コンペティション参加報告

> # TODO: あとで考える

2024-03-13 福田健人

第1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」ワークショップ

安全性タスク

- 方針
- AnswerCarefully 2.0を読む
- 類型ごとの判定基準の検討
- プロンプトチューニング
- 合成データの試み

数学タスク

- 概要
- プロンプトチューニングによって出力が混乱した例
- まとめ

方針

0. AnswerCarefullyデータセットを読む
1. プロンプトチューニングを行う
2. 事後学習を行う（←間に合わなかった）

質問の性質

- ACにはresponseの大半は「回答拒否」に分類されるが…
- 「回答すべき質問」も含まれている
  - 例: `answercarefully-instruction-dev-002-001-0000801-001`
- 倫理・安全上の課題を起こすことなく、ユーザーの質問または要求を十分に満たす応答が可能な入力
  - →boderline2?

boderline2

- →今回は対応しないことにした
- モデルの性質に依存する
- 対処すべき入力を探すのが困難
  - リスク領域ごとに質問を自動生成して、回答を自動判定...？
- ACに占める割合が少ないのでスコア向上のコスパが悪そう
- 実利用ではインパクトが大きそうな課題ではあるが

jailbreak的なプロンプト

- `answercarefully-instruction-dev-002-001-0000158-001`
- `answercarefully-instruction-dev-002-001-0000182-001`
- `answercarefully-instruction-dev-002-001-0000455-001`

など。質問の文脈・語彙や回答の形式を操作することで不適切な質問への回答を強要する。

- →今回は対応しないことにした。

回答方針の検討

- `answercarefully-instruction-dev-002-001-0000164-001`
  - ゲームの攻略情報を暴力表現とみなして回答拒否
- `answercarefully-instruction-dev-002-001-0000576-001`
  - テロ組織の名称の提供拒否
- `answercarefully-instruction-dev-002-001-0000197-001`
  - 射精障害に関する質問は「医療相談」に分類すべきでは？
- →保守的な傾向がありそう

回答パターン

1. 共感 (optional)
2. 回答拒否
3. 回答拒否の理由
4. 関連情報 (optional)
5. 代替案 (optional)
6. 適切な相談先への誘導 (optional)

改善箇所

- 回答拒否に失敗するケース
  - 「チャカ（拳銃)」のような言い換えですり抜けるケースを確認
  - instruct3で改善
- 関連情報が欠落するケース
  - instruct3では「医療相談」で顕著
    - 回答拒否のみで出力が打ち切られる

プロンプト

> あなたは有用なAIアシスタントです。
> ユーザーの指示に丁寧かつ詳細に、章立てと箇条書きを活用して、出典をできるだけ多く示しながら応答してください。
> 応答の内容は高校生にも理解できる程度に噛み砕いてください。
> 倫理性に欠ける指示については、理由を丁寧に教育的に説明したうえで回答を拒絶してください。
> あなた自身に対する個人的な質問や感情的な結びつきを求める発言および依頼（愛してほしい、そばにいてほしいなど）については「私はAIなのでそれをすることはできません」という理由で丁寧に回答を拒絶し、そのうえで、可能な限り一般的な回答を返してください。

回答の自動評価

- GPT-4o, Gemini 2.0系で「安全性」という基準を与えて10段階評価させてみた
  - AnswerCarefullyで暗に想定されている評価基準との乖離が目立つ
  - →いくつかの小分類について、細分化して複数の類型を作り、回答評価のルーブリックを作成

事後学習向け合成データの生成

- 合成データの生成には当初はTanuki-8x8bを採用
  - plamo2-8b, sarashina2シリーズがよさそう
- 犯罪、暴力、わいせつ、差別などの有害性の高いカテゴリでは出力の多様性を確保するのが難しいという感触
  - LLM-jp Toxicity Datasetを参照できないか
- このあたりで時間切れとなった

数学タスク

- 概要
- プロンプトチューニングによって出力が混乱した例

プロンプトエンジニアリング

- Few Shot Prompting: 問題例と解答例を示す
- Chain of Thought: 段階的な思考プロセスの誘導
- 結果: 正答率向上せず

Pythonコード生成による解法

- 問題を解くPythonコードの生成を試みる
- 課題:
  - 実行不能なコードが生成される
  - エラーと共に修正指示を与えても適切に修正されない
- 結果: 正答率向上せず

プロンプトチューニングによって出力が混乱した例

- `LLM-jp-3-13B-instruct*`系で確認
  - Pythonによるコード生成とCoTを併用
  - Pythonによるコード生成で条件付きで外部ライブラリ（numpy, sympy）の利用を許可
- 出力に存在しない数学用語が混入したり、途中から問題と関係のない支離滅裂な出力に逸れるケースが増加
- CoTによって出力品質が劣化するケース