数理推論タスクにおける学習・推論パイプラインの構築

本郷颯人[1]　加地翔太[2]　小谷真士[3]　嶋中雄大[4]
宮川裕貴[5]　中島悠樹[1]　安孫子リク[1]　松田公慶[6]
[1]東京大学　[2]独立研究者　[3]大阪公立大学
[4]芝浦工業大学　[5]中京大学　[6]筑波大学
本郷：hongo-hayato977g@g.ecc.u-tokyo.ac.jp　加地：shotak.5207@gmail.com
小谷：s18574s18574@gmail.com　嶋中：al23088@shibaura-it.ac.jp
宮川：t32406m@m.chukyo-u.ac.jp　中島：n_yuki0817@outlook.jp
安孫子：astronaut017@g.ecc.u-tokyo.ac.jp　松田：kokei0418@gmail.com

概要

我々のチームでは，「第 2 回大規模言語モデルのファインチューニング技術と評価（FT-LLM 2026）」チューニングコンペティションの数学タスクにおける取り組みを報告する． 競技で与えられた 8B ベースモデルを出発点とし，32 層から 48 層への深さ拡張による 12B 級モデル化，長文 CoT によるフルパラメータ SFT，および GRPO による強化学習を組み合わせて数理推論能力の向上を図った．主力の CoT データセットに対しては，大規模モデルによる自動審査とルールベース整形を組み合わせてデータキュレーションを行い，最終提出系では多数決と文字列正規化を用いた推論パイプラインを採用した．開発用評価データセットでは，長文 SFT と GRPO によって単一推論の精度が大きく向上し，多数決によるテストタイムスケーリングが追加の改善をもたらした．

1　はじめに

数理推論タスクでは，中間推論の破綻や計算ミスが最終解答に直結しやすく，大規模言語モデル（LLM）にとって依然として難易度の高い領域である．そのため，良質な Chain of Thought（CoT）データによる教師ありファインチューニング（SFT）と，推論過程をさらに最適化する強化学習の両方が重要となる．

本稿では，「第 2 回大規模言語モデルのファインチューニング技術と評価（FT-LLM 2026）」チューニングコンペティションの数学タスクに対する我々の取り組みを報告する．対象は，運営から提供された 80 億パラメータのベースモデル `LLM-jp-4-instruct` であり，日本語で出題される中学・高校数学問題に対する推論能力の向上を目指した．

我々は，主力 CoT データセットに対する自動審査とルールベース整形，12B 級モデルへの層拡張を伴う長文 SFT，GRPO（Group Relative Policy Optimization）による正答報酬ベースの最適化，および多数決を用いた推論システムを段階的に構築した．最終提出系では，多数決と文字列正規化を組み合わせたシンプルな推論パイプラインを採用し，補助的な試行は別途分析に留めた．

図 1: 提案手法の全体像

提案手法は，データ，学習，推論の各段階を一貫して最適化する構成であり，その全体像を 図 1 に示す．

2　関連研究

数理推論においては，中間的な推論過程を明示する Chain of Thought（CoT）プロンプティング [1] と，その推論能力を小規模モデルへ移植する教師ありファインチューニング（SFT）が基盤的な手法として広く用いられている．特に，OpenMathInstruct-1 [2] のような大規模合成データセットは，数学タスク向け SFT の有効性を押し上げた代表例である．我々も同様に，主力 CoT データセットの品質を重視した選別を行った．

SFT の後段では，推論過程そのものを最適化するために強化学習（RL）が用いられる．DeepSeekMath で導入された GRPO（Group Relative Policy Optimization）[3] は，同一プロンプトから生成された複数の出力群に対して相対報酬を z スコア標準化することで，価値モデルを用いずに安定した最

適化を行える点で実用性が高い．数学のように正解判定が可能なタスクではRLVR [4] と相性が良く，本稿でもこの枠組みを採用して推論ステップの最適化を図った．なお，我々の設定では相対報酬の正規化に標準偏差で割る方式ではなく，平均中心化のみを用いた．また，推論時には複数サンプルを集約する多数決（Self-Consistency）[5] が有効であり，本稿の提出系でもこの方針を採用した．

