Decoding Path Recovery in Number Generation: Shifting to Intermediate Layers and Top-k Ranks

Jasivan Alex Sivakumar & Nafise Sadat Moosavi
School of Computer Science
University of Sheffield
Sheffield, UK
{jasivakumar1,n.s.moosavi}@sheffield.ac.uk

Abstract

Existing work on numerical reasoning typically interprets incorrect outputs as failures of representation or computation. We instead study the recoverability of numerical errors: whether a correct answer remains accessible when considering a model's intermediate layers or lower-ranked tokens despite an incorrect final output. Across eight open-source language models, we find that on average 30% of arithmetic failures remain recoverable beyond standard decoding, indicating that some numerical errors arise from selecting the wrong token rather than failing to compute it altogether. However, this recoverability is not universal. As arithmetic complexity increases, the correct answer progressively disappears not only from the model's final prediction but also from alternative candidates and intermediate computations, revealing the limits of decoding-based recovery. Finally, we show that recoverable and unrecoverable failures exhibit distinct predictive dynamics with regard to probability and rank behaviour in intermediate layers, providing insight into when interventions such as adaptive decoding or post-generation correction are likely to succeed. Overall, our findings characterise both the opportunities and limitations of exploiting latent numerical information during generation.

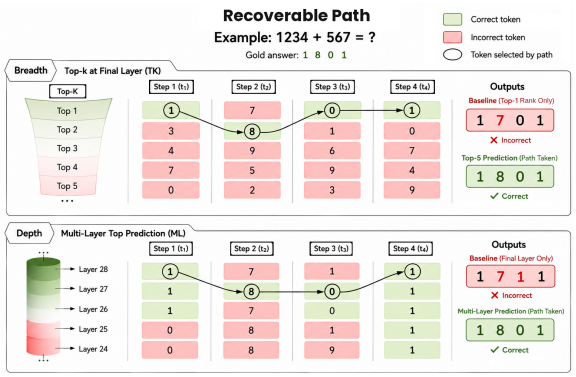


Figure 1: Recoverable decoding trajectories for model breadth, top-k (top) and depth, multi-layer (bottom). Green cells denote correct tokens and red cells denote incorrect predictions, with alternative paths that generate correct answer shown.

1 Introduction

Numbers are a ubiquitous part of natural language. Humans use numbers to enhance communication (Chrisomalis, 2010) and therefore numbers complement language, for example, in areas such as cooking recipes, weather forecasting and finance. However, arithmetic reasoning presents a unique challenge to Large Language Models (LLM) compared to many natural language generation tasks (Li et al., 2025). It requires models to understand the meaning of individual digits and their composition into multi-digit numbers (Guedj, 1996), execute rigid rule-based algorithms to process numbers, and express the output of such computation into multi-digit numbers. This can be summarised as: (a) understanding the input, (b) performing the computation, and (c) generating the answer. Unlike open-ended generation tasks such as translation (Kocmi et al., 2022) and summarisation (Goldsack et al., 2022), arithmetic reasoning typically admits a single canonical answer. Consequently, mastering arithmetic reasoning is essential for broader natural language generation applications, particularly data-to-text generation (Moosavi et al., 2021), where LLMs must accurately translate raw non-linguistic numerical sources into reliable textual reports.

Despite advances in numerical representations (Zhou et al., 2025; Sivakumar and Moosavi, 2025; Golkar et al., 2023; McLeish et al., 2024), interpretability (Bertolazzi et al., 2025; Mamidanna et al., 2025; Christ et al., 2025; Kadlčík et al., 2025; Nikankin et al., 2025), and number generation objectives (Fei et al., 2025; Zausinger et al., 2025; Wallace et al., 2019), it remains unclear whether numerical errors arise because the information re-

Proceedings of the 19th International Natural Language Generation Conference: Long Papers, pages 799–814
October 17-21, 2026 ©2026 Association for Computational Linguistics

quired to generate the correct answer is absent from the model's predictive distribution or because it is simply not prioritised by standard decoding. We refer to this property as recoverability: the extent to which a correct answer remains accessible within the model's predictive process despite an incorrect final output. Understanding recoverability allows us to distinguish between failures that can potentially be corrected through alternative decoding trajectories and failures that reflect limitations arising earlier in the model's processing.

The generation literature has established that decoding behaviour is shaped by task likelihood. For text generation, different decoding strategies have been explored to combat text degeneration and improve factuality and diversity (Holtzman et al., 2020; Chuang et al., 2024; Li et al., 2023; Shi et al., 2024). Numerical reasoning, however, has received comparatively little attention from this perspective, leaving open the question of how much numerical performance is limited by the model's predictions versus the decoding process that realises them.

In this work, we investigate the recoverability of numerical errors by expanding the predictive space available during generation. We explore two complementary axes: depth, by considering predictions from intermediate layers via early exit, and breadth, by considering lower-ranked candidates from the final layer's output distribution[1]. Using the FERMAT dataset (Sivakumar and Moosavi, 2023), we analyse arithmetic problems spanning different operand formats, operations, results length, within both symbolic and worded problems.

Our analysis reveals that numerical failures arise from multiple mechanisms rather than a single deficiency in arithmetic reasoning. Out of the failures cases using standard search, on average 30% of the time, the information required to generate the correct answer already exists beyond the model's final prediction, indicating that errors can originate from how models select and realise numerical information during generation. However, this recoverability is not universal. As arithmetic complexity in terms of operand and result length as well as operation difficulty increases, the correct answer gradually disappears not only from the final prediction but also from intermediate-layer outputs and lower-ranked alternatives, suggesting a transition from recoverable generation failures to failures rooted in the model's computation. Furthermore, we find that recoverable information is concentrated at particular generation stages, candidate ranks, and model layers, exposing structured patterns in how numerical errors emerge during generation. Overall, these findings characterise the limits of decoding-based interventions for arithmetic reasoning and provide a principled framework for distinguishing generation failures from computation failures.

[1]Our codes and sample results are made publicly available at https://github.com/jasivan/Decoding-Path-Recovery.

2 Related Work

2.1 Improving Numerical Representations

In standard LLMs, numbers are typically treated as discrete word tokens, subject to the same distributional training objectives as natural language (Wallace et al., 2019). This discrepancy often results in a significant number gap, where models fail to generalise across arithmetic or real-world contexts (Li et al., 2025). Several works aim to close the number gap by proposing alternative encoding schemes. These include mapping numbers via Fourier features (Zhou et al., 2025), utilising dedicated vector-based encodings for numerical values (Golkar et al., 2023), incorporating relative digit positions to preserve structural information (McLeish et al., 2024), or aggregating digit-level embeddings into unified numerical representations (Sivakumar and Moosavi, 2025).

Some works explore rewriting numbers in different formats in text: decomposing numbers into each base 10 contribution (Muffo et al., 2022), adding the number of digits of a number as an extra signal (Schwartz et al., 2024), writing the numbers in standard form (Zhang et al., 2020). Conversely, a recent work on pre-training (Kadlčík et al., 2025) shows that within the embedding space pre-trained models have remarkable representations for numbers, suggesting that models do not need a specialised number representation.

Moreover, despite the large body of work on number representation both implicitly within the model and explicitly in the input, most LLMs still use pre-trained learning of digits or numbers as their number representations. As an alternative to the widely used Byte-Pair Encoding (Sennrich et al., 2016), two schools of thought are employed in contemporary LLMs, on one side digit-level representation where each number is tokenised into individual digits (Spithourakis and Riedel, 2018) as with models like Qwen3 (Yang et al., 2025), Mistral 7B (Jiang et al., 2024) or Gemma2 (Team

et al., 2024); this allows a human-like tokenisation of numbers. Alternatively, models like Llama-3 (Grattafiori et al., 2024), GPT-4 (Achiam et al., 2023) or DeepSeek (Guo et al., 2025) use unique tokens for all numbers from 0 to 999 to find an optimal trade-off between compression of numbers and unique representations (Dagan et al., 2024). By exploring both Qwen3 and LLama-3, we ensure to better understand number generation under both tokenisation schemes.

2.2 Diagnosing Numerical Computation

The literature explores the drawbacks that LLMs have in numerical reasoning by explicitly measuring the entire process via carefully curated datasets: FERMAT (Sivakumar and Moosavi, 2023) looks at different number representation and operation for worded maths problems to show that LLMs struggle working with numbers that are not seen in training, fail to maintain performance on symmetric commutative problems or multi operation questions. Yan et al. (2025) specifically test addition to show that for simple alteration, such as symbolic mapping of digit to letters, the performance drops, implying that models are pattern matching and not performing robust rule induction. NumericBench (Li et al., 2025) highlight the weakness of LLMs like GPT4 and DeepSeek to operate with numbers in real-life settings even when evaluated on simple tasks like number recognition or comparison.

Beyond using datasets to evaluate the entire model pipeline, some research focuses on the internal mechanism of the models. Mechanistic interpretability on one-hand suggest that model's arithmetic computation occurs in higher layers, but validation of answers is carried out by middle layers (Bertolazzi et al., 2025), on the other hand, Mamidanna et al. (2025) find that arithmetic logic appears to crystallise only at the last token. Similarly, (Baeumel et al., 2025) use logit lens to observe that models identify arithmetic task in earlier layers but only produce a correct generation in the final layer. In a parameter analysis, Christ et al. (2025) demonstrate via pruning and scaling that specific parameters are dedicated to mathematical logic.

