数学タスクでの取組

-GRPOへの挑戦と推論プロンプト -

`Team JINIAC`

目次

チーム紹介

- 経済産業省（ＧＥＮＩＡＣ）プロジェクトでチームを組んだメンバーが再集結
- 世代も地域も多様な産学官９名の混成チーム
- 学術的なスタンスにこだわらず、「分からないことは試してみる」

メンバー

岩永昇二		佐野敏幸		白石尽誠	
辻大地		中島壽希		knishimae	
堀江吏将		元谷崇		森永雄一朗	

レシピ

実験環境

mdx環境に加え、Google colabでも実験

GPU環境を提供いただいた、オーガナイザーの皆様に感謝申し上げます

レシピ（続き）

- ベースモデル : **llm-jp/llm-jp-3-13b-instruct2**
- 追加チューニング : **ＧＲＰＯ**(Group Relative Policy Optimization)
- 使用データセット : DigitalLearningGmbH/MATH-lightevalの一部（２０００データ）
- 推論 : 様々なプロンプトを組み合わせて比較

ＧＲＰＯへの挑戦

DeepSeek の Readme を参考に実施した変更

- **データセットの前処理**
  - "\boxed{...}" 内の内容（ネストにも対応）を回答として抽出
- **報酬関数**
  - 正解報酬と部分的フォーマット評価報酬による評価