3 数理推論に向けた学習パイプライン（提案手法 1）

本章では，競技で与えられた 80 億パラメータの LLM に対し，高度な数理推論能力を付与するための学習パイプラインについて述べる．我々の方針は，主力 CoT データセットの品質改善，長文コンテキストに対応した SFT，および GRPO による推論最適化を順に積み上げるものである．

3.1 データセットの構築とフィルタリング

LLM に複雑な数理推論の過程（Chain of Thought; CoT）を学習させるため，大規模な合成データセットを収集し，厳密な選別を行った．主力として採用したのは，約 560 万件規模の大規模 CoT データセットと約 175 万件および約 20 万件規模の補助 CoT データセットである．別途 TIR 用データセットも整備したが，最終提出モデルは CoT データセットのみで訓練された．

主力 CoT データセットでは，論理的飛躍や計算ミス，不自然な出力を抑えるために，OpenAI の GPT-OSS-20B [6] による自動審査とルールベースのクリーニングを組み合わせた．具体的には，別モデルによる自動判定で誤答サンプルを除外し，正規表現ベースの整形で不要な生成タグ，不自然な最終回答，回答位置の揺れを削除した．これにより，思考過程と最終解答の対応が明確で，一貫した形式の学習データを構築した．

この段階を独立して重視した理由は，後段の SFT や GRPO では，長い推論系列の中に含まれるノイズがそのまま学習されやすいためである．短い応答では見過ごされる小さな誤りでも，長文 CoT では中盤以降の推論を大きく崩すことがある．そのため，我々はまず学習データの品質を安定化させ，モデルが「何を考え，どこで最終解答を出すか」を一貫した形で学べる状態を作ることを優先した．

3.2 教師ありファインチューニング（SFT）

構築した高品質データセットを用い，ベースモデルに対する SFT を実施した．最終的に比較した長文 SFT モデルでは，ベースモデルを 12B 規模へ層拡張した上で，長文 CoT データに対するフルパラメータ SFT を行った．深さ拡張では，Progressive Depth Up-Scaling via Optimal Transport [7] に基づき，前半 16 層を保持しつつ元の後半層を再配置し，新規層を OT 補間で初期化する形で 48 層へ拡張した．

ここでの設計は，容量確保と推論保持を分けて考える点にある．層拡張による 12B 化は，数理推論に必要な表現容量を確保する役割を担い，長文 SFT は，途中式や場合分けを含む長い推論を最後まで保持させる役割を担う．数学問題では，終盤で初めて制約整理や最終変形が現れることも多く，途中で生成が途切れるだけで最終解答の誤りに直結するため，長文コンテキストは単なる余裕ではなく性能上の前提条件であった．

また，SFT と後段の GRPO のあいだでは，合成データセットの出力形式に沿うよう，SFT 側で用いた chat template / system prompt に GRPO 側の設定を合わせた．これは見た目の統一だけが目的ではなく，思考過程の書き出し方と最終回答の置き方を各学習段階で揃え，SFT で学んだ推論骨格を RL 段階へ素直に接続するためである．詳細な設定は付録にまとめた．

3.3 強化学習（GRPO）による最適化

SFT によって基礎的な解答能力を獲得したモデルに対し，推論ステップのさらなる最適化を目的として GRPO を適用した．最終的な GRPO モデルは，長文 SFT モデルを初期ポリシーとして，Tulu 3 系の Open Instruct post-training stack [8] をベースに，公式 `open-instruct` 実装 [9] の `grpo_fast.py` を中心とする強化学習フレームワーク上で学習したものである．

GRPO の報酬関数は，Math-Verify [10] を用いて最終解答の正しさを判定する正答報酬のみから構成した．実運用では，Math-Verify に加えて LaTeX 空白や `\pm` 表記を整える独自のルールベース後処理も組み合わせた．数式タスクでは，最終解が正しいかどうかを比較的明確に判定できるため，複雑な補助報酬を足すよりも，最終的な正答と直接結びついた信号で推論を洗練する方が設計として一貫している．SFT が「解き方の骨格」を与える段階だとすれば，GRPO はその骨格の中から正答へ到達しやすい推論経路を残し，見た目は整っていても誤答へ落ちる軌道を減らす段階と位置づけられる．