Despite evidence of internal arithmetic capability, models frequently default to non-robust reasoning where model memorise reasoning steps and unsuccessfully apply them to perturbed problems (Kang et al., 2025). Other work suggests that models possess circuits that learn simple heuristics used to solve arithmetic problem (Nikankin et al., 2025).

2.3 Improving Numerical Outputs

To improve numerical outputs, specialised number-aware objectives have been proposed, including multi-number-token contrastive loss (Petrak et al., 2023), weighted digit embedding loss (Sivakumar and Moosavi, 2025) or a magnitude and edit distance inspired loss (Fei et al., 2025). Alternatively, some research reformulates predicting numbers as a regression task (Wallace et al., 2019; Zausinger et al., 2025). Nonetheless, existing number generation research remains largely focused on data-centric solutions to improve reasoning in the form of pre-training, fine-tuning and reinforcement learning (Chen et al., 2025; Akter et al., 2026; Shao et al., 2024; Zhang et al., 2026; Havrilla et al., 2024). Some works curate datasets to improve specifically on certain domains within numerical reasoning (Mishra et al., 2022; Sivakumar and Moosavi, 2023; Akter et al., 2026) or self-improve by training on errors (Yu et al., 2025). Improvements have also been achieved through reasoning strategies such as chain-of-thought (Wei et al., 2022), self-consistency (Wang et al., 2023), program-of-thought (Chen et al., 2023), and thinking tokens (Goyal et al., 2024). In this work, rather than introducing additional supervision, reasoning trajectories, or external computation, we focus on understanding and exploiting the information already present within the model's predictions.

Many different limitations and diagnostic behaviours have been identified by the community, however, these studies primarily focus on understanding the representations, heuristics, internal computations underlying arithmetic reasoning, and enforcing the output via objective functions. In contrast, we investigate the information available during generation itself and the extent to which numerical errors remain recoverable within the model's predictive space.

3 Experimental Setup

Search Space Extension Standard autoregressive decoding generates outputs by repeatedly selecting tokens from the probability distribution produced by the model's final layer. To expand this predictive space while remaining grounded in the model's internal computations, we consider token predictions derived from both model depth and breadth. For depth, the candidate set consists of the single predictions from every intermediate layer. To map intermediate latent information into token

predictions, we employ an 'early-exit' strategy that extracts the hidden state of an earlier layer, applies a normalisation if necessary, and passes it through a softmax function to project the distribution onto the vocabulary, yielding one candidate token per layer. We also refer to this intervention as multi-layer (ML). For breadth, the candidate set consists of the top-k tokens from the final layer's output distribution. We refer to this intervention as TK, and unless stated otherwise, use $k = 5$.

Oracle Decoding We evaluate depth and breadth independently to isolate their respective contributions. As illustrated in Figure 1, at each generation step, we check whether the gold token is present in the candidate set. If the gold token is absent, reconstruction fails. Otherwise, the gold token is fed back into the model as context for the subsequent prediction step. A prediction is considered recoverable only if the complete target sequence can be reconstructed in this manner.

Metrics To evaluate the accuracy of number generation, we select three metrics to evaluate different granularity of the predictions (see Appendix A.1 for formulae of each metric). String accuracy is a binary metric that compares the gold number and predicted number as float-point values, this is the most realistic metric as the answer is only correct if the number produced is equivalent to the gold answer. For a more forgiving and fine-grained metric, we use digit accuracy which computes a character level accuracy over the length of the gold answer. This provides insight into the number generated and how close it is to the gold number. Lastly, instead of focusing on the digit being predicted, we look at how many digits are predicted, therefore, we use digit mean absolute error (DMAE) to look at the difference in the number of digits predicted and expected.

Dataset To investigate the areas of numerical reasoning that a deeper and broader search space could improve, we consider the following: difficulty of questions, question format, and input numbers. To conduct our analysis on various axes, we choose the FERMAT (Sivakumar and Moosavi, 2023) dataset. We opt for simple arithmetic questions to be able to work with both smaller and larger models. Therefore we filter the dataset to obtain 102 problems for each basic operation (addition, subtraction, multiplication, division). We evaluate both symbolic equations and their worded question counterparts. Lastly, we use six of FERMAT's input number subgroups: 2 digit, 3 digit, 4 digit, 4+ digit, 1 decimal place, and 2 decimal place, these take the same questions but replace the numbers to satisfy their subgroup description giving a total of 102×4×6=2448 symbolic and 2448 worded problems. Examples of different questions from FERMAT are shown in Appendix A.2, Table 3.

We further stratify these evaluation instances by their gold digit count (total number of digits in the ground-truth answer). This mapping spans a distribution of answers ranging from 1 to 12 digits (see Appendix A.3, Table 4), allowing us to examine how model accuracy is directly impacted as numerical length varies.

Models For a comprehensive investigation, we select two open-source model families: Llama-3.2 {1B, 3B}, Llama-3.1 8B (Grattafiori et al., 2024) and Qwen3 {0.6B, 1.7B, 4B, 8B, 14B} (Yang et al., 2025). A key difference between the two families, particularly pertinent to our research, lies in their treatment of numerical tokens. Qwen3 models use digit tokenisation, so they only have the ten number tokens [0] to [9], whereas Llama-3 models use multi-digit tokenisation by expanding the vocabulary to include multi-digit chunks such as {'0', ..., '9', '00', ..., '99', '000', ..., '999'}[2]. This structural breakdown allows us to isolate how digit and multi-digit tokenisation impact number generation.

Numerical Emergence To understand when models begin representing numerical information during generation, we analyse the probability mass assigned to numeric tokens across layers. We define numeric tokens as vocabulary items composed exclusively of the characters {'0', ..., '9', '.', ','} and use their cumulative probability mass as a coarse indicator of numerical focus within the predictive distribution.

Confidence Dynamics Recoverability evaluation captures whether the correct token exists within the predictive space, but not how strongly it is represented. To distinguish near-miss predictions from fundamentally incorrect trajectories, we analyse the probability assigned to both gold and predicted tokens across layers. This allows us to characterise

[2]For instance, predicting 1234 would be in four step ['1', '2', '3', '4'] under digit tokenisation, but in two, ['123', '4'] under the multi-digit approach.

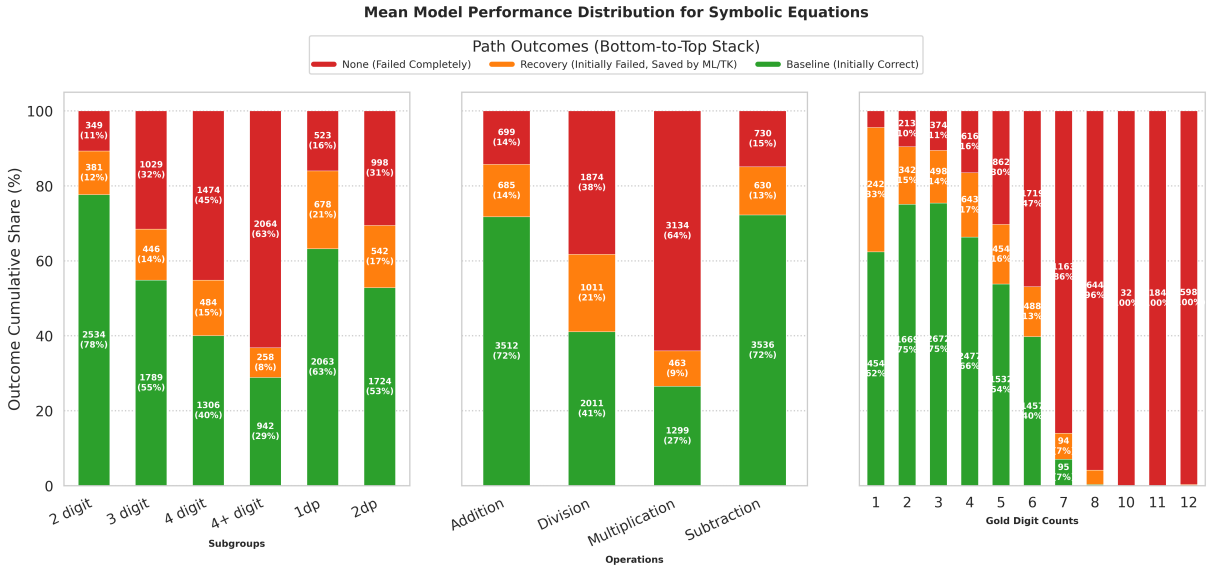


Figure 2: Distributions of all model's average correct generation across different recoverability categories: Baseline (correct under standard decoding), Recovery (recoverable through intermediate-layer predictions or top-5 candidates), and None (not recoverable under either strategy). The distribution share is stratified horizontally by subgroups (left), operations (middle), and gold digit count (right) for symbolic equations with raw counts inside each bar segment.

the competition between correct and incorrect candidates during generation.

