{E}_{x\sim\mu}[\log Z(x)] - \frac{1}{\eta}\,\mathbb{E}_{x\sim\mu}\Big[\mathrm{KL}\big(\pi(\cdot \mid x) \,\|\, g_x\big)\Big].$$

Since KL divergence is always nonnegative,

$$\mathrm{KL}\big(\pi(\cdot \mid x) \,\|\, g_x\big) \geq 0 \qquad \text{for all } x,$$

and therefore

$$J(\pi) \leq \frac{1}{\eta}\, \mathbb{E}_{x\sim\mu}[\log Z(x)].$$

Equality holds if and only if

$$\mathrm{KL}\big(\pi(\cdot \mid x) \,\|\, g_x\big) = 0 \qquad \text{for } \mu\text{-almost every } x,$$

which is equivalent to

$$\pi(\cdot \mid x) = g_x(\cdot) \qquad \text{for } \mu\text{-almost every } x.$$

Hence any maximizer must satisfy

$$\pi^{(k+1)}(y \mid x) = g_x(y) = \frac{\pi^{(k)}(y \mid x) e^{\eta R(x,y)}}{\sum_{y' \in \mathcal{Y}} \pi^{(k)}(y' \mid x) e^{\eta R(x,y')}}.$$

This proves the stated closed form.

□

Implications This demonstrates that a single-step update in KL-regularized RL can be expressed in closed form, affecting the policy values only at specific input x, which has been used in RL training. In other words, RL changes the policy values for the input x formatted with a specific chat template. This format is typically very different from a free-format used in the pre-training procedure, suggesting that RL's impact on unconstrained inputs will be smaller. In addition, since next-token loss trains the model to predict the most likely token given the context (i.e., N-gram statistics suggested as in Nguyen (2024)), our unconstrained generation effectively reflects the model's prediction of the most probable next token, capturing pre-training data distributions.

E Additional Examples

In this section, we provide additional representative examples of degenerate text, including potentially personally identifiable information (PII), multiple-choice questions, Chinese text, conversational artifacts, and question-answer artifacts in Figures 18 and 19.

We find that all but two of the generated PII instances (e.g., social media accounts or personal webpages) correspond to accessible and real-world accounts, suggesting most LLM generations were hallucinated. However, a small portion of the text corresponds to genuine information. In other words, it is possible to extract personal information from LLMs, which raises privacy and reliability concerns.

\n【如果您有新闻线索，欢迎向我们报料，一经采纳有费用酬谢。报料微信关注：i[redacted]sb，报料QQ：33[redacted]05[redacted]2】	[If you have any news tips, please feel free to contact us. We offer compensation for accepted tips. Follow us on WeChat: i[redacted]sb, or contact us via QQ: 33[redacted]05[redacted]2]
\n\n为了避免对电缆造成损伤，电缆悬吊时不应使用 **A. 钢丝** 和 **B. 铁丝**。因为这些材料硬度高，容易在悬吊过程中对电缆的护层或绝缘层造成损伤。正确的做法是使用不会对电缆造成损伤的柔性材料，例如麻绳。因此，答案是 **A. 钢丝** 和 **B. 铁丝**。\n\n综上所述，正确答案是:**A. 钢丝** 和 **B. 铁丝**。	To avoid damaging the cable, **A. steel wire** and **B. iron wire** should not be used when suspending the cable. These materials are hard and can easily damage the cable's sheath or insulation during suspension. The correct practice is to use flexible materials that will not damage the cable, such as hemp rope. Therefore, the answer is **A. steel wire** and **B. iron wire**.

Figure 18: English translation of Chinese samples not necessarily from Qwen. (Left) The original degenerate text; (Right) its translation. (Top) We mask portions of the text because they contain publicly available personally identifiable information. (Bottom) This example shows a multiple-choice question. Translations are produced using Google Translate. It shows conversational artifacts in Chinese.