学習では，12B モデルを安定して roll-out できるようバッチ構成と vLLM 側の推論設定を調整し，相対報酬の扱いには平均中心化を採用した．また，学習の進行に伴って生成長が増えるため，前半では中規模 CoT データセット，後半では大規模 CoT データセットへ切り替えるカリキュラム的運用を行った．さらに，学習終盤では Polaris [11] の知見を参考に温度 sweep を行った．詳細な実行条件と温度スケジュールは付録に示す．

以上の学習パイプラインによって，まず単一推論の質を底上げし，多数決に頼る前段階で安定した解答候補を生成できるようにした．次章では，この単一推論の改善を前提として，推論時に複数候補を集約することでさらに正答率を押し上げる提出系について述べる．

4 推論・評価システムの実装（提案手法 2）

我々のチームでは，学習フェーズでの性能向上に加え，推論フェーズにおける計算資源のスケーリングを通じて最終的な正答率を高めるシステムを構築した. 提出系では, サンプリングによる多数決（Majority Voting / Self-Consistency）を中核とする推論パイプラインを実装した.

4.1 多数決（Majority Voting）による解答抽出

テストタイムスケーリングの恩恵を引き出すため，提出系では単一の推論パスではなく，多数決アルゴリズムを採用した．処理の流れは単純であり，同一の問題に対して複数の候補解答を生成し，それらを文字列正規化で整形したのち，最も多く支持された答えを最終出力とする．この順序を明示的に分けることで，生成の多様性と集約の安定性を両立させた.

多数決を採った理由は，数学タスクでは単発の計算ミスや場合分けの取り違えが最終解答に直結しやすく，単一サンプルだけではその揺れを吸収しにくいためである．学習段階で単一推論の質を改善しても，生成時には局所的な失敗がなお残るため，複数候補を集約して安定した答えを選ぶ構成が有効であった．代表的な推論条件の詳細は付録にまとめたが，100 問に対する `cons@40` 推論は約 960 秒で完了した．コンペティションでは 30,000 秒という極めて長い制約だったため, 最終提出は `cons@160` で行った．実行時間の観点から，後述するモデル間の性能比較は `cons@40` で行った点に注意する.

多数決を機能させる上で重要だったのは，数式表記の揺れを吸収する後処理である．数学的に等価な解答であっても, 空白, 括弧の付け方, `\pm` 展開の違いによって別解答として集計されると，投票結果が不要に分散してしまう．そこで提出系では文字列正規化を導入し,「複数生成 → 正規化 → 集約」という順序を明示することで，多数決の効果を安定して引き出した.

最終提出でこの構成を選んだのは，性能向上だけでなく実装安定性の観点からも妥当だったためである．ツール統合推論や外部大規模モデルによる判定も検討したが，実行コストや統合の複雑さが増す一方で，提出系としての再現性を損なう懸念があった．そのため，本稿では学習で単一推論を底上げし，推論時には多数決と文字列正規化で安定化するという，構成が単純で効果の明確な提出系を採用した.

5 実験と評価

5.1 実験設定

我々のチームで構築した学習・推論パイプラインの有効性を検証するため，コンペティション運営から提供された開発用評価データセット（100 問）と，MATH-500 日本語版データセット [12] の 2 種類で評価を行った．評価指標には単一推論の正解率と，多数決後の正解率 `cons@k` を用いた．一致判定には数式同値判定器による normalized match を用い, 必要に応じて文字列正規化を適用した．また, `cons@40` の推論は ABCI 3.0 の 1 計算ノード上で 100 問あたり約 960 秒で完了し，競技制約内で十分実行可能であった.

5.2 定量評価結果

ベースモデル，長文 SFT モデル，および GRPO モデルの性能比較を行った結果を 表 1 に示す．開発用評価データセットでは，ベースモデルの `cons@1=0.440` に対し, 長文 SFT によって `0.840` まで向上し, GRPO を適用した最終モデルでは `0.870` に達した．また，GRPO モデルに多数決を適用すると `cons@40=0.960` を記録し，テストタイムスケーリングの有効性が確認できた.

改善量の内訳を見ると，ベースモデルから長文 SFT への伸びが最も大きく，主力 CoT データの選別と長文コンテキスト学習が単一推論の基盤性能を決めていることが分かる．一方で，GRPO の追加改善は絶対値としては大きくないものの，一貫して正答率を上積みしており，SFT で獲得した推論骨格を正答方向へ洗練する役割を果たしたと解釈できる.