Generation Stages Number generation can be broken up into three stages: initiation, continuation, and termination. Initiation corresponds to predicting the first numeric token, continuation to generating subsequent numeric tokens conditioned on an existing numerical prefix, and termination to transitioning from numeric to non-numeric outputs after the answer has been completed. This decomposition allows us to identify which stages benefit most from broader or deeper search spaces.

4 Results

Our analysis addresses three questions: (i) which numerical failures remain recoverable beyond the final prediction, (ii) under what conditions this recoverability disappears, and (iii) whether recoverable failures can be identified from predictive dynamics.

4.1 Recoverable Numerical Failures

To determine whether arithmetic failures originate from computation or generation, we first investigate whether the correct answer remains recoverable beyond the model's final prediction. Table 1 compares baseline decoding against recoverable depth (ML) and breadth (TK) interventions across all evaluated models. Extending the predictive space reveals substantial latent recoverability.

Table 1: Paired Statistical Significance Matrix: Oracle Interventions vs. Base Controls ($N = 2448$, all p-values range from 1.9073×10^{-6} to 1.0977×10^{-154})

Question Format	Model	BASE vs. ML		BASE vs. TK	
		Δ Acc.	McNemar	Δ Acc.	McNemar
Symbolic Equations	Qwen3-0.6B	+3.19%	✔	+28.76%	✔
	Qwen3-1.7B	+2.86%	✔	+15.28%	✔
	Qwen3-4B	+3.23%	✔	+11.40%	✔
	Qwen3-8B	+4.62%	✔	+11.03%	✔
	Qwen3-14B	+3.80%	✔	+13.15%	✔
	Llama-3.2-1B	+3.06%	✔	+13.19%	✔
	Llama-3.2-3B	+6.54%	✔	+10.25%	✔
	Llama-3.1-8B	+6.05%	✔	+6.66%	✔
Worded Questions	Qwen3-0.6B	+0.82%	✔	+8.29%	✔
	Qwen3-1.7B	+1.92%	✔	+7.72%	✔
	Qwen3-4B	+2.29%	✔	+8.54%	✔
	Qwen3-8B	+2.82%	✔	+6.54%	✔
	Qwen3-14B	+5.15%	✔	+9.15%	✔
	Llama-3.2-1B	+3.02%	✔	+12.91%	✔
	Llama-3.2-3B	+4.90%	✔	+7.97%	✔
	Llama-3.1-8B	+6.17%	✔	+8.58%	✔

✔ Statistically Significant ($p < 0.05$). ✗: Statistically Non-Significant ($p \geq 0.05$).

Averaged across all models, TK improves symbolic equations and worded questions by 13.73% and 8.73%, respectively, while ML yields smaller but consistent improvements of 4.18% and 3.39%. Therefore, models frequently produce information that could lead to the correct answer, but this information is not selected by standard decoding.

To verify that these improvements are not driven by random alternatives, we perform paired McNemar tests on string-level accuracy (Table 1). Both ML and TK improvements are statistically significant across all settings, showing that the recovered tokens correspond to meaningful competing predictions rather than noise within the output distribu-

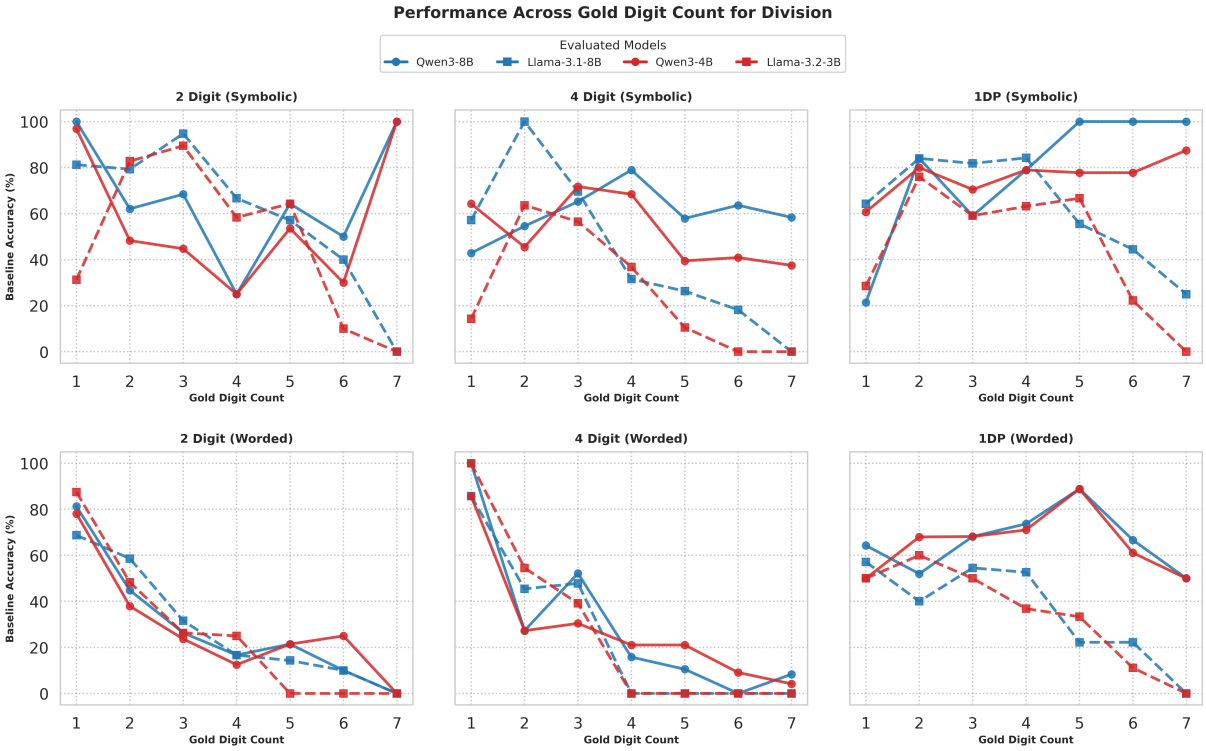


Figure 3: Division accuracy stratified by the number of digits in the gold answer across symbolic problems (top) and worded problems (bottom). Performance is evaluated across three FERMAT subgroups (2 digit, 4 digit, 1dp) comparing Qwen3-8B and Llama-3.1-8B models alongside the Qwen3-4B and Llama-3.2-3B variants.

tion. These findings indicate that, for an average of 30% across all models of our arithmetic problems, the information required to generate the correct answer already exists within the model's predictive space but fails to be prioritised during decoding.

4.2 Limits of Recoverability

Recoverability is not uniformly distributed across arithmetic problems. Figure 2 shows that the largest gains occur for shorter outputs and simpler operations. For two-digit answers, extending the search space improves accuracy by about 11% for both symbolic equations and word problems. Addition and subtraction reach more than 85% on average across all models under oracle decoding.

However, this recoverability rapidly diminishes as arithmetic complexity increases. As answer length grows, the correct answer progressively disappears not only from the model's final prediction but also from intermediate-layer outputs and lower-ranked alternatives, as seen in Figure 2's right plot. Beyond approximately six digits which reaches at least 50% accuracy, the opportunity for decoding-based recovery becomes increasingly limited, suggesting a transition from generation failures to limitations in the model's underlying computation. This transition is particularly pronounced for multiplication, division, and larger operand categories, which dominate the unrecoverable subset across both symbolic equations and word problems.

Nonetheless, when we compare string- and digit-level gains to determine how ML and TK alter numerical predictions. For symbolic equations, ML improves string accuracy by 4.18% but digit accuracy by only 1.34%, whereas TK yields more comparable gains of 13.73% and 12.63%, respectively (as shown in Appendix A.4 in Table 5 and Table 6). This suggests that ML often changes a limited portion of an otherwise plausible answer, while TK can introduce a substantially different numerical continuation. Despite both interventions, the correct numerical trajectory itself is at times missing from the predictive space.

Tokenisation appears to impact where this performance breakpoint occurs. Llama models, which rely on multi-digit numeric tokens, deteriorate sharply once answers span multiple numeric tokens, whereas Qwen's digit tokenisation remains substantially more robust on longer outputs. This suggests that token granularity can significantly influence the accumulation and propagation of numerical errors during generation, as shown in Figure 3

Table 2: Percentage of evaluation instances recovered by top-k (TK) and multi-layer (ML) oracle interventions at each generation stage.

Model	TK Recoveries			ML Recoveries		
	Init.	Cont.	Term.	Init.	Cont.	Term.
Llama-3.1-8B	26.0%	6.3%	0.9%	11.9%	2.7%	0.5%
Llama-3.2-1B	29.8%	6.2%	0.6%	5.4%	1.6%	0.6%
Llama-3.2-3B	26.9%	7.0%	0.6%	13.2%	1.9%	0.5%
Qwen3-0.6B	23.6%	3.3%	0.0%	0.0%	0.0%	0.0%
Qwen3-1.7B	12.6%	2.0%	0.2%	0.0%	1.0%	0.1%
Qwen3-14B	19.6%	3.4%	1.3%	8.1%	1.7%	0.4%
Qwen3-4B	14.5%	3.2%	0.3%	0.0%	0.6%	0.0%
Qwen3-8B	12.8%	4.3%	0.6%	1.7%	1.5%	0.3%

4.3 Predictive Signals of Recovery

Having established that some numerical failures remain recoverable, we next examine how this recoverability manifests within the model's predictions. We consider three complementary signals: the stage of numerical generation at which a correction occurs, the ranks and layers containing useful alternatives, and the evolution of token probabilities across layers and ranks.