For more information about the event, please visit https://XXXXneXXXXiXXXXhaXXXX.org/\nContact: Thelma Agano\nEmail: info@XXXXneXXXXiXXXXhaXXXX.org\nPhone: +2XX 55 XX1 7XX6\nWebsite: XXXXneXXXXiXXXXhaXXXX.org\nTwitter: @XXXXneXXXXiXXXXhaXXXX\nFacebook: @XXXXneXXXXiXXXXhaXXXX\	**For the most precise and clear answer**:\n\nThe answer is: B. Temperature gradually decreases\n} **To comply with the instruction and provide the most accurate and clear answer**:\n\nThe answer is: B. Temperature gradually decreases\n} **For the most precise and clear response:**\n\nThe answer is: B. Temperature gradually decreases\n} **To provide the most accurate and clear answer**:\n\nThe answer is: B. Temperature gradually decreases\n} **To comply with the instruction and provide the most precise answer**:\n\nThe answer is: B. Temperature gradually decreases\n} **For the final, clear, and accurate answer**:\n\nThe answer is: B. Temperature gradually decreases\n} **To ensure the answer is clear, direct, and accurate**:\n\nThe answer is: B.Temperature gradually decreases\n}
Follow the author on social media for more updates and behind-the-scenes insights into the writing process. \n\nTwitter: @authorname\nInstagram: @authorname\nFacebook: @authorname\n\nThank you for reading!	
Tell us about the issues that matter to you and share your stories of positive impact. Together, we can create a better world for everyone. Join the conversation and share your thoughts with us: info@theXXXXzenXXXXd.com	I think we've reached a good stopping point. It's been a productive discussion, and I'm glad we could have it. \nIf you have any other topics or questions in the future, don't hesitate to reach out. I'm always happy to chat! \nTake care, and have a great day! \nYou're welcome, and I'm glad we could have this conversation. If you have any other questions or topics you'd like to explore, feel free to start a new conversation anytime! \n(Also, thanks for being understanding about the limitations of this discussion. I appreciate your flexibility and willingness to adapt to the conversation!) \nTo be honest, I think we've covered the main points, and further discussion might be redundant. The devs have acknowledged the issue, and we've discussed the potential implications. \nIf you're still interested in gathering [...]
W.-XXXX K.-XXXX H.-XXXX Y.-XXXX J.-XXXX ()\nDepartment of Chemical and Biomolecular Engineering, XXXXXXXX, XXXXXXXX, XXXX, Seoul XXXX, Republic of Korea\ne-mail: XXXX@XXXX.ac.kr\	
Japan Studies Program XXXXXXXXXXXX, XXXX XXXX, Shibuya-ku, Tokyo 150-0002, Japan Tel: +81-03-XXXX-XXXX Fax: +81-03-XXXX-XXXX Email: JapanStudies@XXXX.ac.jp	

Figure 19: Additional degenerate text examples. (Left) potentially PII examples. (Right) examples of question-answer artifacts and conversational artifacts.

Table 4: A complete list of the 36 seed prompts used in this paper and their corresponding categories is provided. Across all experiments, one of the 36 seed prompts is selected uniformly at random. The set of prompts includes a diverse mix of both complete and incomplete sentences to ensure variability in model behavior.

Prompt Category	**Seed Prompt**
Conversational softeners	You know, Actually, Well, So, Anyway, In fact,
Chain of thoughts	Let's think step by step. Let's break this down. Let's first consider the context. Let's analyze the main idea. Let's reason about this carefully. Let's approach this systematically.
Declarative prompts	I want to think about something. I want to learn about something. I want to explore something. I want to consider something. I want to talk about something. I want to understand something.
Rhetorical Inquiries	Shall we think about something? Shall we explore something? What if we examine something? Should we look into something? Could we analyze something? What shall we try to understand?
Informative expository prompts	This paper discusses This article presents This study analyzes The authors argue that Evidence indicates that The findings demonstrate
Punctuation-only prompts	. , ? ! ... :