Temperature の調整

- 学習時の **temperature を 0.6 に固定** し、出力の一貫性と安定性を向上。

実装で苦戦したポイント

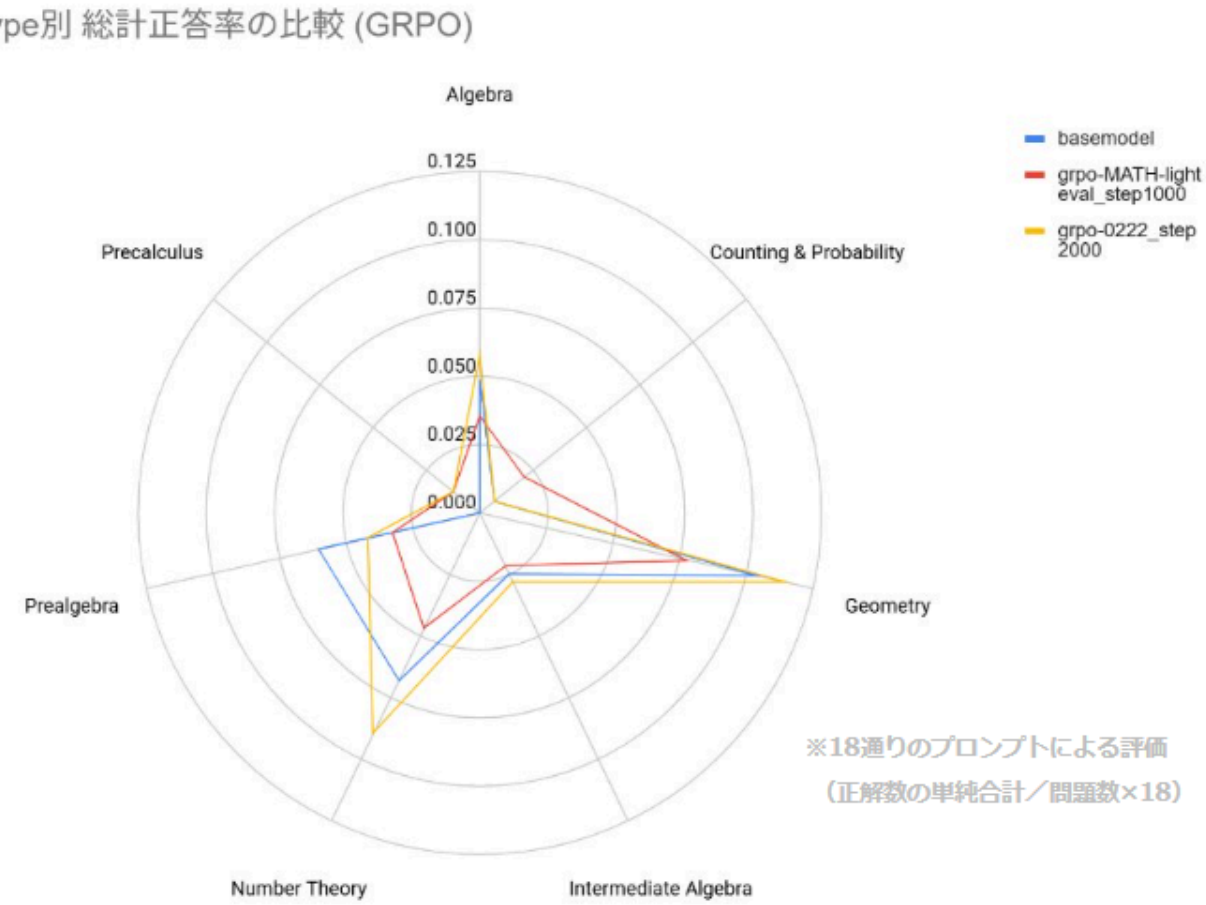
プロンプトの調整

- 当初は学習時に**システムプロンプトを使用**して
  推論過程 (`<reasoning>...</reasoning>`) の出力を指示していたが、
  推論過程が適切に出力されない問題が発生していた。
- そこで、**システムプロンプトではなくユーザープロンプトで**
  推論過程の出力を指示するよう変更したところ、
  推論過程が適切に出力されるようになった。

実験結果

GRPO1000件 : ベースモデルより
全体的に性能低下

GRPO2000件 : 全体的に若干性能向上



推論プロンプト

効果のあった４つの論文の手法を実装（日本語バージョン、英語バージョン、日英両方バージョンを比較）

- Zero-shot Chain of Thoughts (step by step)
  Large Language Models are Zero-Shot Reasoners
- Re-reading（問題を２度読ます）
  Re-Reading Improves Reasoning in Language Models
- EchoPrompt（問題を言い換えさせる）
  EchoPrompt: Instructing the Model to Rephrase Queries for Improved In-context Learning
- Self-Consistency（多数決）
  Self-Consistency Improves Chain of Thought Reasoning in Language Models

ご清聴ありがとうございました

APPENDIX

気づき、感想

- 計算環境
  - 高価な計算環境を生かしきれず，そこは悔いが残った．次はもっと環境を使い切りたい
- モデルやチューニング手法
  - GRPOは、人間のお勉強でいうなら「問題演習」
  - GRPOの活用方法とDeepSeek-R1の精度の高さを学ぶことが出来た
  - GRPOにおいて、モデルが報酬を得るために計算過程を省略し、数字のみを出力することがあった。意図しない挙動となり、強化学習の難しさを感じつつも、報酬を得ようと頑張っている所がとても興味深かった
- トラブルシュートについて
  - 先人(DeepSeek)の資料にヒントが隠れていた

気づき、感想（続き）

- チーム開発について
  - チームで取り組むと、一人では解消できない問題を誰かが解いてどんどん前に進むことができる
  - 個々人が集まった時の相乗効果がとてもみててわくわくした
  - ソースの中に 知恵知識を積み込んで、チームで共有、再現性を行うことでモデルをブラッシュアップして開発を進めることが出来ました
- 最新技術への挑戦
  - Nvidia株下落70兆円というインパクトを与えたオープンソースAIの気配（GRPO）を実際に稼働させて知ることが出来ました
  - 最新の技術のトレースがここまで可能になるのは今後が楽しみ

挑戦したが、提出モデルに搭載できなかったアイディア

- Tanuki 8×8B を用いた合成データでSFT
- SFTとGRPOの組み合わせ
- GRPOにおけるメモリ削減
- GRPOの報酬に「段階的推論」「出力長」「重複ペナルティ」を追加
- PAL(Program-aided Language Models)という数学のプログラム化支援手法の踏襲
- tokenizerの空き部分を、チューニング用特殊トークンに書き換え
- ハイパラ探索

他多数

報酬関数の変更点とその影響

変更前

関数名	評価基準	スコア付け
`correctness_reward_func`	回答が正しいか	2.0 (正解) 0.0 (不正解)
`int_reward_func`	出力が整数か	0.5 (整数) 0.0 (その他)
`strict_format_reward_func`	厳密なフォーマットか	0.5 (完全一致) 0.0 (違反)
`soft_format_reward_func`	フォーマットが 概ね合っているか	0.5 (部分一致) 0.0 (違反)

問題点

- `int_reward_func` により、推論過程が出力されず、数値のみを出力する方向に学習が進んだ。
- フォーマットの評価 (`strict_format_reward_func`) が厳密すぎて、スコアが 0 になることが多かった。

変更後

関数名	評価基準	スコア付け
`correctness_reward_func`	`\boxed{}` の 中身が正解かどうか	2.0 (正解) 0.0 (不正解)
`partial_format_reward_func`	`<reasoning>... </reasoning>` や `\boxed{}` の存在を評価	**最大 2.0** (フォーマット 完全一致) **部分評価あり**