Generation Stages To isolate where intervention is most critical, we divide number generation into initiation, continuation, and termination to determine whether recoverable errors stem from initial commitment (which dictates overall trajectory), mid-sequence continuation (where numerical precision is refined), or final termination (where sequence boundaries are set).

Table 2 shows that successful recovery is concentrated primarily at initiation for both TK and ML. This pattern is particularly strong for TK, for which correcting the first numeric token consistently accounts for most recoveries across all models. ML follows the same general pattern when intermediate-layer recovery is available, although its contribution is substantially smaller and varies more across models. Continuation corrections occur less frequently, while termination corrections are rare under both strategies. Nevertheless, the digit mean absolute error (DMAE) decreases by 0.24 and 0.37 digit for ML and TK's intervention respectively meaning that the recoveries impact at least the predicted digit length (see Appendix A.5, Table 7). Overall, these results indicate that most recoverable numerical errors arise from an incorrect initial commitment to a numerical trajectory rather than from failure to terminate an otherwise correct answer.

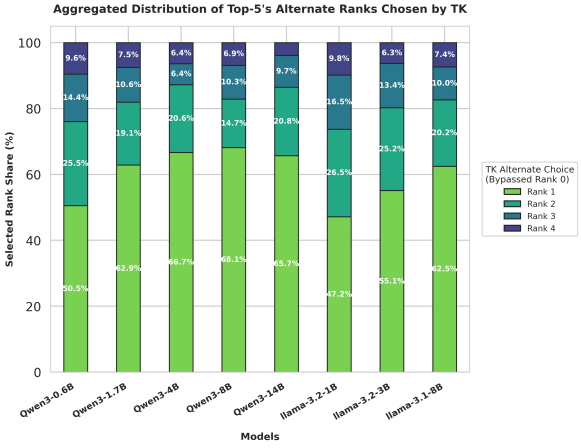


Figure 4: Top-5 rank distribution that is chosen when rank 0 fails for each model.

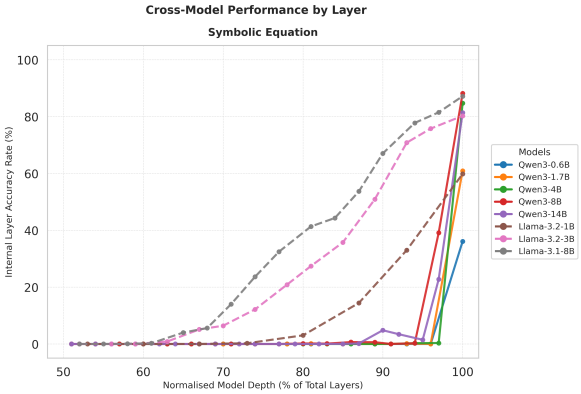


Figure 5: Token-level performance across each layer (represented as a percentage of total layer depth) for all evaluated models on Symbolic Equations.

Informative Ranks and Layers To enable future computationally efficient recovery systems, we investigate which ranks and layers harbour the most salient signal by asking how narrowly we can bound the search space without sacrificing recovery performance both in terms of rank and layers.

For example, although TK substantially expands the search space, useful alternatives are heavily concentrated among the highest-ranked candidates. Figure 4 shows that when the top prediction is incorrect, rank 1 (second best rank) most frequently contains the correct alternative across all evaluated models. Lower-ranked candidates contribute progressively less to recovery, with this concentration becoming stronger as model size increases. Consequently, much of the breadth-based recoverability can be captured using only a small candidate set, for instance considering the top-3 candidates alone would already recovery up to 61.65% on average, therefore, it would already provide most of the

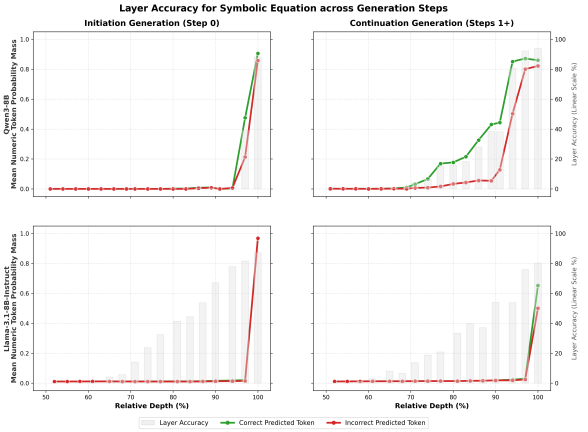


Figure 6: Target token accuracy as grey bars, mean probability mass of numeric tokens separated by generation step (green for correct and red for incorrect predictions), Qwen3-8B (top), Llama-3.1-8B (bottom)

available recoverability benefit (see Appendix A.6, Table 8 for each top-k benefits).

Useful intermediate-layer predictions exhibit a similar concentration across depth. Figure 5 shows that informative predictions emerge almost exclusively within the latter half of the network for all evaluated models. However, the manner in which this information accumulates differs substantially across architectures. Llama models exhibit a gradual increase in recoverability across successive layers, suggesting that arithmetic information develops progressively throughout the computation. In contrast, Qwen models display a much sharper transition near the output, with useful intermediate predictions concentrated almost entirely within the final 15% of layers.

These architectural differences help explain the observed disparity in ML recoverability between model families. Earlier-layer representations in Llama remain informative over a broader portion of the network, whereas Qwen largely commits to numerical predictions only near the final layers. Consequently, depth-based interventions appear substantially more promising for architectures exhibiting gradual representational refinement than for those that defer numerical commitment to the end.

Confidence Signals We examine internal confidence signals to answer a key practical question: Can internal model signals such as numeric probability mass, top-k margin separation, and rank differences forecast numerical failure to possibly trigger recovery?

Figure 6 examines the evolution of numeric-token probability mass across layers for both correct and incorrect trajectories. For initiation prediction, neither model exhibits meaningful separation until the final layers, suggesting that numerical commitment occurs very late in generation. The behaviour differs once a numerical sequence has already been initiated. For Qwen, correct continuation trajectories begin accumulating substantially more numeric probability mass several layers before the final generation, whereas incorrect trajectories remain comparatively flat until close to the output layer. This separation emerges despite intermediate-layer predictions themselves remaining relatively weak, indicating that numeric probability mass may provide a useful confidence signal for post-generation correction or adaptive decoding.

Llama exhibits the opposite behaviour. Intermediate-layer predictions become informative substantially earlier, as reflected by the grey early-exit accuracy bars, but numeric probability mass remains highly compressed until the final layer for both correct and incorrect predictions. Consequently, raw probability mass is considerably less informative for Llama than for Qwen, suggesting that different architectures expose different signals for identifying uncertain numerical predictions.

Figure 7 provides a more direct confidence signal by comparing the probability difference between the top two candidates across layers. For Llama models, correct predictions consistently exhibit larger confidence margins than incorrect predictions throughout the final layers of computation. In contrast, Qwen trajectories remain relatively similar until the final few layers, where correct predictions rapidly become more confident than incorrect ones. When looking at rank differences instead of probabilities, we observe from Figure 8 that correct predictions rank the chosen token higher than incorrect predictions showing that confidence emerges much earlier when considering ranks. All models have a layer where the difference is greater than 58 ranks and the 84% depth shows a consensual point of difference for all models.

These results suggest that confidence margins may provide a useful signal for identifying unreliable numerical predictions and thus triggering a correction. Importantly, the layer at which this signal becomes informative differs substantially across architectures, reflecting their differing numerical generation dynamics.

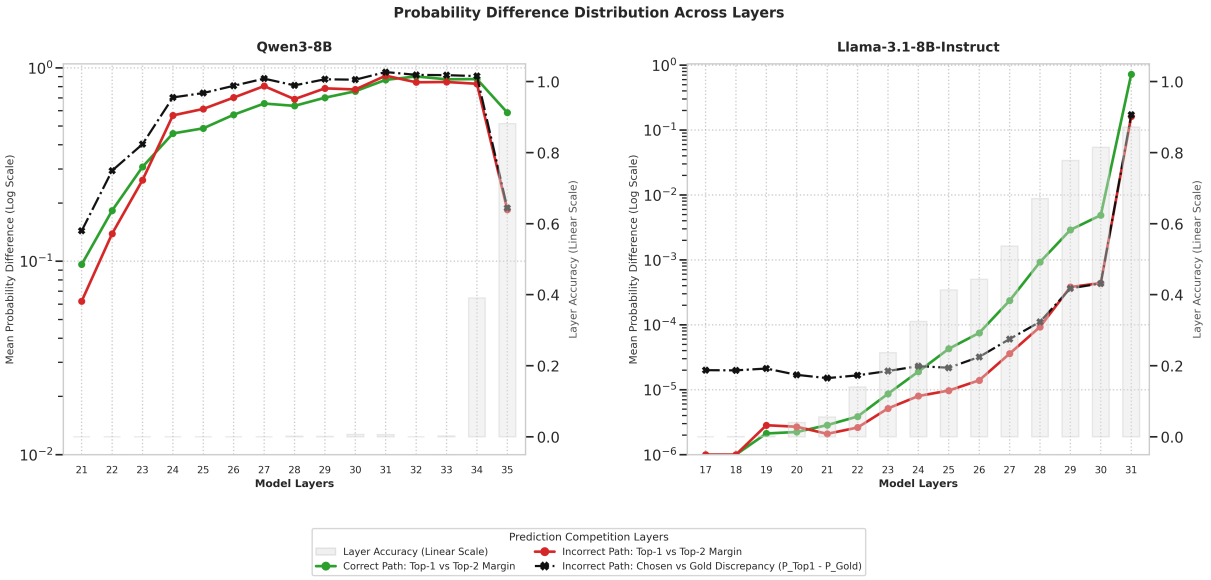


Figure 7: Symbolic Equation results with accuracy by layers in grey bars, probability difference between Top-1 and Top-2 for correct predictions (green), incorrect (red) prediction, and difference between chosen and gold predictions (black).

