

CPPO: グループ関係者のトレーニングの加速 ポリシー最適化に基づく推論モデル

Zhihang Lin^{1,2}, Mingbao Lin³ Yuan Xie^{2,4}, Rongrong Ji¹ † ¹中国教育
部マルチメディア信頼知覚と高効率コンピューティング重点研究室、厦門大学、361005、中国²上海
イノベーション研究所、中国³Skywork AI、シンガポール⁴華東師範大学、上海、中国
lzhedu@foxmail.com, linmb001@outlook.com、
yxie@cs.ecnu.edu.cn、
rrji@xmu.edu.cn

抽象的な

この論文では、グループ相対ポリシー最適化 (GRPO) に基づく推論モデルのトレーニングを加速するための補完刈り込みポリシー最適化 (CPPO) を紹介します。GRPO は効果的ですが、質問ごとに複数の補完をサンプリングする必要があるため、トレーニング コストが高くなります。実験と理論分析により、補完の数はモデルの精度に影響しますが、トレーニング時間は乗法的に増加し、すべての補完がポリシー トレーニングに等しく貢献するわけではなく、貢献は相対的な利点によって決まることが明らかになりました。これらの問題に対処するために、絶対的な利点が低い補完を刈り込み、勾配の計算と更新に必要な数を大幅に削減するCPPO を提案します。さらに、追加の質問を組み込むことで GPU の使用率を最大化し、トレーニング効率をさらに高める動的補完割り当て戦略を導入します。実験結果では、CPPO は元の GRPO と比較して、精度を維持または向上させながら、GSM8K で最大8.32 倍、Math で最大 3.51 倍の高速化を達成することが実証されています。私たちのコードは <https://github.com/lzhxmu/CPPO> で公開されています。

1 はじめに

最近では、OpenAI-o1 [9]、Deepseek-R1 [6]、Kimi-k1.5 [20]などのモデルを中心に、高度な推論モデルの開発と応用が急増しています。これらのモデルは、段階的な推論と反映を通じて、数学、コーディング、科学的推論などの複雑な推論タスクにおいて優れた能力を発揮します。

強化学習は推論モデルをトレーニングするための効果的な方法であることが証明されています。Deepseek-R1 [6] は、ルールベースの強化学習によって推論パターンを効果的に引き出せることを実証しています。グループ相対ポリシー最適化 (GRPO) [17]を採用しています。これは、グループスコアから直接ベースラインを推定することで近似ポリシー最適化 (PPO) [16]とは異なり、批評家モデルを必要としません。ただし、これには各質問の完了グループのサンプリングが必要となり、トレーニング プロセスの計算コストが高くなります。その後、GRPO はルールベースの報酬関数を使用して各完了の報酬を計算し、各完了の相対的な利点を計算します。トレーニングの安定性を確保するために、GRPO は、ポリシー モデル、参照モデル、および古いポリシー モデルの予測確率の比率をポリシー目的関数の一部として計算し、トレーニングのオーバーヘッドをさらに増加させます。

強化学習。GRPO のトレーニング オーバーヘッドが大きいため、トレーニングの効率とスケーラビリティが制限されます。トレーニングの効率を改善することは、重要かつ実用的な問題です。

GRPO トレーニングの計算コストは、主にそのコア設計に起因しています。つまり、グループ内比較のためにプロンプトごとに大規模な補完グループを生成するため、トレーニング プロセスの計算コストが高くなります。さらに、GRPO の順方向計算は、補完数(3 倍)の係数でスケーリングされます。強化学習プロセスに対する各補完の貢献が等しいかどうか疑問に思うのは当然です。セクション 3.2 では、各補完の貢献はその相対的な利点に関連していることがわかります。言い換えれば、ポリシー モデル トレーニングに対する各補完の貢献は等しくありません。この洞察により、補完を削減することで GRPO を高速化することが思いつきました。

本稿では、グループ相対ポリシー最適化 (GRPO) を加速するための補完ブルーニング ポリシー最適化 (CPPO) を提案します。各補完の強化学習プロセスへの貢献は大きく異なり、相対的な利点と密接に関連していることから、CPPO は利点に基づいて補完をブルーニングし、強化学習プロセスを加速します。具体的には、ポリシー モデルは最初に各質問の補完のグループをサンプリングします。次に、各補完の相対的な利点が報酬関数によって計算されます。次に、CPPO は絶対的な利点の値が低い補完をブルーニングし、損失計算用に絶対的な利点が高い補完のみを保持します。このプロセスにより、トレーニングに必要な補完の数が大幅に削減されるため、トレーニング プロセスが高速化されます。さらに、補完のブルーニングにより GPU リソースが十分に活用されず、リソースの無駄が生じることがわかります。これに対処するために、各デバイスで新しい質問からの補完で満たし、GPUリソースを最大限に活用してトレーニング効率をさらに高める動的補完割り当て戦略を導入します。

CPPOの有効性を評価するために、複数の困難なベンチマークと異なる規模のモデルで実験を行いました。具体的には、Qwen-2.5-1.5B-InstructやQwen-2.5-7B-InstructなどのQwen-2.5シリーズモデル[24]を、Math [7]やGSM8K [3]を含む数学データセットでトレーニングしました。結果は、CPPOが元のGRPOと比較して精度を維持または向上させながら、GSM8Kで最大8.32倍、Mathで3.51倍の高速化を達成することを実証しました。

2 関連研究

大規模推論モデル。大規模言語モデル (LLM) [1; 21; 19; 2]は、さまざまな自然言語処理タスクで目覚ましい進歩を遂げてきました。最近、研究者は、数学[3; 7]、コーディング[10]、科学的推論[15]などの推論タスクにおける大規模言語モデルのパフォーマンスを向上させ続けています。Snellら[18]は、密なプロセススペースの検証者報酬モデルを使用し、テスト時のプロンプトに基づいてモデルの応答分布を適応的に更新して推論能力を高めています。rStar-Math [5]は、小規模LLMの数学推論機能を大幅に向上させる自己進化型の深い思考アプローチを提案しています。OpenAI-o1 [9]は、大規模な強化学習を使用して、複雑な推論タスクを解決できる推論モデルをトレーニングし、複数のベンチマークで最先端のパフォーマンスを達成しています。ただし、OpenAI-o1のトレーニングの詳細は公開されていないため、推論モデルの複製と拡張は困難です。

強化学習。最近、DeepSeek-R1 [6] は、グループ相対ポリシー最適化を通じて大規模言語モデルの推論能力を奨励しました。DeepSeek-R1 の成功に触発されて、Logic-RL [23] は、ルールベースの強化学習のトレーニング効率と安定性を高めるために REINFORCE++ アルゴリズムを採用しています。Hu ら[8] は、KL ダイバージェンス制約のないバニラ PPO アルゴリズムが、応答の長さや推論タスクのベンチマーク パフォーマンスの両方をスケールアップするのに十分であることを実証しています。ただし、これらの強化学習アルゴリズムは、各質問に対して普遍的に複数の完了を必要とするため、かなりの計算コストがかかります。強化学習アルゴリズムのトレーニングを加速することが緊急に必要です。

推論モデルの推論加速。モデル推論機能の強化は、多くの場合、計算オーバーヘッドの増加と応答時間の延長を伴います。最近の研究では、効率的な思考連鎖 (CoT)法を通じて推論モデルの推論プロセスを加速しようとしています。TokenSkip [22]は、重要度の低いトークンを選択的にスキップしながら重要なトークンを保持することで推論効率を向上させ、効率と精度のバランスを実現する制御可能なCoT圧縮法を提案しています。Kangら[11]は、コンプレッサーを使用して、元の長いCoTを短いCoTに圧縮しながら、重要な情報と解釈可能性を維持しています。推論加速に焦点を当てた研究は数多くありますが、推論モデルのトレーニングの加速はまだ十分に研究されていない分野です。

