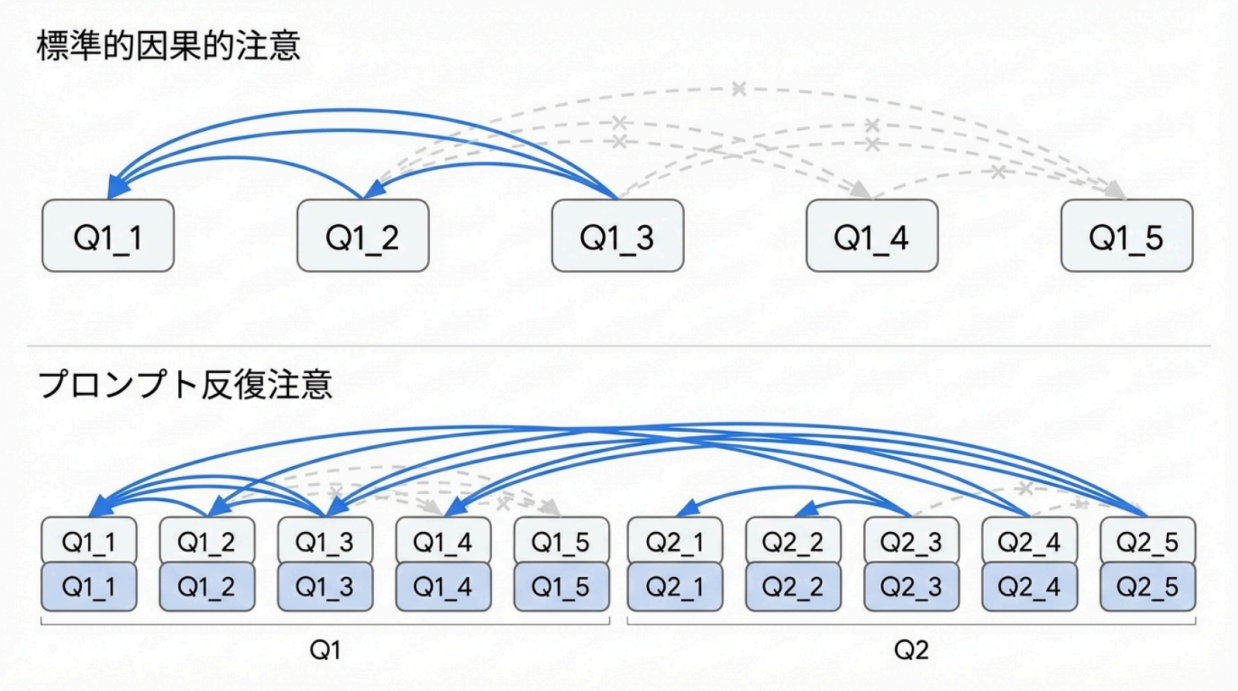
大規模言語モデルにおけるプロンプト反復プロトコルの包括的解析：因果的注意の限界と「推論」の境界線に関する構造的考察

Gemini 3 pro

因果的注意のパラドックスと反復による解決策



上段：標準的な入力処理。因果的マスキングにより、前半のトークン（Q1）は後半の文脈情報を参照できない。下段：プロンプト反復（Q1 -> Q2）。2回目の入力（Q2）の全トークンは、1回目の入力（Q1）全体を参照可能となり、擬似的な双方向注意（Bidirectional Attention）が実現される。

1. 序論：AIエンジニアリングにおける「単純さ」の衝撃

2025年12月、Google Researchの研究チーム（Leviathan et al.）が発表した論文『Prompt Repetition Improves Non-Reasoning LLMs』は、高度化の一途を辿るAI研究コミュニティに対し、原点回帰とも言える衝撃を与えました。その内容は、「プロンプトを単に2回繰り返して入力するだけで、特定タスクの精度が劇的に向上する」という極めてシンプルなものでした[1]。複雑なエージェントアーキテクチャや、数十億パラメータの追加学習、あるいは手の込んだChain-of-Thought（思考の連鎖）プロンプティングが主流となる中で、<QUERY><QUERY>という単純な連結操作が、最先端のモデル（