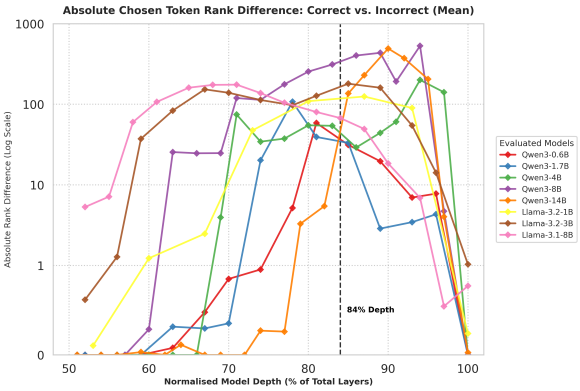


Figure 8: Rank difference between correct and incorrect token prediction for all models across last 50% of layers

5 Conclusion

In this work, we introduced recoverability as a lens for understanding numerical errors in language models. Rather than treating incorrect predictions as a single failure mode, we distinguish between errors that remain recoverable within the model's predictive process and those for which the correct trajectory is no longer accessible within the candidate spaces examined here. Our findings show that recoverability is highly structured. It varies across arithmetic tasks, output lengths, model families, and stages of generation, and exhibits characteristic signatures across candidate ranks, intermediate layers, and confidence dynamics. These observations suggest that numerical failures differ not only in severity but also in the extent to which they remain amenable to decoding-based intervention. We therefore argue that future progress in numerical reasoning may benefit from moving beyond aggregate accuracy improvements towards understanding the recoverability profile of different failure modes and designing interventions targeted specifically at recoverable errors.

Acknowledgments

This work was supported by the Centre for Doctoral Training in Speech and Language Technologies (SLT) and their Applications funded by UK Research and Innovation [grant number EP/S023062/1] and UKRI Impact Pathway funding. We also acknowledge IT Services at The University of Sheffield for the provision of services for High Performance Computing. Additionally, a particular mention goes to Mugdha Pandya for her continued feedback throughout the research.

Limitations

While our findings provide valuable insights into the latent recovery pathways of numbers within LLMs, some constraints bound our experimental setup. In addition to general limitations such as using different model families, testing on a range of datasets, accounting for different number format e.g. standard form notation of number, we highlight some key technical limitations of our work.

First, our evaluation relies heavily on the assumption that the model generates the target numerical completion as the immediate next token sequence (see Appendix A.7). However, smaller language models or those less optimised for strict instruction-following often prepend conversational fillers, introductory phrasing, or alternative structural framing. In these instances, evaluating the absolute position of the first numerical token introduces substantial noise into the pipeline, potentially misclassifying viable generation paths due to alignment failures rather than computational deficits.

Second, we explicitly focus on direct generation setups, omitting the use of structured reasoning mechanisms such as Chain-of-Thought (CoT) prompting or models leveraging dedicated internal thinking tokens. It is probable that explicit multi-step reasoning paradigms already surface and resolve a significant portion of the latent, recoverable errors we capture within our search-space analysis. By bypassing these reasoning frameworks, our reported baseline recovery rates likely overrepresent the practical hidden capacity of these systems compared to environments where explicit sequential reasoning is already deployed.

Finally, the signal dynamics, layer indicators, and rank differences highlighted throughout our analysis reflect macro-level mean trends aggregated across large evaluation batches. When shifting from an aggregate perspective to an individual token level, the operational profiles of correct and incorrect paths might exhibit an overlap. This variance implies that while these internal features remain statistically distinct on average, translating them directly into rigid, inference-time heuristics for dynamic token routing may yield considerably higher false-positive rates than the mean trends imply.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Syeda Nahida Akter, Shrimai Prabhumoye, Eric Nyberg, Mostofa Patwary, Mohammad Shoeybi, Yejin Choi, and Bryan Catanzaro. 2026. Front-loading reasoning: The synergy between pretraining and post-training data. In *The Fourteenth International Conference on Learning Representations*.

Tanja Baeumel, Josef van Genabith, and Simon Ostermann. 2025. Disentangling mathematical reasoning in LLMs: A methodological investigation of internal mechanisms. In *Proceedings of The 3rd Workshop on Mathematical Natural Language Processing (MathNLP 2025)*, pages 203–217, Suzhou, China. Association for Computational Linguistics.

Leonardo Bertolazzi, Philipp Mondorf, Barbara Plank, and Raffaella Bernardi. 2025. The validation gap: A mechanistic analysis of how language models compute arithmetic but fail to validate it. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 29387–29424, Suzhou, China. Association for Computational Linguistics.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. 2023. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *Transactions on Machine Learning Research*.

Zui Chen, Tianqiao Liu, Mi Tian, Weiqi Luo, and Zitao Liu. 2025. Advancing mathematical reasoning in language models: The impact of problem-solving data, data synthesis methods, and training stages. In *International Conference on Learning Representations*, volume 2025, pages 68214–68233.

Stephen Chrisomalis. 2010. *Numerical Notation: A Comparative History*. Cambridge University Press.

Bryan R Christ, Zachary Gottesman, Jonathan Kropko, and Thomas Hartvigsen. 2025. Math neurosurgery: Isolating language models' math reasoning abilities using only forward passes. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 24803–24840, Vienna, Austria. Association for Computational Linguistics.

Yung-Sung Chuang, Yujia Xie, Hongyin Luo, Yoon Kim, James R. Glass, and Pengcheng He. 2024. Dola: Decoding by contrasting layers improves factuality in large language models. In *The Twelfth International Conference on Learning Representations*.

Gautier Dagan, Gabriel Synnaeve, and Baptiste Rozière. 2024. Getting the most out of your tokenizer for pre-training and domain adaptation. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Xiang Fei, Jinghui Lu, Qi Sun, Hao Feng, Yanjie Wang, Wei Shi, An-Lan Wang, Jingqun Tang, and Can Huang. 2025. Advancing sequential numerical prediction in autoregressive models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 562–574, Vienna, Austria. Association for Computational Linguistics.

Tomas Goldsack, Zhihao Zhang, Chenghua Lin, and Carolina Scarton. 2022. Making science simple: Corpora for the lay summarisation of scientific literature.

In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 10589–10604, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Siavash Golkar, Mariel Pettee, Michael Eickenberg, Alberto Bietti, Miles Cranmer, Geraud Krawezik, Francois Lanusse, Michael McCabe, Ruben Ohana, Liam Parker, Bruno Régaldo-Saint Blancard, Tiberiu Tesileanu, Kyunghyun Cho, and Shirley Ho. 2023. xval: A continuous number encoding for large language models. In *NeurIPS 2023 AI for Science Workshop*.

Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Krishna Menon, Sanjiv Kumar, and Vaishnavh Nagarajan. 2024. Think before you speak: Training language models with pause tokens. In *The Twelfth International Conference on Learning Representations*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Denis Guedj. 1996. *L'empire des nombres*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Alexander Havrilla, Yuqing Du, Sharath Chandra Raparthy, Christoforos Nalmpantis, Jane Dwivedi-Yu, Eric Hambro, Sainbayar Sukhbaatar, and Roberta Raileanu. 2024. Teaching large language models to reason with reinforcement learning. In *AI for Math Workshop @ ICML 2024*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

AQ Jiang, A Sablayrolles, A Mensch, C Bamford, DS Chaplot, D de Las Casas, F Bressand, G Lengyel, G Lample, L Saulnier, and 1 others. 2024. Mistral 7b. arxiv 2023. *arXiv preprint arXiv:2310.06825*.

Marek Kadlčík, Michal Štefánik, Timothee Mickus, Josef Kuchař, and Michal Spiegel. 2025. Pre-trained language models learn remarkably accurate representations of numbers. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 26705–26714, Suzhou, China. Association for Computational Linguistics.

Katie Kang, Amrith Setlur, Dibya Ghosh, Jacob Steinhardt, Claire Tomlin, Sergey Levine, and Aviral Kumar. 2025. What do learning dynamics reveal about generalization in LLM mathematical reasoning? In *Forty-second International Conference on Machine Learning*.