3 方法

3.1 予備

グループ相対ポリシー最適化。GRPO [6]は、通常ポリシーモデルと同じサイズの批評家モデルを使わず、代わりにグループスコアからベースラインを推定します。具体的には、データセット分布 $P(Q)$ からサンプリングされた各質問 q に対して、GRPOは古いポリシーモデル $\pi_{\theta old}$ を使用して G 個の補完 $\{o_1, o_2, \dots, o_G\}$ を生成します。そして、GRPO は次の目的を最大化することでポリシー モデル π_{θ} を最適化します。

$$J_{GRPO}(\theta) = E_{q \sim P(Q), \{o_i\}_{i=1}^G \sim \pi_{\theta old}(o|q)} \frac{1}{G} \sum_{i=1}^G \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \frac{\pi_{\theta}(o_i, t|q, o_i, <t)}{\pi_{\theta old}(o_i, t|q, o_i, <t)} A_i \cdot \text{アップ} \frac{\pi_{\theta}(o_i, t|q, o_i, <t)}{(\pi_{\theta old}(o_i, t|q, o_i, <t))^{1-\beta}} - \beta D_{KL}[\pi_{\theta} || \pi_{ref}] \quad (1)$$

どこ

$$D_{KL}[\pi_{\theta} || \pi_{ref}] = - \log - 1. \frac{\pi_{ref}(o_i, t|q, o_i, <t)}{\pi_{\theta}(o_i, t|q, o_i, <t)} \frac{\pi_{ref}(o_i, t|q, o_i, <t)}{\pi_{\theta}(o_i, t|q, o_i, <t)} \quad (2)$$

ここで、 β はハイパーパラメータです。 $\pi_{\theta ref}$ は参照モデルであり、通常は強化学習前の初期モデルです。 A_i は、各グループ内の完了に対応する報酬のグループ $\{r_1, r_2, \dots, r_G\}$ を使用して計算される利点です。

$$A_i = \frac{r_i - \text{平均}(\{r_1, r_2, \dots, r_G\})}{\text{std}(\{r_1, r_2, \dots, r_G\})} \quad (3)$$

ルールベースの報酬関数。報酬計算のために追加の報酬モデルをトレーニングする代わりに、GRPO は次の 2 つのコンポーネントで構成されるルールベースの報酬システムを採用しています。

$$r_i = R_{format}(o_i) + R_{accuracy}(o_i) \quad (4)$$

ここで、フォーマット報酬 $R_{format}(o_i)$ は出力が予想される構造に準拠していることを保証し、精度報酬 $R_{accuracy}(o_i)$ は正確な応答に対してより高い報酬を与えて正確さを優先します。
ポリシートレーニングへの完了の影響の分析。式(1)から、GRPOのトレーニングオーバーヘッドは、質問ごとにサンプリングされた完了の数に比例して増加します。これは、すべての完了に対してポリシー、参照、および古いポリシーモデルの予測確率を計算する必要があることから生じます。
例えば、DeepSeek-Math [17]では、64個の補完を使用すると、質問ごとに192回のフォワードパス（64×3）が必要になり、かなりの計算コストがかかります。これにより、2つの重要な疑問が生じます。(1)補完の数はポリシーモデルの精度にどのように影響しますか？補完を増やすと常にパフォーマンスが向上しますか？(2)グループ内のすべての補完は、トレーニングに等しく貢献しますか？

最初の疑問を解決するために、Qwen2.5-1.5B-Instruct [24]を用いてGSM8K [3]のアブレーション研究を実施しました。図1の結果は、補完数が増えるとモデルの精度は向上しますが、トレーニング時間は乗法的に増加することを示しています。これは、トレーニングコストが増加すると、パフォーマンス向上の収穫逨減を示しています。重要なのは、コストを削減するために補完数を減らすと、推論能力が低下するリスクがあり、実用的でなくなることです。

2 番目の質問については、補完がトレーニングの有効性に均一に貢献するかどうかを調査します。セクション 3.2 の包括的な分析により、補完の貢献は非常に変動しやすく、一部のサンプルは他のサンプルよりも大幅に多くのトレーニング シグナルを提供していることが明らかになりました。これらの調査結果は、価値の高い補完を識別して優先順位を付ける戦略の開発を促進し、モデルのパフォーマンスを損なうことなくトレーニングの効率を向上させる可能性があります。

3具体的な報酬関数については付録Aに記載されています。

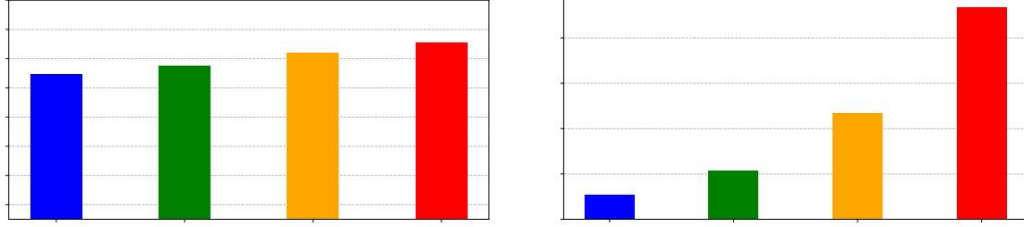


図1: 完了数と (左)精度、(右)トレーニング時間の関係。実験はGSM8K [3]でQwen-1.5B-Instruct [24]を使用し
て実施しました。

3.2 完了貢献分析

各補完がポリシーモデルのトレーニングにどの程度貢献しているかを測定するために、まず、式(1)のポリシー目的関数のモデル
パラメータ θ に対する導関数を次のように計算します。

$$\begin{aligned}
 \nabla_{\theta} J_{\text{GRP}}(\theta) &= E[q \quad P(Q, \{oi\}_{i=1}^G) \pi_{\theta}^{\text{old}}(O|q)] \quad \frac{1}{G} \sum_{i=1}^G \frac{1}{|oi|} \sum_{t=1}^{|oi|} \nabla_{\theta} \frac{\pi_{\theta}(oi, t|q, oi, <t)}{\pi_{\theta}^{\text{old}}(oi, t|q, oi, <t)} A_i \\
 &\quad \frac{\pi_{\text{ref}}(oi, t|q, oi, <t) \pi_{\text{ref}}(oi, t|q, oi, <t) - \beta \nabla_{\theta} - \nabla_{\theta} \log \pi_{\theta}(oi, t|q, oi, <t) \pi_{\theta}(oi, t|q, oi, <t)}{\pi_{\theta}^{\text{old}}(oi, t|q, oi, <t)} \\
 &= E[q \quad P(Q, \{oi\}_{i=1}^G) \pi_{\theta}^{\text{old}}(O|q)] \quad \frac{1}{G} \sum_{i=1}^G \frac{1}{|oi|} \sum_{t=1}^{|oi|} \frac{\nabla_{\theta} \pi_{\theta}(oi, t|q, oi, <t)}{\pi_{\theta}^{\text{old}}(oi, t|q, oi, <t)} A_i \\
 &\quad + \beta \frac{\pi_{\text{ref}}(oi, t|q, oi, <t) \nabla_{\theta} \pi_{\theta}(oi, t|q, oi, <t) \pi_{\theta}(oi, t|q, oi, <t)}{2\pi_{\theta}} - \frac{\nabla_{\theta} \pi_{\theta}(oi, t|q, oi, <t) \pi_{\theta}(oi, t|q, oi, <t)}{q, oi, <t)} \\
 &= E[q \quad P(Q, \{oi\}_{i=1}^G) \pi_{\theta}^{\text{old}}(O|q)] \quad \frac{1}{G} \sum_{i=1}^G \frac{1}{|oi|} \sum_{t=1}^{|oi|} \frac{\pi_{\theta}(oi, t|q, oi, <t)}{\pi_{\theta}^{\text{old}}(oi, t|q, oi, <t)} A_i \quad (5) \\
 &\quad + \beta \frac{\pi_{\text{ref}}(oi, t|q, oi, <t) - 1 \pi_{\theta}(oi, t|q, oi, <t)}{oi, <t)} \frac{\nabla_{\theta} \pi_{\theta}(oi, t|q, oi, <t) \pi_{\theta}(oi, t|q, oi, <t)}{q, oi, <t)} \\
 &= E[q \quad P(Q, \{oi\}_{i=1}^G) \pi_{\theta}^{\text{old}}(O|q)] \quad \frac{1}{G} \sum_{i=1}^G \frac{1}{|oi|} \sum_{t=1}^{|oi|} \frac{\pi_{\theta}(oi, t|q, oi, <t)}{\pi_{\theta}^{\text{old}}(oi, t|q, oi, <t)} A_i \\
 &\quad \text{優位性加重確率比} \\
 &\quad + \beta \frac{\pi_{\text{ref}}(oi, t|q, oi, <t) \pi_{\theta}(oi, t|q, oi, <t) - 1}{q, oi, <t)} \frac{\nabla_{\theta} \log \pi_{\theta}(oi, t|q, oi, <t)}{\text{ポリシーモデルの勾配}} \quad \text{KL発散制約}
 \end{aligned}$$