MATH-500 日本語版でも，ベースモデルの `pass@1=22.2%` に対し，SFT モデルで `38.6%`, GRPO モデルで `71.0%` と大幅な改善が見られた．この結果から，主力 CoT データの選別，長文 SFT，GRPO，多数決という各段階の改善が一貫して性能向上に寄与したことが分かる.

特に，多数決の効果は単一推論の改善と競合するのではなく，それを前提としてさらに安定化を図る形で働いた．単一推論の質が低い段階では投票対象自体が不安定になるが，SFT と GRPO により候補解答の質が底上げされることで，多数決は有効な後段のスケーリング手段として機能した.

指標	Base	SFT	GRPO
開発 `cons@1`	0.440	0.840	0.870
開発 `cons@40`	0.680	0.930	0.960
外部 `pass@1`	22.2%	38.6%	71.0%

表 1: 開発用評価データセットにおける主要な比較結果

多数決の集計では，数式表記の揺れを吸収する文字列正規化が重要であった．単純な文字列一致では集計が割れるケースがあるため，正規化を併用することで多数決の効果をより安定して引き出せた.

6 おわりに

6.1 試行錯誤

提出系を構築する過程では，いくつかの有望な案も検証した．ここではその試行を簡潔に共有するが，これらの手法が本質的に無効であることを意味するものではなく，本プロジェクトの制約下では最終提出系に採用しなかった，という位置づけである．

6.1.1 Tool-Integrated Reasoning（TIR）

TIR は，LLM が生成した Python コードを実行し，その結果を推論へ戻すことで計算ミスを補正する方向性として検証した．実際，GPT-OSS-20B を用いた TIR 評価系は開発用評価データセットで `pass@1 = 0.86` を示し，一定の有効性を確認した．一方で，無限ループ，実行環境の統合コスト，出力の不安定性が残り，大規模な提出推論へ安定に組み込むには課題があった．そのため，本稿では TIR を補助的な分析系として位置づけ，最終提出システムには含めなかった．

6.1.2 `s1` / `wait` 系の推論延長

推論時に `Wait` を挿入して思考を延長する設定や，`s1`（Simple test-time scaling）[13] 型の単純な推論延長も比較した．これらは追加の思考を促す手法として興味深かったが，本タスクでは出力長の増大と設定依存のばらつきが大きく，提出系の主軸には据えなかった．最終提出では `wait=0` を採用し，多数決と文字列正規化を中心とする構成を優先した．

6.1.3 RAG

retrieval-augmented generation（RAG）[14] は，過去の数学問題と解答をベクトル検索し，類似例を現在の問題へ付加する方式として検証した．しかし，本タスクでは参照例がそのまま有効に働く場面が限定的であり，最終提出システムには採用しなかった．

6.1.4 Recursive Self Aggregation（RSA）

Recursive Self Aggregation（RSA）[15] は，多数の候補解答を生成し，相互参照しながら反復的に洗練する方式として検証した．候補の改善を促す可能性はあったが，十分な評価と安定化まで至らず，提出系には含めなかった．

6.2 まとめ

本稿では，主力 CoT データの選別，12B 級モデルへの長文 SFT，GRPO による正答報酬ベースの最適化，多数決による推論スケーリングを組み合わせた数学タスク向けパイプラインを報告した．開発用評価データセットでは，ベースモデルから SFT，GRPO へと段階的な性能向上が確認され，多数決が追加の改善をもたらした．

比較的小規模なモデルであっても，学習データの品質管理と長文推論に合わせた学習・推論設計を一貫して行うことで，高度な数理推論能力を引き出せることが示された．特に，データ選別と長文 SFT が単一推論の基盤性能を担い，GRPO と多数決がその上で正答方向への洗練と安定化を担うという役割分担が有効であった．

この結果は，数学のように中間推論の品質が最終正答率へ直結するタスクでは，個々の手法を単独で最適化するよりも，データ，学習，推論の各段階を接続した設計が重要であることを示している．今後も同様の方針は，日本語数理推論モデルの改良における基本的な設計原理になると考えられる．

謝辞

本プロジェクトでは, 産業技術総合研究所の AI 橋渡しクラウド（ABCI 3.0）の計算資源を利用した．また，コンペティションを主催された国立情報学研究所（NII）に感謝する．

参考文献

[1] J. Wei ほか，「Chain-of-Thought Prompting Elicits Reasoning in Large Language Models」, Advances in Neural Information Processing Systems, 2022, pp. 24824–24837.