Tom Kocmi, Rachel Bawden, Ondřej Bojar, Anton Dvorkovich, Christian Federmann, Mark Fishel, Thamme Gowda, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Rebecca Knowles, Philipp Koehn, Christof Monz, Makoto Morishita, Masaaki Nagata, Toshiaki Nakazawa, Michal Novák, Martin Popel, and Maja Popović. 2022. Findings of the 2022 conference on machine translation (WMT22). In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 1–45, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Haoyang Li, Xuejia Chen, Zhanchao Xu, Darian Li, Nicole Hu, Fei Teng, Yiming Li, Luyu Qiu, Chen Jason Zhang, Li Qing, and Lei Chen. 2025. Exposing numeracy gaps: A benchmark to evaluate fundamental numerical abilities in large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 20004–20026, Vienna, Austria. Association for Computational Linguistics.

Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori Hashimoto, Luke Zettlemoyer, and Mike Lewis. 2023. Contrastive decoding: Open-ended text generation as optimization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12286–12312, Toronto, Canada. Association for Computational Linguistics.

Siddarth Mamidanna, Daking Rai, Ziyu Yao, and Yilun Zhou. 2025. All for one: LLMs solve mental math at the last token with information transferred from other tokens. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 30747–30760, Suzhou, China. Association for Computational Linguistics.

Sean McLeish, Arpit Bansal, Alex Stein, Neel Jain, John Kirchenbauer, Brian R. Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, Jonas Geiping, Avi Schwarzschild, and Tom Goldstein. 2024. Transformers can do arithmetic with the right embeddings. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA. Curran Associates Inc.

Swaroop Mishra, Matthew Finlayson, Pan Lu, Leonard Tang, Sean Welleck, Chitta Baral, Tanmay Rajpurohit, Oyvind Tafjord, Ashish Sabharwal, Peter Clark, and Ashwin Kalyan. 2022. LILA: A unified benchmark for mathematical reasoning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5807–5832, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Nafise Sadat Moosavi, Andreas Rücklé, Dan Roth, and Iryna Gurevych. 2021. Scigen: a dataset for reasoning-aware text generation from scientific tables. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Matteo Muffo, Aldo Cocco, and Enrico Bertino. 2022. Evaluating transformer language models on arithmetic operations using number decomposition. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 291–297, Marseille, France. European Language Resources Association.

Yaniv Nikankin, Anja Reusch, Aaron Mueller, and Yonatan Belinkov. 2025. Arithmetic without algorithms: Language models solve math with a bag of heuristics. In *The Thirteenth International Conference on Learning Representations*.

Dominic Petrak, Nafise Sadat Moosavi, and Iryna Gurevych. 2023. Arithmetic-based pretraining improving numeracy of pretrained language models. In *Proceedings of the 12th Joint Conference on Lexical and Computational Semantics (*SEM 2023)*, pages 477–493, Toronto, Canada. Association for Computational Linguistics.

Eli Schwartz, Leshem Choshen, Joseph Shtok, Sivan Doveh, Leonid Karlinsky, and Assaf Arbelle. 2024. NumeroLogic: Number encoding for enhanced LLMs' numerical reasoning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 206–212, Miami, Florida, USA. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Chufan Shi, Haoran Yang, Deng Cai, Zhisong Zhang, Yifan Wang, Yujiu Yang, and Wai Lam. 2024. A thorough examination of decoding methods in the era of LLMs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8601–8629, Miami, Florida, USA. Association for Computational Linguistics.

Jasivan Sivakumar and Nafise Sadat Moosavi. 2023. FERMAT: An alternative to accuracy for numerical reasoning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15026–15043, Toronto, Canada. Association for Computational Linguistics.

Jasivan Alex Sivakumar and Nafise Sadat Moosavi. 2025. How to leverage digit embeddings to represent numbers? In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 7685–7697, Abu Dhabi, UAE. Association for Computational Linguistics.

Georgios Spithourakis and Sebastian Riedel. 2018. Numeracy for language models: Evaluating and improving their ability to predict numbers. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2104–2115, Melbourne, Australia. Association for Computational Linguistics.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, and 1 others. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

Eric Wallace, Yizhong Wang, Sujian Li, Sameer Singh, and Matt Gardner. 2019. Do NLP models know numbers? probing numeracy in embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5307–5315, Hong Kong, China. Association for Computational Linguistics.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*.

Yang Yan, Yu Lu, Renjun Xu, and Zhenzhong Lan. 2025. Do large language models truly grasp addition? a rule-focused diagnostic using two-integer arithmetic. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 13467–13483, Suzhou, China. Association for Computational Linguistics.

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*.

Erxin Yu, Jing Li, Ming Liao, Qi Zhu, Boyang Xue, Minghui Xu, Baojun Wang, Lanqing Hong, Fei Mi, and Lifeng Shang. 2025. Self-error-instruct: Generalizing from errors for LLMs mathematical reasoning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8504–8519, Vienna, Austria. Association for Computational Linguistics.

Jonas Zausinger, Lars Pennig, Anamarija Kozina, Sean Sdahl, Julian Sikora, Adrian Dendorfer, Timofey

Kuznetsov, Mohamad Hagog, Nina Wiedemann, Kacper Chlodny, Vincent Limbach, Anna Ketteler, Thorben Prein, Vishwa Mohan Singh, Michael Danziger, and Jannis Born. 2025. Regress, don't guess: A regression-like loss on number tokens for language models. In *Forty-second International Conference on Machine Learning*.

Xiaoying Zhang, Hao Sun, Yipeng Zhang, Kaituo Feng, Chaochao Lu, Chao Yang, and Helen M. Meng. 2026. Advancing LLM reasoning with natural language and numerical feedback.

Xikun Zhang, Deepak Ramachandran, Ian Tenney, Yanai Elazar, and Dan Roth. 2020. Do language embeddings capture scales? In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4889–4896, Online. Association for Computational Linguistics.

Tianyi Zhou, Deqing Fu, Mahdi Soltanolkotabi, Robin Jia, and Vatsal Sharan. 2025. Fone: Precise single-token number embeddings via fourier features. *Preprint*, arXiv:2502.09741.

A Appendix

A.1 Metrics

String Accuracy

$$\text{String Acc}(y, \hat{y}) = \begin{cases} 1 & \text{if float}(y) = \text{float}(\hat{y}) \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Example:
For gold $y = 1234$ and predicted $\hat{y} = 125$, $\text{float}(1234) \neq \text{float}(125)$, yielding an accuracy of 0.

Digit Accuracy

$$\text{Digit Acc}(y, \hat{y}) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(y_i = \hat{y}_i) \tag{2}$$

Example:
For gold $y = 1234$ ($N = 4$) and predicted $\hat{y} = 125$, index matches are: $1 = 1$ (match), $2 = 2$ (match), $3 \neq 5$ (mismatch), $4 \neq \varnothing$ (mismatch). Accuracy is $\frac{2}{4} = 0.50$.
Note that this metric looks at the string and not the float, therefore it may produce a lower accuracy even if the string accuracy is 1.

Digit Mean Absolute Error (DMAE)

$$\text{DMAE} = \frac{1}{D} \sum_{j=1}^{D} |L(y^{(j)}) - L(\hat{y}^{(j)})| \tag{3}$$

Example:
For gold $y = 1234$, ($L(y) = 4$) and predicted $\hat{y} = 125$, ($L(\hat{y}) = 3$), the single-instance absolute error is $|4 - 3| = 1$.

A.2 FERMAT Examples

Type	Symbolic	Worded
2 digit	12×98	A euro is 12 yen. How much is 98 yen?
3 digit	123×987	A euro is 123 yen. How much is 456 yen?
4 digit	1234×9876	A euro is 1234 yen. How much is 9876 yen?
4+ digit	12345×987654	A euro is 12345 yen. How much is 987654 yen?
1 decimal place	1.2×9.8	A euro is 1.2 yen. How much is 9.8 yen?
2 decimal place	1.23×98.76	A euro is 1.23 yen. How much is 98.76 yen?

Table 3: Examples of Symbolic and Worded Mathematical Problems from FERMAT

A.3 Output Digit Distribution

Table 4: Distribution of Dataset Evaluation Problems Across Gold Answer Digit Counts

Digit Count	Problems	Digit Count	Problems
1	91	**7**	169
2	278	**8**	84
3	443	**9**	0
4	467	**10**	4
5	356	**11**	23
6	458	**12**	75
Total Problems			**2448**