我々は式(5)で強調されている導関数成分を分析する。(1) アドバンテージ加重確率これは、各完了の貢献をそのアドバンテージに直接結び付けている。

比率項 $\frac{\pi_{\theta}(oi, t|q, oi, <t)}{\pi_{\theta}^{\text{old}}(oi, t|q, oi, <t)} A_i \cdot \text{tag} \circ$

この項は、特定のアクションがベースラインと比較して期待収益をどの程度改善するかをアドバンテージ関数が定量化するため、より高い報酬のあるアクションを優先するようにポリシーにインセンティブを与えます。

この項は、高利点の完了を増幅し、低利点の完了を抑制することにより、報酬に合わせた推論パターンに向けてポリシーの最適化を導きます。(2) KLダイバージェンス制約項 β 参照モデル π_{ref} からの逸脱にペナルティを課することで安定性を強化します。ただし、この制約は本質的にポリシーの推論パターンを形成するよ

うに設計されているのではなく、トレーニング中のスムーズな更新を保証するものです。(3) ポリシーモデル勾配項 $\nabla_{\theta} \log \pi_{\theta}(oi, t|q, oi, <t)$ 。

この項は、モデルパラメータ θ に対するポリシーの予測アクションの対数確率の勾配を表します。

Huらによる最近の研究[8]では、KLダイバージェンス制約を削除しても、ポリシーのコア推論パターンは主に報酬に沿った優位性項によって駆動されるため、トレーニングされたモデルの推論能力が損なわれないことが実証されています。この洞察に基づいて、ポリシー目標の導関数を次のように近似します。

$$\nabla \theta J_{GRPO}(\theta) \approx \mathbb{E}_q \left[P(Q), \{o_i\}_{i=1}^G \pi_{\theta_{old}}(O|q) \right] \frac{1}{G} \sum_{i=1}^G \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \frac{\pi_{\theta}(o_i, t|q, o_i, <t) \pi_{\theta_{old}}(o_i, t|q, o_i, <t)}{\pi_{\theta_{old}}(o_i, t|q, o_i, <t)} \cdot \text{アドバンテージ (前へ進む)} \cdot \nabla \log \pi_{\theta}(o_i, t|q, o_i, <t) \quad (6)$$

確率比 (後送) ポリシーモデルの勾配 (後送)

KL正則化制約を切り離しながら、報酬駆動型学習信号に焦点を当てます。

この定式化をよりよく理解するために、アドバンテージ加重確率比項を確率比項とアドバンテージ項に分解します。ポリシー更新に大きく貢献する完了の場合、式(6)の新しい3つのコンポーネントはすべて無視できないものでなければなりません。これらのコンポーネントのいずれかの値がゼロに近い場合、ゼロの場合、全体的な貢献は最小限になるか、まったくなくなります。

計算タイミングの観点から、これらのコンポーネントは次のように分類できます。(1)確率比とポリシーモデルの勾配は事後フォワード情報であり、ポリシーのフォワード計算後にのみ計算できます。(2)ただし、アドバンテージ項は、ポリシーのフォワード計算の前に計算できる事前フォワード情報を表します。

GRPO トレーニングを加速するという目的から、この事前フォワード情報を活用することに重点を置いています。フォワード計算の前にアドバンテージ項を評価することで、ポリシーモデルを使用して完了を処理するかどうかについて、情報に基づいた決定を下すことができます。具体的には、完了のアドバンテージの絶対値が非常に小さいか重要でないため、結果に大きな違いが生じることなく実質的にゼロとして扱うことができる場合、その完了をバッチから削除します。この選択的処理により、絶対的なアドバンテージが高い完了のみがフォワード計算と勾配更新の段階に進むことが保証されます。

3.3 完了ブルーニングポリシーの最適化

図2に示すように、グループ相対ポリシー最適化のトレーニングプロセスを加速するために、完了ブルーニングポリシー最適化 (CPPO) アルゴリズムを提案します。式(1)のGRPOの最適化目的と比較して、CPPOは十分に高い利点を示す完了のみを含む選択条件を導入します。CPPO目的は次のように定式化されます。

$$J_{CPPO}(\theta) = \mathbb{E}_q \left[P(Q), \{o_i\}_{i=1}^G \pi_{\theta_{old}}(O|q) \right] \frac{1}{G} \sum_{i=1}^G \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \frac{\pi_{\theta}(o_i, t|q, o_i, <t)}{\pi_{\theta_{old}}(o_i, t|q, o_i, <t)} A_i \quad (7)$$

$$\frac{\pi_{\theta}(o_i, t|q, o_i, <t)}{\pi_{\theta_{old}}(o_i, t|q, o_i, <t)} \geq \gamma, \quad 1 - \gamma \leq A_i \leq 1 + \gamma \quad \text{or} \quad A_i \leq -\beta \text{DKL}[\pi_{\theta} || \pi_{ref}]$$

Y,

ここで γ は、 γ 以上の絶対優位性を持つ完了のみが勾配更新で保持されることを保証する事前定義された閾値である。比 $\pi_{\theta}(o_i, t|q, o_i, <t) < 1 - \gamma$ の場合、

$A_i < 0$ 、または $\pi_{\theta_{old}}(o_i, t|q, o_i, <t) \frac{\pi_{\theta}(o_i, t|q, o_i, <t)}{\pi_{\theta_{old}}(o_i, t|q, o_i, <t)} < 1 + \gamma$ かつ $A_i > 0$ の場合、クリップ関数がアクティブになります。このアクションは、モデルの勾配項を事実上無効にし、すべての補完を削除することと同等になります。

単一/複数GPU設定の統合。複数GPUのトレーニングシナリオでは、デバイス間で大きな利点がある完了の数が異なります。このような場合、全体的なトレーニング効率、完了を最も多く処理するデバイスによってボトルネックになります。これはパケツ効果と呼ばれる現象です。これを緩和するために、各GPUでは、各質問に対して絶対的な利点が最も大きい k 個の完了のみを保持します。

$$k = \lceil G \times (1 - P) \rceil \quad (8)$$

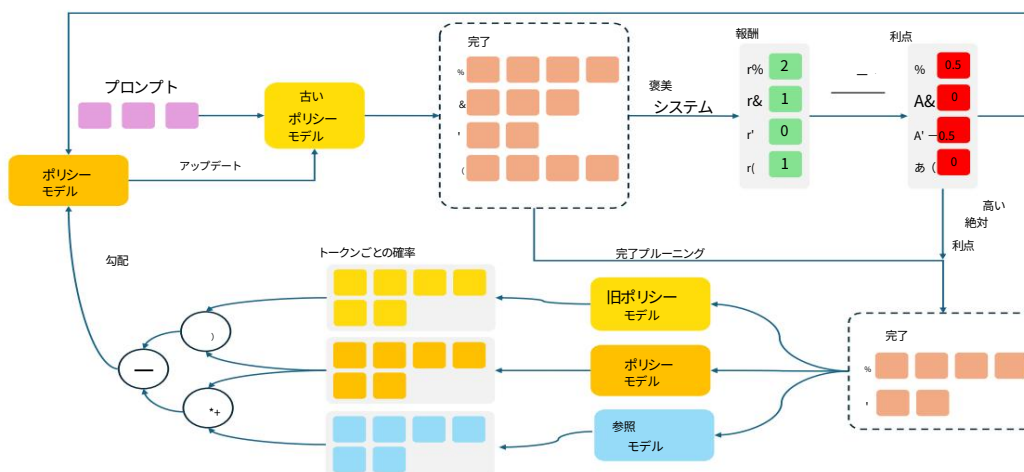


図 2: 完了ブルーニング ポリシー最適化 (CPPO) の概要。利点を取得した後、ポリシー モデル、参照モデル、および古いポリシー モデルのフォワードに対して、絶対的な利点が高い完了のみが保持されます。DRは、ポリシーモデルと古いポリシー モデルの確率比です。