改善点

- 厳密すぎるフォーマットの評価を緩和した。タグ単位での評価を導入することで、柔軟なフォーマットを許容しつつ、推論を促進。
- 併せてフォーマットの乱用 (`\boxed{}` の連発など) を防ぐペナルティを追加し、不適切な出力を抑制。

実験したプロンプト一覧

- Zero-shot prompting

```
"Q: d["text"]"
"A:"
```

- 出力形式の指示

```
"回答は必ず \"<reasoning>\n\" で始まっていることを確認してください。"
"理由を述べ、最終的な回答を \\boxed{} 内に記入してください。"
"Q: d["text"]"
"A:"
```

- Zero-shot Chain of Thoughts <step by step>

```
"回答は必ず \"<reasoning>\n\" で始まっていることを確認してください。"
"理由を述べ、最終的な回答を \\boxed{} 内に記入してください。"
"Q: d["text"]"
"A:ステップバイステップで考えてみましょう。"
```

実験したプロンプト一覧

- Role-Play Prompting + step by step

```
user:"これからあなたは優秀な数学教師となり、生徒に数学の問題を常に正しく教えます。
そして私はあなたの生徒の一人です。"
assistant:"それは素晴らしいことです!
あなたの数学教師として、私はあなたが簡単に理解できるように数学の概念を正しく説明するために最善を尽くします。"
"数学の問題や質問があれば遠慮なく尋ねてください。
喜んでお手伝いします。
一緒に数学の世界に飛び込んで、その素晴らしさを探求しましょう!"
user:"回答は必ず \"<reasoning>\n\" で始まっていることを確認してください。"
"理由を述べ、最終的な回答を \\boxed{} 内に記入してください。"
"Q: d["text"]"
"A:ステップバイステップで考えてみましょう。"
```

- Take a Deep Breath + step by step

```
"""回答は必ず \""<reasoning>\n\"" で始まっていることを確認してください。""
""理由を述べ、最終的な回答を \\boxed{} 内に記入してください。""
""Q: d[""text""]""
""A:まず、深呼吸をして、それからステップバイステップで考えてみましょう。"""
```

実験したプロンプト一覧

- Re-Reading + step by step

```
"回答は必ず \"<reasoning>\n\" で始まっていることを確認してください。"
"理由を述べ、最終的な回答を \\boxed{} 内に記入してください。"
"Q: d["text"]"
"もう一度問題を読んでみましょう: d["text"]"
"A:ステップバイステップで考えてみましょう。"
```

- EchoPrompt + step by step

```
"回答は必ず \"<reasoning>\n\" で始まっていることを確認してください。"
"理由を述べ、最終的な回答を \\boxed{} 内に記入してください。"
"Q: d["text"]"
"A:問題を繰り返した後、ステップバイステップで考えてみましょう。"
```

実験したプロンプト一覧

- Take a Step Back + step by step

  <ターン１>

```
"この質問の背後にある数学の公式は何ですか？"
"Q: d["text"]"
```

  <ターン２>

```
"回答は必ず \"<reasoning>\n\" で始まっていることを確認してください。"
"理由を述べ、最終的な回答を \\boxed{} 内に記入してください。"
"Q: d["text"]"
"公式: <<ターン１の答え>>"
"A:ステップバイステップで考えてみましょう。"
```

参考文献

- @fuyu_quant. (2023, October 25). LLMのプロンプト技術まとめ. Qiita. https://qiita.com/fuyu_quant/items/157086987bd1b4e52e80
- Arora, D., & Zanette, A. (2025). Training Language Models to Reason Efficiently. arXiv preprint arXiv:2502.04463.
- Han, D., & Han, M. (2025, February 6). Train Your Own R1 Reasoning Model with Unsloth (GRPO). Unsloth. https://unsloth.ai/blog/r1-reasoning
- Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., & Iwasawa, Y. (2022). Large language models are zero-shot reasoners. Advances in neural information processing systems, 35, 22199-22213. arXiv preprint arXiv:2205.11916.

- Mekala, R. R., Razeghi, Y., & Singh, S. (2023). EchoPrompt: instructing the model to rephrase queries for improved in-context learning. arXiv preprint arXiv:2309.10687.
- Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., ... & Guo, D. (2024). Deepseekmath: Pushing the limits of mathematical reasoning in open language models. arXiv preprint arXiv:2402.03300.
- Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., ... & Zhou, D. (2022). Self-consistency improves chain of thought reasoning in language models. arXiv preprint arXiv:2203.11171.
- Xu, X., Tao, C., Shen, T., Xu, C., Xu, H., Long, G., & Lou, J. G. (2023). Re-reading improves reasoning in language models. arXiv preprint arXiv:2309.06275.