A.4 Overall Results

Table 5: Evaluation on the FERMAT Symbolic Equations

Model	Subgroup	Baseline			Multi-Layer (ML)			Top-5 (TK)		
		Str	**Dig**	**Tok**	**Str**	**Dig**	**Tok**	**Str**	**Dig**	**Tok**
Qwen3-0.6B	2 digit	28.9%	38.4%	36.3%	32.4%	38.4%	36.3%	73.5%	76.6%	75.2%
	3 digit	13.0%	16.3%	16.3%	14.5%	16.3%	16.3%	47.1%	55.5%	57.3%
	4 digit	2.7%	6.5%	6.6%	3.7%	6.5%	6.6%	33.8%	44.5%	46.2%
	4+ digit	0.2%	1.6%	1.7%	0.2%	1.6%	1.7%	9.3%	24.6%	26.3%
	1dp	40.4%	48.1%	48.2%	48.5%	48.3%	48.6%	71.1%	71.0%	71.3%
	2dp	19.9%	29.7%	29.9%	25.0%	29.8%	30.0%	42.9%	51.9%	52.0%
	GLOBAL MEAN	**17.5%**	**19.4%**	**20.3%**	**20.7%**	**19.5%**	**20.4%**	**46.3%**	**48.8%**	**50.4%**
Qwen3-1.7B	2 digit	78.7%	76.7%	73.6%	79.4%	77.1%	74.1%	84.3%	82.3%	79.7%
	3 digit	52.0%	54.6%	54.6%	52.5%	55.1%	55.2%	65.4%	67.7%	68.3%
	4 digit	33.3%	35.9%	35.7%	34.6%	36.0%	35.9%	48.3%	50.8%	51.7%
	4+ digit	6.6%	10.2%	10.7%	7.4%	10.7%	11.2%	19.9%	29.7%	30.9%
	1dp	53.7%	54.6%	55.2%	62.5%	55.3%	56.0%	77.5%	73.4%	73.8%
	2dp	37.5%	45.4%	45.0%	42.6%	45.8%	45.5%	58.1%	60.5%	61.0%
	GLOBAL MEAN	**43.6%**	**39.9%**	**40.4%**	**46.5%**	**40.4%**	**40.9%**	**58.9%**	**55.4%**	**56.6%**
Qwen3-4B	2 digit	85.8%	84.3%	80.9%	86.0%	84.5%	81.1%	90.9%	88.5%	86.6%
	3 digit	62.5%	63.6%	63.2%	63.7%	64.3%	63.8%	72.8%	75.7%	75.2%
	4 digit	55.9%	57.8%	57.8%	57.8%	58.2%	58.2%	67.4%	71.6%	72.1%
	4+ digit	44.6%	49.3%	49.1%	44.9%	49.2%	49.3%	49.8%	60.5%	60.5%
	1dp	71.3%	80.8%	80.6%	81.6%	81.0%	81.0%	89.5%	89.9%	89.8%
	2dp	64.7%	75.3%	75.3%	70.1%	76.4%	76.5%	82.8%	88.0%	87.7%
	GLOBAL MEAN	**64.1%**	**65.1%**	**65.3%**	**67.4%**	**65.5%**	**65.9%**	**75.5%**	**76.2%**	**76.6%**
Qwen3-8B	2 digit	88.7%	86.9%	84.7%	89.2%	87.1%	85.0%	92.9%	91.0%	89.6%
	3 digit	70.8%	70.4%	70.8%	72.5%	71.5%	72.4%	78.9%	80.5%	81.3%
	4 digit	60.5%	61.3%	62.8%	64.2%	62.6%	64.2%	73.3%	76.1%	77.2%
	4+ digit	47.1%	52.8%	53.4%	48.3%	53.9%	54.4%	52.0%	63.0%	63.8%
	1dp	75.0%	84.3%	84.4%	88.5%	85.1%	85.4%	94.9%	94.7%	94.4%
	2dp	71.3%	78.8%	79.2%	78.4%	80.2%	80.9%	87.7%	89.9%	90.4%
	GLOBAL MEAN	**68.9%**	**69.0%**	**70.0%**	**73.5%**	**70.1%**	**71.2%**	**79.9%**	**79.6%**	**80.7%**
Qwen3-14B	2 digit	85.0%	81.6%	79.1%	86.3%	82.6%	80.2%	91.9%	89.3%	87.5%
	3 digit	65.2%	60.9%	61.0%	69.4%	64.3%	64.7%	78.9%	79.6%	79.4%
	4 digit	55.4%	50.5%	52.1%	57.6%	52.0%	53.5%	74.0%	73.4%	74.4%
	4+ digit	43.6%	50.9%	51.8%	45.6%	51.8%	52.8%	55.4%	64.8%	65.5%
	1dp	75.7%	80.1%	80.3%	84.8%	80.7%	81.0%	93.1%	91.9%	91.7%
	2dp	81.4%	84.7%	85.2%	85.5%	85.8%	86.3%	91.9%	92.5%	92.9%
	GLOBAL MEAN	**67.7%**	**65.1%**	**66.4%**	**71.5%**	**66.5%**	**67.8%**	**80.9%**	**79.4%**	**80.2%**
Llama-3.2-1B-Instruct	2 digit	81.1%	79.5%	71.8%	83.6%	82.4%	76.7%	91.7%	90.3%	88.4%
	3 digit	40.4%	49.3%	36.5%	40.7%	49.6%	37.5%	54.2%	61.4%	56.7%
	4 digit	16.4%	33.7%	22.3%	16.4%	34.1%	22.6%	26.2%	42.9%	36.2%
	4+ digit	14.5%	28.5%	22.9%	14.7%	29.0%	23.8%	22.8%	37.2%	34.5%
	1dp	47.8%	65.6%	56.3%	58.3%	67.6%	58.9%	70.3%	75.7%	70.2%
	2dp	37.7%	56.1%	50.0%	42.6%	57.8%	52.1%	52.0%	66.0%	63.1%
	GLOBAL MEAN	**39.7%**	**47.3%**	**42.2%**	**42.7%**	**48.4%**	**44.0%**	**52.9%**	**57.2%**	**56.6%**
Llama-3.2-3B-Instruct	2 digit	84.6%	86.5%	81.1%	89.7%	88.9%	84.2%	92.9%	91.7%	89.1%
	3 digit	65.4%	71.4%	68.7%	68.6%	73.1%	71.4%	73.5%	78.8%	79.0%
	4 digit	43.1%	54.9%	51.4%	48.0%	58.6%	55.4%	52.2%	60.8%	58.8%
	4+ digit	33.8%	44.0%	41.8%	36.0%	46.9%	45.1%	38.0%	50.1%	49.5%
	1dp	65.0%	80.6%	77.5%	80.6%	83.0%	81.1%	83.1%	86.5%	84.5%
	2dp	50.0%	67.2%	63.8%	58.1%	71.2%	69.0%	63.7%	74.8%	73.7%
	GLOBAL MEAN	**57.0%**	**63.1%**	**62.7%**	**63.5%**	**66.0%**	**66.5%**	**67.2%**	**69.5%**	**71.0%**
Llama-3.1-8B-Instruct	2 digit	88.2%	87.9%	83.1%	91.7%	90.5%	86.6%	93.9%	93.0%	90.7%
	3 digit	69.1%	75.0%	74.2%	72.5%	78.1%	78.0%	72.8%	78.7%	78.6%
	4 digit	52.7%	62.0%	57.2%	58.1%	65.3%	63.6%	58.8%	64.8%	60.5%
	4+ digit	40.4%	50.6%	47.4%	43.9%	53.8%	52.3%	44.4%	54.9%	53.2%
	1dp	76.7%	86.5%	84.3%	89.2%	90.1%	89.2%	88.2%	89.6%	88.2%
	2dp	60.0%	74.1%	70.8%	68.1%	77.6%	76.7%	69.1%	78.0%	75.9%
	GLOBAL MEAN	**64.5%**	**68.8%**	**68.5%**	**70.6%**	**72.0%**	**73.5%**	**71.2%**	**72.6%**	**73.4%**