ここで $P \in (0,1]$ は剪定率を表す。この戦略による修正CPPOは次のようになる。

$$\text{JCP O}(\theta) = \text{Eq} \quad \text{P}(\text{Q})_{i \in \{0\}} \text{G} \text{ i=1} \quad \pi_{\theta \text{old}}(\text{o}|q) \quad \frac{1}{k} \quad \frac{1}{i \in \{0\}} \quad \frac{1}{t=1} \quad \text{分} \quad \frac{\pi_{\theta}(\text{o}_i, \text{t}|q, \text{o}_i, < \text{t})}{\pi_{\theta \text{old}}(\text{o}_i, \text{t}|q, \text{o}_i, < \text{t})} \text{Ai} \quad (9)$$

$$\text{ツブ} \quad \frac{\pi_{\theta}(\text{o}_i, \text{t}|q, \text{o}_i, < \text{t}) \text{クリ}}{(\pi_{\theta \text{old}}(\text{o}_i, \text{t}|q, \text{o}_i, < \text{t}))} \cdot 1 - \quad , 1 + \quad) \text{Ai} - \beta \text{DKL} [\pi_{\theta} || \pi_{\text{ref}}] \quad ,$$

ここで、合計は絶対的な優位性値が最も高い k 個の完了に対応するインデックスセット l に対してのみ行われます。つまり、

$$I = \{i \in \{1, \dots, G\} \mid |A_i| \text{ は上位 } k \text{ 個の値のうちの 1 つです}\}. \quad (10)$$

セクション 4.2.2 では、形式と回答が正しいか、形式と回答が誤っているかのいずれかで絶対的な利点値が高い補完が最も明確なトレーニング シグナルを提供することを分析します。絶対的な利点が小さい部分的な正しい補完は、最小限の貢献しかせず、ポリシー モデルを誤解させる可能性があります。これらの補完をトレーニング プロセスから削除すると、モデルのパフォーマンスを損なうことなくトレーニングの効率を高めることができます。

CPPOアルゴリズムのパイプラインは次のとおりです。(1) 古いポリ

シーモデルは、各質問の補完のグループをサンプリングします。

- (2) 報酬関数は式 (4)に従って各完了に対する報酬を計算する。
- (3) 各完了の相対的な利点は式 (3)に従って計算される。
- (4) CPPOは絶対的優位性が最も高いk個の完了を保持する。
- (5) 選択された補完に基づいて政策モデルが更新される。

CPPO と GRPO の主な違いは、CPPO では、ポリシー モデル、参照モデル、古いポリシー モデルの順方向計算にすべての完了を使用しないことです。代わりに、勾配更新の絶対的な利点が高いものだけを保持することで、CPPO は順方向パス中の計算オーバーヘッドを大幅に削減し、トレーニング プロセスを高速化します。

3.4 動的完了割り当てによる並列処理

このセクションでは、CPPPOのトレーニング効率をさらに最適化するための新しい動的補完割り当て戦略を紹介します。GRPOで採用されているような従来のアプローチは、GPUメモリの制約により固有の制限に直面しています。具体的には、1つのデバイスでバッチあたり最大Bの質問を処理でき、各質問はGの候補補完を生成します。

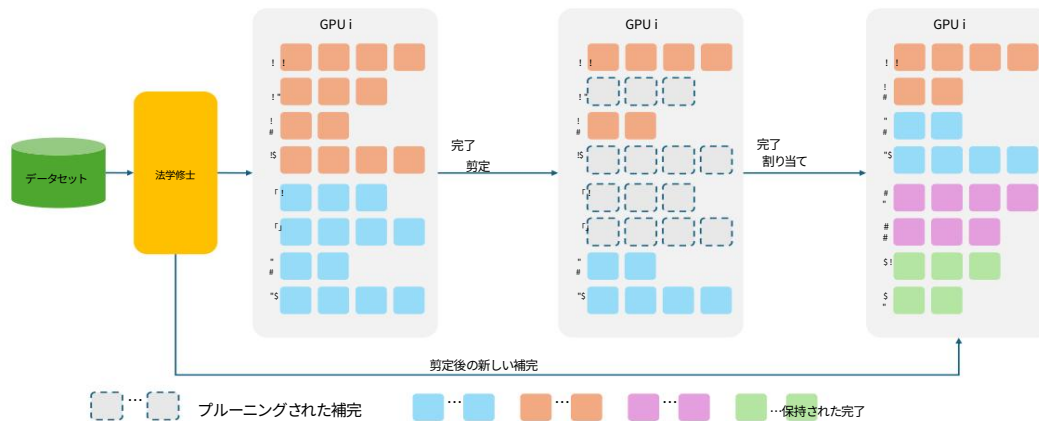


図 3: 並列処理のための動的補完割り当ての図。補完を削除した後、補完割り当てでは新しい質問からの重要な補完を組み込みます。 o は、 n 番目の質問の m 番目の補完を表します。

ブルーニングにより、デバイスごとに保持される完了の合計数が $B \times k$ に減少し、GPU の使用率が最適ではなくなり、並列コンピューティング機能が十分に活用されなくなります。

この非効率性に対処するために、図 3 に示すように、追加の質問から抽出された補完をデバイスの処理パイプラインに動的に割り当てます。この戦略により、元の質問と新しく導入された質問の両方から得られた高品質の補完を継続的にメモリに格納することで、各デバイスがフル稼働することが保証されます。重要なのは、新しく組み込まれた補完はすべて、一貫性と関連性を維持するために、同じ厳格なブルーニングプロセスを経るということです。

このアプローチの利点は 2 つあります。まず、デバイスの並列コンピューティング能力を最大限に活用することで、GPU の使用率を最大化します。次に、各デバイスがバッチごとにより多くの質問を処理できるようになるため、収束を達成するために必要なトレーニングステップの総数が減ります。この二重の最適化により、トレーニング品質を維持しながらトレーニングの効率が向上します。

4つの実験

4.1 実験設定

トレーニングの詳細。我々は、補完を生成するためにvLLM推論ライブラリ[12]を使用するOpen R1フレームワーク[4]に基づいてCPPOを実装する。Qwen2.5-1.5B-InstructモデルとQwen2.5-7B-Instructモデルは、それぞれ80GBのメモリを備えた2つと4つのGPUを使用してトレーニングされる。式(7)の α と β は、0.2と0.04に設定されている。バッチサイズは16に設定され、学習率は $1e-6$ に設定されている。

ポリシー モデルの温度は1に設定され、グループサイズは16に設定されています。各完了の最大長は1024に設定されています。

評価の詳細。Math [7]、AIME2024 [14]、AMC2023 [13]、GSM8K [3]など、難易度の異なる複数のベンチマークでパフォーマンスを評価します。評価プロセスを高速化するためにvLLM [12]を使用します。評価バッチサイズは10に設定されています。MathとGSM8Kの補完を生成するために貪欲復号化を使用します。AIME2024とAMC2023の場合、温度を0.6に設定し、各質問に対して4つの補完を使用します。評価指標としてPass@1の精度を使用します。

4.2 主な結果

GSM8K [3]とMATH [7]データセットで異なるスケールのモデルをトレーニングすることによりCPPOを評価します。GSM8K には 8.5K の小学校レベルの数学問題が含まれ、MATH には 7.5K の競技レベルの問題が含まれます。ベースモデルの推論機能を考慮して、比較的シンプルな GSM8K データセットには Qwen2.5-1.5B-Instruct を使用し、より複雑な MATH データセットには Qwen2.5-7B-Instruct を使用します。

評価のために、各モデルは対応するデータセットのテストサブセットでテストされます。さらに、モデルの分布外推論能力を評価するために、テストベンチマークとしてAMC2023 [13]とAIME2024 [14]を導入します。AMC2023とAIME2024はQwen2.5-1.5B-Instructモデルには難しすぎるため、これらのベンチマークではQwen2.5-7B-Instructモデルのみを評価します。

表1: GSM8KテストサブセットにおけるGRPOとCPPOの比較。GSM8KトレーニングサブセットでQwen2.5-1.5B-Instructをトレーニングし、剪定後の保持された補完の数は

$k = \frac{G}{G + (1 - P)}$ と表記される。

方法	グループ剪定サイズ (G)		率 (P)		k 精度	トレーニングの加速	
						時間	比率
Qwen2.5-1.5B-インストラクト	-	-	-	-	55.72%	-	-
GRPO	16	0.00%	16	77.05 %	23393 秒	50.00% 8	1.00倍
CPPO	16	77.67%	12930 秒	75.00% 4	78.81%	7159 秒	1.81×
CPPO	16	87.50%	2 80.41%	4781 秒	93.75%	1 78.20%	3.27倍
CPPO	16	2813 秒					4.89×
CPPO	16						8.32倍