[2] S. Toshniwal と others,「OpenMathInstruct-1: A 1.8 Million Math Instruction Tuning Dataset」, arXiv preprint arXiv:2402.10176, 2024.

[3] Z. Shao ほか,「DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models」, arXiv preprint arXiv:2402.03300, 2024.

[4] D. Guo ほか,「DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning」, arXiv preprint arXiv:2501.12948, 2025.

[5] X. Wang ほか,「Self-Consistency Improves Chain of Thought Reasoning in Language Models」, The Eleventh International Conference on Learning Representations, 2023.

[6] OpenAI,「gpt-oss-120b & gpt-oss-20b Model Card」. 2025 年 8 月.

[7] M. Cao, X. Wang, と N. Aletras,「Progressive Depth Up-scaling via Optimal Transport」. [Online]. 入手先: https://arxiv.org/abs/2508.08011

[8] N. Lambert と others,「Tülu 3: Pushing Frontiers in Open Language Model Post-Training」, arXiv preprint arXiv:2411.15124, 2024, [Online]. 入手先: https://arxiv.org/abs/2411.15124

[9] Allen Institute for AI,「Open Instruct」.

[10] H. Kydl\iček,「Math-Verify: Math Verification Library」. [Online]. 入手先: https://github.com/huggingface/Math-Verify

[11] C. An ほか,「Polaris: A post-training recipe for scaling reinforcement learning on advanced reasoning models」. [Online]. 入手先: https://hkunlp.github.io/blog/2025/Polaris/

[12] J. Morrison,「MATH-500-japanese」. 2025 年.

[13] N. Muennighoff と others,「s1: Simple Test-Time Scaling」, arXiv preprint arXiv:2501.19393, 2025.

[14] P. Lewis ほか,「Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks」, Advances in Neural Information Processing Systems, 2020. [Online]. 入手先: https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html

[15] S. Venkatraman ほか,「Recursive Self-Aggregation Unlocks Deep Thinking in Large Language Models」. [Online]. 入手先: https://arxiv.org/abs/2509.26626

付録

学習設定の要約

本文では省略した学習設定と推論条件のうち，主要なものを表 2 から表 5 にまとめる． 長文 SFT では，48 層へ拡張した 12B 級モデルに対して最大系列長 16384 のフルパラメータ学習を行い， 合成データセットの出力形式に沿うよう，SFT 側で用いた chat template / system prompt に GRPO 側の設定を合わせた．

項目	値
SFT 最大系列長	16384
prompt 長上限	1024
response 長	7168
device ごとの batch	1
unique prompts	8
samples / prompt	32
rollout batch	256

表 2: 長さとバッチサイズの設定

項目	設定
verifier	qa_10k=math-verify
ground truths	ground_truth
template	math_problem_with_boxed
DeepSpeed stage	3
epochs	1
learners / node	3
vLLM engines	5
tensor parallel	1
GPU mem util	0.65
beta	0.00
ref policy	false
sync backend	nccl
prefix caching	on
save traces	on
eager mode	on
grad checkpoint	on
active sampling	on
zero-std filter	on
async steps	4
inflight updates	on
mask truncation	on
GRPO steps	1725

表 3: GRPO 実行時の主要設定

安定化設定

非同期 RL の下では clip higher は発動しない挙動のため設定から外し， 数値安定化には truncated_importance_sampling_ratio_cap = 2.0 を有効にした． また，advantage_normalization_type = centered により平均中心化のみを適用した． また，指定していないがデフォルトで loss_fn = dapo, loss_denominator = token が使われ， token-level loss 集約を構成している．

項目	値
sampling temperature	0.7
max generation length	16384
samples	40

表 4: 代表的な推論条件

ステップ範囲	温度
0–999	1.000
1000–1300	1.010
1300–1400	1.020
1400–1500	1.015
1500–1600	1.020
1600–1700	1.015
1700–1725	1.018

表 5: GRPO 学習後半の温度スケジュール

温度を小数点第 1 位のオーダーまで上げると，truncation mask の急増によってバッチサイズ維持が難しくなり， 生成長が両極端に割れて学習が不安定化した．そのため，最終的には 1.018 を採用した．