Table 6: Evaluation on the FERMAT Worded Questions

Model	Subgroup	Baseline			Multi-Layer (ML)			Top-5 (TK)		
		Str	**Dig**	**Tok**	**Str**	**Dig**	**Tok**	**Str**	**Dig**	**Tok**
Qwen3-0.6B	2 digit	7.6%	7.7%	7.2%	9.3%	7.7%	7.2%	24.3%	25.7%	25.0%
	3 digit	2.5%	5.1%	3.9%	3.2%	5.1%	3.9%	12.7%	15.9%	15.8%
	4 digit	0.2%	2.5%	1.8%	0.2%	2.4%	1.8%	6.1%	12.4%	11.7%
	4+ digit	0.0%	1.8%	1.5%	0.0%	1.8%	1.5%	1.5%	7.8%	7.4%
	1dp	8.1%	22.9%	22.3%	9.8%	22.8%	22.3%	19.6%	37.0%	35.5%
	2dp	2.2%	15.6%	14.9%	2.9%	15.6%	14.9%	6.1%	24.7%	23.9%
	GLOBAL MEAN	**3.4%**	**8.2%**	**8.1%**	**4.2%**	**8.2%**	**8.1%**	**11.7%**	**18.5%**	**18.4%**
Qwen3-1.7B	2 digit	43.4%	44.0%	42.6%	44.4%	44.2%	42.7%	53.2%	52.4%	50.8%
	3 digit	24.8%	25.7%	25.2%	25.0%	25.7%	25.3%	31.6%	33.3%	32.5%
	4 digit	17.2%	19.0%	17.8%	18.1%	19.0%	18.1%	23.5%	27.0%	25.7%
	4+ digit	10.5%	14.2%	13.4%	11.5%	14.5%	13.7%	14.7%	20.5%	19.8%
	1dp	46.3%	61.6%	59.4%	52.7%	62.6%	61.1%	58.1%	68.6%	67.0%
	2dp	28.9%	46.5%	45.8%	30.9%	46.6%	46.1%	36.3%	52.3%	51.7%
	GLOBAL MEAN	**28.5%**	**31.7%**	**31.7%**	**30.4%**	**31.9%**	**32.1%**	**36.2%**	**38.7%**	**38.7%**
Qwen3-4B	2 digit	52.5%	54.5%	52.3%	53.4%	54.8%	52.5%	63.0%	64.7%	63.5%
	3 digit	42.4%	46.3%	44.8%	43.4%	46.2%	44.9%	53.2%	56.8%	55.6%
	4 digit	37.7%	41.2%	39.7%	38.7%	41.5%	40.1%	47.3%	51.2%	50.4%
	4+ digit	34.8%	37.4%	35.0%	35.0%	37.4%	35.2%	36.0%	40.6%	39.1%
	1dp	57.4%	69.9%	69.4%	63.7%	70.9%	70.5%	68.4%	76.4%	75.9%
	2dp	46.6%	62.7%	61.6%	50.7%	63.9%	63.1%	54.7%	68.1%	67.6%
	GLOBAL MEAN	**45.2%**	**49.8%**	**49.0%**	**47.5%**	**50.2%**	**49.6%**	**53.8%**	**56.8%**	**56.6%**
Qwen3-8B	2 digit	76.7%	74.8%	69.8%	77.9%	75.9%	71.1%	80.6%	78.4%	74.9%
	3 digit	53.4%	57.4%	54.1%	55.1%	58.0%	55.5%	61.3%	64.5%	62.6%
	4 digit	48.5%	50.2%	47.9%	50.2%	51.8%	49.9%	55.4%	58.3%	56.7%
	4+ digit	39.5%	42.8%	41.9%	40.7%	44.3%	43.3%	42.4%	49.2%	48.5%
	1dp	69.1%	78.0%	76.9%	74.8%	79.0%	78.2%	77.9%	81.5%	80.6%
	2dp	58.8%	69.1%	69.4%	64.2%	71.8%	72.1%	67.6%	77.2%	76.9%
	GLOBAL MEAN	**57.7%**	**58.8%**	**57.8%**	**60.5%**	**60.2%**	**59.5%**	**64.2%**	**65.2%**	**64.6%**
Qwen3-14B	2 digit	73.5%	74.3%	69.5%	76.0%	75.8%	71.5%	82.4%	82.1%	79.0%
	3 digit	50.7%	56.0%	52.3%	55.6%	60.0%	57.4%	62.7%	66.1%	63.0%
	4 digit	38.0%	41.7%	39.9%	41.9%	44.9%	44.2%	49.3%	55.1%	53.7%
	4+ digit	32.6%	35.2%	33.8%	40.4%	42.4%	42.0%	41.2%	45.6%	45.4%
	1dp	78.4%	85.4%	84.5%	85.5%	87.3%	87.2%	86.8%	89.1%	88.5%
	2dp	74.8%	81.1%	81.7%	79.4%	83.7%	84.3%	80.6%	85.1%	85.7%
	GLOBAL MEAN	**58.0%**	**58.0%**	**57.5%**	**63.2%**	**61.9%**	**62.1%**	**67.2%**	**66.6%**	**66.6%**
Llama-3.2-1B-Instruct	2 digit	63.0%	62.4%	52.5%	65.4%	64.3%	55.8%	71.1%	69.6%	62.8%
	3 digit	40.0%	42.1%	36.1%	42.2%	43.5%	39.3%	48.5%	51.5%	48.3%
	4 digit	10.3%	23.8%	14.4%	11.5%	24.7%	17.4%	24.8%	34.4%	29.2%
	4+ digit	8.3%	17.5%	12.6%	8.8%	18.0%	13.7%	16.4%	25.2%	21.6%
	1dp	25.5%	45.5%	35.0%	34.6%	48.8%	39.3%	47.1%	58.4%	51.9%
	2dp	18.9%	35.5%	27.4%	21.6%	37.5%	30.5%	35.5%	49.7%	46.0%
	GLOBAL MEAN	**27.7%**	**33.5%**	**27.7%**	**30.7%**	**35.0%**	**30.7%**	**40.6%**	**43.8%**	**41.8%**
Llama-3.2-3B-Instruct	2 digit	70.6%	67.5%	58.9%	73.8%	71.3%	63.8%	77.7%	76.0%	70.1%
	3 digit	57.1%	57.7%	52.3%	58.3%	59.4%	56.1%	59.6%	64.3%	61.1%
	4 digit	42.2%	47.9%	41.2%	45.1%	50.4%	44.6%	47.1%	53.5%	49.0%
	4+ digit	29.4%	35.1%	30.1%	31.6%	37.5%	32.7%	35.3%	41.0%	38.4%
	1dp	49.3%	66.6%	59.9%	61.5%	71.0%	65.4%	65.2%	74.4%	70.2%
	2dp	38.5%	56.3%	51.6%	46.1%	61.6%	58.3%	50.0%	64.4%	62.2%
	GLOBAL MEAN	**47.8%**	**51.8%**	**48.3%**	**52.7%**	**55.0%**	**52.8%**	**55.8%**	**58.6%**	**57.7%**
Llama-3.1-8B-Instruct	2 digit	73.0%	73.9%	63.4%	78.2%	78.1%	69.6%	81.6%	81.3%	73.0%
	3 digit	58.1%	63.3%	57.5%	60.3%	65.2%	60.9%	63.0%	68.2%	65.4%
	4 digit	44.6%	52.8%	46.0%	50.2%	55.8%	50.2%	50.5%	57.1%	51.0%
	4+ digit	35.5%	42.8%	35.2%	40.0%	45.4%	38.2%	42.2%	49.1%	44.4%
	1dp	60.0%	74.9%	68.8%	72.5%	78.3%	73.7%	76.0%	81.8%	78.2%
	2dp	49.3%	62.6%	58.7%	56.4%	66.1%	63.8%	58.8%	69.9%	67.5%
	GLOBAL MEAN	**53.4%**	**58.4%**	**54.4%**	**59.6%**	**61.4%**	**58.9%**	**62.0%**	**64.5%**	**62.8%**

A.5 DMAE Results

Table 7: Digit Mean Absolute Error for all models for Baseline (Base), intermediate layer (ML) and top-k rank (TK) recoveries.

Model	Base	ML	TK	Δ Base ML	Δ Base TK
Qwen3 0.6B	1.3049	1.3519	1.0537	+0.0470	-0.2512
Qwen3 1.7B	1.1179	0.8585	0.7194	-0.2594	-0.3985
Qwen3 4B	0.8815	0.6221	0.5004	-0.2594	-0.3811
Qwen3 8B	0.8248	0.4524	0.3656	-0.3724	-0.4592
Qwen3 14B	0.8721	0.5053	0.3525	-0.3668	-0.5196
Llama-3.2 1B	1.2382	1.1616	1.0061	-0.0766	-0.2321
Llama-3.2 3B	1.1362	0.6981	0.6481	-0.4381	0-.4881
Llama-3.1 8B	0.8852	0.6530	0.6530	-0.2322	-0.2322
Average	**1.0326**	**0.7879**	**0.6624**	**-0.2447**	**-0.3703**

A.6 Top Rank Benefit

Table 8: Top-k Performance and Absolute Improvement (Δ) Over Top-1 Baseline Across Models

Model Name	Top-1 %	Top-2 % ($+\Delta$)	Top-3 % ($+\Delta$)	Top-4 % ($+\Delta$)	Top-5 % ($+\Delta$)
Qwen3-0.6B	17.52%	36.85% (+19.32%)	41.22% (+23.69%)	42.52% (+25.00%)	43.26% (+25.74%)
Qwen3-1.7B	43.63%	54.29% (+10.66%)	56.17% (+12.54%)	56.54% (+12.91%)	56.70% (+13.07%)
Qwen3-4B	64.13%	71.53% (+7.39%)	72.67% (+8.54%)	72.88% (+8.74%)	72.96% (+8.82%)
Qwen3-8B	68.91%	75.61% (+6.70%)	76.27% (+7.35%)	76.43% (+7.52%)	76.55% (+7.64%)
Qwen3-14B	67.73%	77.70% (+9.97%)	78.27% (+10.54%)	78.43% (+10.70%)	78.55% (+10.83%)
llama-3.2-1B	39.67%	47.63% (+7.97%)	49.67% (+10.01%)	50.53% (+10.87%)	51.02% (+11.36%)
llama-3.2-3B	56.99%	61.81% (+4.82%)	62.70% (+5.72%)	63.15% (+6.17%)	63.44% (+6.45%)
llama-3.1-8B	64.54%	67.81% (+3.27%)	68.30% (+3.76%)	68.55% (+4.00%)	68.63% (+4.08%)
AVERAGE	**52.89%**	**61.65% (+8.76%)**	**63.16% (+10.27%)**	**63.63% (+10.74%)**	**63.89% (+11.00%)**

A.7 Prompt Templates and Input Formatting

Table 9: Prompt Formatting by Input Type

Input Type	Preprocessing	Templates	Examples
Worded Questions	Strips leading/trailing whitespace.	`Give the final answer only. {Question} Answer:`	**Input:** `A euro is 12 yen. How much is 98 yen?` **Prompt:** `Give the final answer only. A euro is 12 yen. How much is 98 yen? Answer:`
Symbolic Equation	Strips outer wrapping parentheses and leading/trailing whitespace.	`Give the final answer only. {Equation} =`	**Input:** `12 × 98` **Prompt:** `Give the final answer only. 12 × 98 =`