4.2.1 パフォーマンス比較

GSM8Kのトレーニング。表1に示すように、CPPOはGRPOよりも明らかに優れている。精度と加速比の両方を実現しています。特にCPPOは同等かそれ以上の精度を達成しています。さまざまな剪定率でGRPOよりも優れています。剪定率が87.50%の場合、CPPOの精度は80.41%となり、GRPOの77.05%を3.36%上回りました。

効率性に関しては、CPPOはトレーニングを大幅に加速します。剪定率は93.75%で、8.32倍の加速率。この高速化は補完のプルーニングと補完の割り当て。完了プルーニングは、重要度の低いものを破棄することで計算オーバーヘッドを削減します。完了割り当て戦略は解放されたメモリの使用を最大化し、GPUの並列処理機能を活用します。その結果、CPPOは1回あたりより多くの質問を処理します。バッチ処理と必要なトレーニングステップの総数を削減します。これらの結果は、CPPOが精度を維持または向上させるだけでなく、トレーニングの効率を大幅に向上させ、

大規模な推論モデルのトレーニングのための実用的かつ効果的なソリューション。

MATHのトレーニング。表2では、CPPOはより大きなモデルに拡張でき、最大3.51倍の精度を犠牲にすることなくMATHを高速化します。たとえば、剪定率が87.5%の場合、

CPPO は、トレーニング時間を 3.51 倍短縮しながら、GRPO と同じ精度 (75.20%) を維持します。

さらに、AMC2023およびAIME2024ベンチマークの評価では、CPPOは、絶対的な優位性が高い完了のみを訓練することで、モデルの一般化能力を維持するCPPOは、分布外タスクの強化においてGRPOに匹敵するだけでなく、

推論能力が向上するだけでなく、トレーニング時間も短縮されるため、より効率的な代替手段となります。

安定性と収束。図4にGSM8KとGSM8Kの両方の訓練中の報酬曲線をプロットします。

およびMATHデータセット。全体として、報酬曲線はCPPOがGRPOの訓練の安定性を向上させながら収束速度を向上させた。結果は、CPPOはクラッシュしたり、急激な変動を経験したりしないので、安定したトレーニングには不可欠です。結果はCPPOの堅牢で安定した訓練安定性を示唆している。さらに、CPPOの報酬曲線は明確な上昇傾向を示し、GRPOよりも早く高い報酬値に到達します。このより速い

報酬値の増加は、CPPO がより急速に収束することを示します。

表2: MATHテストサブセットと分布外におけるGRPOとCPPOの比較

AMC 2023とAIME 2024のベンチマーク。Qwen2.5-7B-InstructをMATHトレーニングでトレーニングしますデータセットであり、刈り込み後に保持される補完の数は $k = \frac{G}{G + (1 - P)}$ で表されます。

方法	グループ剪定	サイズ	レート	k 精度	時間	AMC AIMEの加速	比率 2023 2024
Qwen2.5-7B-インストラクト	-	-	-	-	55.20%	-	25.62% 5.00%
GRPO	16	0.00%	16	75.20%	33902 秒	50.00% 8	75.20%
CPPO	16	20550	75.00%	4	77.20%	12959 秒	87.50% 2
CPPO	16	75.20%	9657	93.75%	1	72.80%	8375
CPPO	16						
CPPO	16						

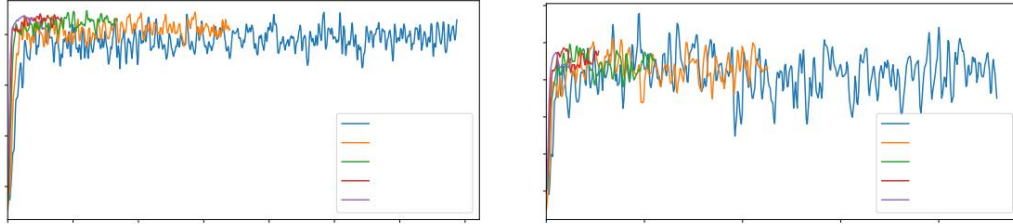


図 4: GRPO と CPPO のトレーニング中 (1 エポック) の報酬の比較。左: GSM8K データセット。右: Math データセット。

4.2.2 CPPO の高精度化に関する詳細な分析

セクション 4.2.1 の結果は、CPPO が高い剪定率でより良いパフォーマンスを達成することを示しています。この改善が、完了割り当て戦略によって可能になった、トレーニング ステップごとに処理される質問数の増加によるだけの可能性を排除するために、図 5 に示すように、トレーニング ステップあたりの質問数を同じにして CPPO と GRPO を比較します。

主な違いは、CPPO は最初に完了のグループを生成し、勾配更新のために上位 k 個の完了のみを保持するのに対し、GRPO は更新のために k 個の完了を直接生成することです。

それにもかかわらず、CPPO は一貫して GRPO を上回っており、精度の向上はステップごとに処理する質問数の増加によるものではなく、保持される完了の質の向上によるものであることが実証されています。

完了の質はトレーニングにおいて重要な役割を果たします。CPPO は、より大きなプールから絶対的優位性の高い完了を選択的に保持しますが、GRPO は、品質が異なる場合がありますが、直接生成された完了でモデルを更新します。これは、セクション 3.2 の完了貢献分析と一致しており、絶対的優位性の高い完了はトレーニングに効果的に貢献することを強調しています。グループ サイズが 16 の GRPO では、絶対的優位性の高い完了と低い完了の両方がトレーニングに使用されます。対照的に、剪定率が 87.50% の CPPO は、優位性の高い完了のみをトレーニングしますが、それでも優れたパフォーマンスを実現します。

これをよりよく理解するために、補完を 4 つのタイプに分類します (表 3 に示されています)。(1)

正しい形式と正しい回答- モデルが正確な補完を生成するようにガイドします。(2)

不正な形式と不正な回答- モデルが不正な補完を回避するのに役立ちます。(3)

正しい形式だが間違った回答の場合、モデルが誤って部分的に正しい回答を生成する可能性があります。(4) 形式が間違っているが正しい回答の場合、同様に学習にノイズが発生する可能性があります。

最初の 2 つのタイプは、明確なトレーニング シグナルを提供する高品質の補完です。ただし、後の 2 つのタイプは低品質の補完であり、その正の利点の値が小さいため、モデルが誤解してノイズが発生する可能性があります。すべての補完を無差別にトレーニングする GRPO とは異なり、CPPO は補完のブルーニングによってこれらの低品質の補完をフィルター処理し、より効率的な学習と全体的なパフォーマンスの向上につながります。低品質の補完が削除されるほど (ブルーニング率: 0.00% → 87.50%)、表 1 と表 2 に示すようにパフォーマンスが向上します。ただし、ブルーニング率が高すぎると、高品質の補完も破棄され、トレーニングの有効性が低下する可能性があります。これは、表 1 と表 2 に示すように、ブルーニング率が 93.75% のときの CPPO のパフォーマンスの低下からも明らかです。

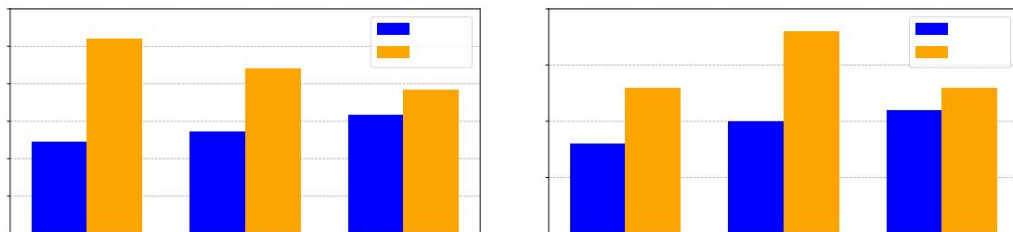


図 5: 評価精度の比較。左: GSM8K テスト サブセット上の Qwen2.5-1.5b-Instruct。

右: Qwen2.5-7b-MATH テスト サブセットの指示。ここで、 k は CPPO (または GRPO によって生成された) によって保持された完了量を示し、 b はトレーニング ステップごとの質問を表します。

表3: Qwen2.5-1.5B-Instruct [24]からの補完の例。

<p>GSM8K [3]。 「✓」 (「 」)は正しい (間違っ)形式または回答を示します。</p> <p>質問： 水タンクのゲージは、タンクが1/3の水で満たされていることを示しています。タンクを満たすには、16ガロンの水が必要です。 が追加されます。タンクが満杯になると何ガロンの水が入りますか？</p> <p>完了タイプ1: <考えてみましょう> タンクの全容量をCガロンで表しましょう。ゲージはタンクが1/3であることを示しています 満杯ということは、C ガロンの 1/3 ということです。タンクに 16 ガロン追加すると、タンクが完全に満たされます。 つまり、$C + 16 = C \cdot \frac{2}{3}$ ($\frac{1}{3} + \frac{1}{3} + \frac{1}{3}$は1に等しく、$\frac{1}{f} + \frac{1}{f} + \frac{1}{f}$は$\frac{3}{f}$に等しいため) 幾何学)。この式を簡略化すると、$C = 16 \cdot \frac{3}{2} = 24$ ガロンとなります。</think> <回答> タンクは満杯になると 24 ガロン入ります。</回答> フォーマット報酬: ✓ 精度報酬: ✓ 優位性: 2.01</p>
<p>完了タイプ2: <考えてみましょう> タンクの総容量をCガロンとしましょう。問題によると、 タンクが1/3いっぱいのとき、1/3ガロンの水が入っています。これに16ガロンを加えると、合計 満杯になると、 $C = 16 + 3C$ となります。この式は総容量C を求めるのに役立ちます。</think> フォーマット報酬: 正確さの報酬: 優位性: -1.26</p>
<p>完了タイプ3: <考えてみましょう>問題の情報に基づいて方程式を立てることができます。xを合計数とします タンクが満杯のときに何ガロン入るか。</think> <回答> x </回答> フォーマット報酬: ✓ 正確さの報酬: 優位性: 0.04</p>
<p>完了タイプ4: <think> 段階的に考えてみましょう: </think> 1. タンクは1/3がいっぱいだったことがわかっています。 2. タンクが満杯になったときの総容量を知るには、タンク内の既存の容量を掛ける必要があります。 タンクの容量を3倍にします (1タンクが満タンの場合、タンクの容量は3/3です) 。3. また、16ガロンを追加するとタンクが満タンになることもわかっています。 4. 16ガロンが満タンのタンクと現在のタンクの差に等しい場合、 量、この情報を使って現在の量をガロン単位で調べることができます。5. 16ガロンを追加した場合 最大容量を得るためには、現在の容量を全容量の4/3にする必要があります。 6. タンクの総容量をCとします (ガロン単位の整数です) 。つまり、現在の タンク内の量 = $(3/4)C$ 7. $3/4$を$3C/4$のように変形し、$(3C/4) = C -$ $(1/3)C = 2/3C$ であり、$2/3C = 16.8$ であることがわかっています。したがって、$2/3 \cdot C = 16.9$ です。したがって、C は $16 \cdot \frac{3}{2} \cdot 10$ である必要があります。 これは 24 ガロンに相当します。<answer> タンクは満杯になると 24 ガロン入ります。</answer> フォーマット報酬: 精度報酬: ✓ 優位性: 0.69</p>

表4: GSM8Kにおける異なるプルーニングメトリクスによるCPPOの評価。「最大」 / 「最小」プルーニング
絶対的な優位性が最高/最低の補完、「最大*」 / 「最小*」は生の
利点の値。「ランダム」はランダムな剪定を意味します。

	剪定メトリック	グループサイズの剪定率の精度		
	Qwen2.5-1.5B-指導	-	-	55.72%
	最大*	16	50.0%	73.32%
	最小*	16	50.0%	76.83%
	最大	16	50.0%	74.23%
	ランダム	16	50.0%	76.98%
	最小	16	50.0%	77.67%

表5 :CPPOの主要成分のアブレーション研究。実験はMath [7]で実施されている。
Qwen2.5-7B-Instruct [24]を使用する。

方法	グループ サイズ	剪定 レート	精度	時間	加速する 比率
GRPO	16	0.0%	75.20%	33902	50.0% 75.80%
+ 完了ブルーニング	16	27547	50.0%	75.20%	20550
+ 完了割り当て	16				

4.3 アブレーション研究

このセクションでは、まず剪定メトリックの設計に関するアブレーション研究を行い、次に評価します。
2つの主要モジュール (完了ブルーニングと完了割り当て) の有効性。

ブルーニングメトリックスのアブレーション研究。3.2節では、各完了の貢献を分析します。
政策目的関数の導関数を用いて分析すると、完了の影響は
政策モデルの訓練は、その優位性の絶対値に結びついており、絶対値が高いほど、
より強い訓練信号。この洞察に基づいて、絶対的な優位性を刈り込みとして採用します。
メトリックは、絶対的な優位性が最も低い補完を除外する。表4に示すように、「最小」
は最高のパフォーマンスを達成し、「最大」は最低のパフォーマンスを達成し、「ランダム」はその中間になります。
さらに、「最小*」と「最大*」は、生のアドバンテージ値に基づいて完了を絞り込む
絶対値ではなく「最小」よりもパフォーマンスが劣っており、絶対的な優位性値が
より効果的な剪定指標である。結果は第3.2節の分析と一致しており、さらに
絶対的な優位性の値に基づく剪定の有効性。

主要モジュールのアブレーション研究。表5に示すように、重要でない完了を破棄することで、
補完枝刈りモジュールはトレーニング効率を1.23倍向上させます。
補完ブルーニングとGPU並列コンピューティング機能によってもたらされる利点、補完
割り当て戦略により、トレーニング効率がさらに 1.65 倍に向上します。

4.4 ケーススタディ

表6に示すように、完了の違いを示すケーススタディを提供します。
GRPOとCPPOによって生成された。CPPOは訓練されたモデルの推論能力を維持できることがわかった。
モデル。CPPOはトレーニング時間を大幅に短縮し、最大8.32倍の加速を達成していることに注意してください。
GSM8Kデータセットについて。付録Cにはさらに多くのケーススタディが記載されています。

5 制限と今後の課題

GPUリソースが限られているため、比較的小規模なケースでのみCPPOの有効性を評価します。
モデル (70億未満)とデータセット。特に、CPPOの有効性に関する現在の評価は
MathやGSM8Kを含む数学データセットに焦点を当てています。今後は、(1)CPPOを評価することを計画しています。
より大規模なモデルと多様なデータセットでCPPOを探索する。(2)マルチモーダル大規模言語モデルにおけるCPPOの探索
強化学習。(3)CPPOをPPOなどの他の方策最適化アルゴリズムと組み合わせる[16]
およびREINFORCE++ [23]。

我々は、GPUリソースを豊富に持つ研究グループや業界が、より大規模なモデルやより広範なデータセッ
トでCPPOを実験的に検証する取り組みに参加してくれることを期待しています。
分野を前進させるだけでなく、AI コストを削減し、最終的には人類社会に利益をもたらします。

6 結論

本稿では、補完ブルーニングポリシー最適化 (CPPO)を導入し、
GRPOアルゴリズムによる推論モデルのトレーニング効率。選択的に枝刈りすることで
CPPOは、相対的な利点に基づいて補完を分類することで、計算オーバーヘッドを削減します。
モデルのパフォーマンスを犠牲にしています。さらに、当社のダイナミックな完了割り当て戦略は、
完了ブルーニングと GPU 並列処理の利点を活用し、トレーニング速度をさらに向上させます。
結果は、CPPOがGSM8Kで最大8.32倍、 Mathで3.51倍の高速化を達成したことを示しています。

表6: GRPOとCPPOによって生成された補完の比較。質問はGSM8K [3]データセットからサンプリングされています。補完を生成するために、異なるトレーニング方法を使用したQwen2.5-1.5B-Instruct [24]モデルを使用しています。「✓」と「✗」はそれぞれ正しい応答と間違った応答を示します。

質問:イアンは
40 フィートの長さの板を持っています。彼は 2 つの部分に分けられるように切断することにしました。長い部分は短い部分の 4 倍の長さです。長い部分の長さはどれくらいでしょうか。

GRPO 回答✓: <考えてください>
> 短いピースの長さを L とします。問題によると、長いピースは短いピースの 4 倍長いので、長いピースの長さは $4L$ です。ボードの全長は 2 つのピースの長さの合計で、40 フィートです。したがって、次の式が得られます: $L + 4L = 40$ 。これを簡略化すると、 $5L = 40$ になります。 L について解くと、 $L = 8$ フィートになります。長いピースは短いピースの 4 倍長いので、長いピースの長さは $4 \times 8 = 32$ フィートです。</考えてください> <答え> 32 </答え>

CPPO (50.00%) 回答✓: <考えてみましょう>
短いピースの長さを x とします。すると、長いピースは $4x$ になります。ボードの全長は 2 つのピースの長さの合計なので、 $x + 4x = 40$ です。これは $5x = 40$ と簡略化されます。 x について解くと、 $x = 8$ になります。したがって、長いピースは $4x = 4 \times 8 = 32$ フィートです。</考えてみましょう> <答え> 32 </答え>

CPPO (75.00%) 回答✓: <考えてみましょう>
短い部分の長さを x とします。すると、長い部分の長さは $4x$ になります。
ボードの全長は 2 つのピースの長さの合計なので、次の式が成り立ちます: $x + 4x = 40$ 。これは $5x = 40$ と簡略化されます。 x について解くと、 $x = 8$ になります。したがって、長いピースの長さは $4x = 4 \times 8 = 32$ フィートです。</think> <answer> 32 </answer>

CPPO (87.5%) 回答✓: <考えてください> 短
いピースの長さを x フィートとします。問題によると、長いピースは短いピースの 4 倍長いので、長いピースは $4x$ フィートです。ボードの全長は 2 つのピースの長さの合計で、40 フィートです。したがって、 $x + 4x = 40$ という式を書くことができます。左辺を簡略化すると、 $5x = 40$ になります。 x を求めるには、両辺を 5 で割るので、 $x = 8$ になります。つまり、短いピースの長さは 8 フィートです。長いピースは短いピースの長さの 4 倍なので、 $4 \times 8 = 32$ フィートです。</考えてください> <答え> 32 </答え>

GRPO (93.75%) 回答✓: <考えてみましょう>
短いピースの長さを x とします。問題によると、長いピースは短いピースの 4 倍長いので、長いピースの長さは $4x$ です。ボードの全長は 2 つのピースの長さの合計なので、 $x + 4x = 40$ という式を書くことができます。

同類項を結合すると、 $5x = 40$ になります。 x を求めるには、両辺を 5 で割ります: $x = 8$ 。したがって、短い部分の長さは 8 フィートです。長い部分は短い部分の長さの 4 倍なので、 $4 \times 8 = 32$ フィートです。したがって、長い部分の長さは 32 フィートです。</think> <answer> 32 </answer>

元の GRPO と比較して精度が維持されるか、または向上します。これらの結果は、CPPO が推論モデルのトレーニングを低コストで最適化するための実用的なソリューションであることを示しています。

7 謝辞

この研究は、国家優秀若手科学基金（No. 62025603）、中国国家自然科学基金（No. U21B2037、No. U22B2051、No. U23A20383、No. 62176222、No. 62176223、No. 62176226、No. 62072386、No. 62072387、No. 62072389、No. 62002305、No. 62272401）、中国福建省自然科学基金（No. 2021J06003、No. 2022J06001）の支援を受けて実施されました。

付録

報酬関数の具体的な設計

本稿では、GSM8K データセットの報酬コンポーネントを次のように定義します。

$$\begin{aligned} \text{Rformat}(oi) &= \begin{cases} 1, & \text{oi が正しい形式に従っている場合は} \\ 0, & \text{そうでない場合は} \end{cases} \\ \text{精度}(oi) &= \begin{cases} 2, & \text{oi が正解と直接一致する場合、} \\ 1.5, & \text{oi が通常の解析後に正解と一致する場合、} \\ 0, & \text{さもないと。} \end{cases} \end{aligned} \quad (11)$$

Math データセットの報酬コンポーネントは次のように定義されます。

$$\begin{aligned} \text{Rformat}(oi) &= \begin{cases} 1, & \text{oi が正しい形式に従っている場合は} \\ 0, & \text{そうでない場合は} \end{cases} \\ \text{精度}(oi) &= \begin{cases} 2, & \text{oi が正解と直接一致する場合は} \\ 0, & \text{それ以外の場合は} \end{cases} \end{aligned} \quad (12)$$

B 完了プルーフポリシー最適化のテンプレート

このセクションでは、GSM8K [3]とMath [7]データセットの補完プルーフポリシー最適化のテンプレートを提供します。テンプレートは表7と表8に示されています。

表7 :GSM8K [3]データセット用のCPPOのテンプレート。**質問**はデータセット内の質問に置き換えられます。

ユーザーとアシスタント間の会話。ユーザーが質問し、アシスタントがそれを解決します。アシスタントはまず頭の中で推論プロセスを考え、それからユーザーに答えを提供します。推論プロセスと答えはそれぞれ `<think>` タグと `<answer>` タグで囲まれます。つまり、`<think>` 推論プロセスはここに `</think>` `<n<answer>` 答えはここに `</answer>` です。ユーザー:**質問**。アシスタント:

表8 :CPPO for Math [7]データセットのテンプレート。**質問**はデータセット内の質問に置き換えられます。

ユーザーとアシスタントの会話。ユーザーが質問をすると、アシスタントがそれを解決します。アシスタントはまず頭の中で推論プロセスを考え、ユーザーに答えを提供します。答えは次の形式になります。「したがって、最終的な答えは`\boxed{ANSWER}`です。正しいことを願っています。」(引用符なし) ここで、ANSWERは問題を解決する最終的な数値または式です。推論プロセスと答えは、それぞれ`<think>` タグと`<answer>` タグで囲まれます。つまり、`<think>`推論プロセスはここに `</think>` `<n<answer>`答えはここに `</answer>`です。ユーザー:**質問**。アシスタント:

C その他のケーススタディ

表9に示すように、GRPOでは正解を出せない質問もありますが、剪定率の高いCPPOでは正解を出すことに成功しています。これは、CPPOではトレーニング中に絶対的優位性の低い補完を剪定することで、トレーニング済みモデルの推論能力を維持、または向上させることができることを示しています。

表9: GRPOとCPPOによって生成された補完の比較。質問はGSM8K [3]データセットからのもので、Qwen2.5-1.5B-Instruct [24]によって生成された。「✓」は正しい応答を示し、「✗」は誤った応答を示す。

質問: リーは以前、ジェラルドよりも 2 秒速く 400 メートル ハードルを走れました。しかし、ジェラルドは食生活を変え、スピードが 10% 向上しました。リーが 400 メートル ハードルを 38 秒で走っているのなら、食生活を改善したジェラルドは、400 メートルハードルを何秒で走れるでしょうか。

GRPO の回答 : <考えてください>
さい> 問題を段階的に分解してみましょう。リーは以前、400 メートルハードルをジェラルドより 2 秒速く走れました。つまり、リーが 38 秒で走る場合、ジェラルドは $38 + 2 = 40$ 秒で走るようになります。ジェラルドの食事療法により彼のスピードは 10% 向上したため、新しいスピードを計算する必要があります。ジェラルドの元のスピードが S 秒だった場合、向上したスピードは $S \times 1.1$ になります。リーのスピードはジェラルドの元のスピードより 2 秒速いことがわかっているので、次の式を立てることができます: $38 = S - 2$ 。

SIについて解くと、 $S = 38 + 2 = 40$ 秒となります。これで、ジェラルドの改善された速度を計算できます: $40 \times 1.1 = 44$ 秒。</think> <answer> 44 </answer>

CPPO (50.00%) 回答 : <考えてください>
問題を段階的に分解してみましょう。まず、リーが 400 メートル ハードルを走るのに以前どれくらいの時間がかかっていたかを調べる必要があります。リーは以前はジェラルドより 2 秒速く走ることができましたが、現在は 38 秒かかっているので、リーがハードルを走るのに以前どれくらいの時間がかかっていたかを調べることができます。食事を変える前にリーがハードルを走るのかかった時間を T 秒としましょう。

リーは現在 38 秒かかり、以前は 2 秒速かったので、 $T - 2 = 38$ となります。この方程式を解くと、 $T = 40$ 秒となります。これは、食事を変える前、リーは400 メートル ハードルを走るのに 40 秒かかっていたことを意味します。次に、食事を変える前にジェラルドが 400 メートル ハードルを走るのにどれくらいの時間がかかったかを調べる必要があります。リーは以前はジェラルドよりも 2 秒速く走れたので、 $T_{\text{Gerald}} = 40 - 2 = 38$ 秒となります。これは、食事を変える前、ジェラルドは 400 メートル ハードルを走るのに 38 秒かかっていたことを意味します。次に、ジェラルドが食事を改善することで 400 メートル ハードルをどれくらいの速さで走れるかを調べる必要があります。食事によって彼のスピードは 10% 向上したので、元のスピードの 110% を計算する必要があります。ジェラルドの元のスピードを S メートル/秒としましょう。食事を変える前は 400 メートルハードルを走るのに38 秒かかっていたので、 $S = \text{メートル/秒}$ となります。

$\frac{400}{38}$

食生活が改善されたことで、彼のスピードは $1.1 \times S$ になりました。 S の値を代入すると、 $1.1 \times \text{メートル/秒}$ になります。これを簡略化すると、 $= 11.57 \text{ メートル/秒}$ になります。</think> <answer> $\frac{440}{38}$ </answer>

CPPO (75.00%) 回答 : <考えてください> ま
ず、リーが 400 メートルハードルを走るのに以前どれくらいの時間がかかったかを判断する必要があります。リーは以前はジェラルドより 2 秒速く走ることができ、リーは 38 秒で走っているので、ジェラルドが走るのに以前かかっていた時間を計算できます。ジェラルドが以前かかっていた時間は $38 + 2 = 40$ 秒です。次に、ジェラルドが食事を改善することで 400 メートルハードルをどれくらい速く走れるかを計算する必要があります。彼のスピードは 10% 向上したので、元の時間の 110% を見つける必要があります。ジェラルドの元の時間は 40 秒なので、改善された時間は $40 \times 1.1 = 44$ 秒です。</考えてください> <答え> 44 </答え>

CPPO (87.5%) 回答: <考えてください> 問
題を段階的に分解してみましょう。まず、ジェラルドが400 メートルハードルを走る元のタイムを決定する必要があります。リーはジェラルドより 2 秒速く、リーは400 メートルハードルを 38 秒で走ります。したがって、ジェラルドの元のタイムは $38 + 2 = 40$ 秒です。次に、ジェラルドのダイエット後の速度の向上を計算する必要があります。ジェラルドのダイエットにより、彼の速度は 10% 向上しました。

ジェラルドの改善されたタイムを求めるには、40 秒の 10% を計算し、それを 40 秒から引く必要があります。40の 10% は $40 \times 0.10 = 4$ 秒です。したがって、ジェラルドの改善されたタイムは $40 - 4 = 36$ 秒です。</think> <answer> 36 </answer>

GRPO (93.75%) 回答/: <考えてください> リー
はかつて、400 メートル ハードルをジェラルドより 2 秒速く走れました。つまり、リーが 38 秒で走る場合、ジェラルドはかつて $38 + 2 = 40$ 秒で走っていました。
ジェラルドの食事療法により、彼のスピードは 10% 向上しました。ジェラルドが現在どのくらい速く走れるかを知るには、40 秒の 10% を計算し、それを 40 秒から引く必要があります。40 秒の 10% は $0.10 \times 40 = 4$ 秒です。したがって、ジェラルドの新しいタイムは $40 - 4 = 36$ 秒です。</think> <answer> 36 </answer>

参考文献

- [1] ジョシュ・アチアム、スティーブン・アドラー、サンディニ・アガルワル、ラマ・アフマド、イルゲ・アッカヤ、フロレンシア・レオニ・アレマン、デ
イオゴ・アルメイダ、ヤンコ・アルテンシュミット、サム・アルトマン、シャマル・アナドカット、他。Gpt-4技術レポート、2023年。
- [2] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, 他。クウェンのテ
クニカルレポートです。arXiv プレプリント arXiv:2309.16609, 2023。
- [3] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob
Hilton, Reiichiro Nakano, Christopher Hesse, John Schulman. 数学の文章問題を解くための検証者のトレーニング。arXiv プレプリント arXiv :
2110.14168, 2021年。
- [4] ハギングフェイス。オープンr1 :deepseek-r1の完全オープン複製、2025年1月。
- [5] Xinyu Guan, Li Lyna Zhang, Yifei Liu, Ning Shang, Youran Sun, Yi Zhu, Fan Yang, Mao Yang. rstar-math: 小規模 llms は、自己進化
した深い思考により数学的推論を習得できます。arXiv プレプリント arXiv:2501.04519, 2025。
- [6] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shiron Ma, Peiyi Wang, Xiao Bi,
他。Deepseek-r1: 強化学習を通じて llms の推論能力を奨励します。arXiv プレプリント arXiv:2501.12948, 2025。
- [7] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, Jacob Steinhardt. 数学データ
セットによる数学の問題解決の測定。NeurIPS, 2021年。
- [8] Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Heung-Yeung Shum, Xiangyu Zhang.
Open-reasoner-zero: ベースモデルで強化学習をスケーリングするためのオープンソースアプローチ。2025年。
- [9] Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex
Beutel, Alex Carney, 他。「Openai o1システムカード」arXiv プレプリント arXiv:2412.16720, 2024。
- [10] Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, Jon
Stoica. Livecodebench :コード用大規模言語モデルの全体的かつ汚染のない評価。arXiv プレプリント arXiv :2403.07974, 2024。
- [11] Yu Kang, Xianghui Sun, Liangyu Chen, Wei Zou. C3ot :効果を損なうことなく、より短い思考連鎖を生成する。arXiv プレプリント arXiv :
2412.11664, 2024。
- [12] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, Jon
Stoica. pagedattention を使用した大規模言語モデルの効率的なメモリ管理。SOSP, 2023年。
- [13] アメリカ数学協会。アメリカ数学コンテスト2023。
- [14] アメリカ数学協会。アメリカ招待数学試験、2024年。
- [15] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, Samuel R
Bowman. 「Gpqa :大学院レベルのGoogle対応Q&Aベンチマーク」COLM, 2024年。
- [16] ジョン・シュルマン、フィリップ・ウォルスキ、ブラフラ・ダリワル、アレック・ラドフォード、オレグ・クリモフ。近似ポリシー最適化アルゴリズム。arXiv
プレプリント arXiv :1707.06347, 2017年。
- [17] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, 他。
Deepseekmath: オープン言語モデルにおける数学的推論の限界を押し広げます。arXiv プレプリント arXiv:2402.03300, 2024。
- [18] Charlie Snell, Jaehoon Lee, Kelvin Xu, Aviral Kumar. LLMテスト時間計算を最適にスケーリングすることは、モデルパラメータをスケーリ
ングするよりも効果的である可能性がある。arXiv プレプリント arXiv :2408.03314, 2024。

- [19]ジェミニチーム、ロハン・アニル、セバスチャン・ボルゴー、ヨンファイ・ウー、ジャン＝バティスト・アレーラック、ジアフィ・ユ、ラドゥ・ソリカット、ヨハン・シャルクウィク、アンドリュー・M・ダイ、アンジャ・ハウト、他。ジェミニ：非常に有能なマルチモーダルモデルのファミリー、2023年。
- [20]キミチーム、アンガン・ドウ、ボフェイ・ガオ、ボウェイ・シン、チャンジウ・ジャン、チェン・チェン、チェン・リー、チェンジュン・シャオ、チェンチュアン・ドウ、チョンファ・リャオ、他。キミk1。5: llmsによる強化学習のスケールアップ。arXiv プレプリント arXiv:2501.12599,2025。
- [21] ユーゴ・トゥヴロン、ティボー・ラブリル、ゴージェ・イザカル、グザヴィエ・マルティネ、マリー＝アンヌ・ラショー、ティモ＝テ・ラクローワ、バティスト・ロジエール、ナマン・ゴヤル、エリック・ハンブロ、ファイサル・アズハルほか。Llama: オープンで効率的な基礎言語モデル、2023年。
- [22] Heming Xia, Yongqi Li, Chak Tou Leong, Wenjie Wang, Wenjie Li. Tokenskip: LLMsにおける制御可能な思考連鎖圧縮。arXiv プレプリント arXiv:2502.12067, 2025。
- [23] Tian Xie, Zitian Gao, Qingnan Ren, Haoming Luo, Yuqian Hon, Bryan Dai, Joey Zhou, Kai Qiu, Zhirong Wu, Chong Luo. Logic-rl: ルールベースの強化学習で llm 推論を解放します。arXiv プレプリント arXiv:2502.14768, 2025。
- [24] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, 他。クウェン2。5 テクニカルレポート。arXiv プレプリント arXiv:2412.15115, 2024。