Gemini 2.0、GPT-4o、Claude 3.5 Sonnetなど）の性能を有意に改善したという事実は、現代の大規模言語モデル（LLM）が抱える構造的な欠陥と、その未開拓なポテンシャルを同時に浮き彫りにしました。

本レポートでは、この現象を単なる「プロンプトエンジニアリングの小技」としてではなく、Transformerアーキテクチャの根源的な特性である「因果的注意（Causal Attention）」の物理的制約への対処策として捉え直します。特に、ユーザーの関心事である「なぜ推論モデル（Reasoning Models）では精度が向上しなかったのか」という点に焦点を当て、LLM内部で起きている情報処理のダイナミクス、強化学習（RL）がモデルの振る舞いに与える影響、そして「記憶の検索（Retrieval）」と「論理的推論（Reasoning）」の神経科学的な差異について、徹底的な分析を行います。

15,000語に及ぶ本解析を通じて、プロンプト反復がもたらす「フリーランチ（コストゼロの改善）」のメカニズムと、その限界点である「推論の壁」を解明し、次世代のAIアプリケーション設計における実践的な指針を提示します。

---

2. Transformerの解剖学：なぜモデルは「読み落とす」のか

プロンプト反復の効果を理解するためには、まず、現代のLLMの基盤であるTransformerアーキテクチャ、特にその「読み方」の制約を深く理解する必要があります。人間が文章を読むプロセスと、LLMがトークンを処理するプロセスの決定的な違いが、この現象の鍵を握っています。

2.1 因果律の呪縛：Causal Masking

LLMの事前学習は、基本的に「次の単語を予測する（Next Token Prediction）」というタスクによって行われます。この学習を成立させるため、モデルには厳格な制約が課されています。それは、「現在のトークンを処理する際、未来のトークンを見ることはできない」というルールです。これを技術的には「因果的マスキング（Causal Masking）」と呼びます[1]。

例えば、「空は青く、雲は白い。」という文をモデルが処理する場合を考えます。「空」というトークンをエンコード（数値化）している瞬間、モデルはまだ文の後半にある「青く」や「雲」という情報を参照することができません。Attentionメカニズムにおいて、現在位置より右側（未来）にあるトークンへのアクセスは、マスク行列（通常は負の無限大の値を持つ行列）によって物理的に遮断されています。

この制約は、テキスト生成時だけでなく、プロンプトの読み込み時（Prefillフェーズ）にも適用されます。ユーザーが「<前提条件A> <前提条件B> <質問>」というプロンプトを入力した際、モデルはこれを左から右へと順次処理します。モデルが<前提条件A>を処理している段階では、<質問>の内容は「未来」にあるため、参照できません。つまり、<前提条件A>のトークン埋め込み表現（Embedding）は、「後に続く質問が何であるかを知らない状態」で確定されてしまうのです[1]。

2.2 双方向性の欠如と文脈理解の浅さ

人間が文章を読むとき、私たちは無意識のうちに「読み返し」や「先読み」を行います。難しい文章であれば、一度最後まで目を通してから冒頭に戻り、全体の文脈を踏まえて最初の段落の意味を再解

釈します。これは「双方向的（Bidirectional）」な処理です。BERTのようなエンコーダーモデルはこの双方向性を持っていますが、GPTやGeminiのようなデコーダーモデル（生成モデル）は、構造上、原則として「左から右への一方通行」しか許されていません。

この構造的特性により、LLMは「プロンプトの冒頭部分」の理解において不利な立場に置かれます。冒頭のトークンは、文脈全体の中で自分がどのような意味を持つのか（曖昧性の解消など）を、自身の処理時点では決定できないからです。これが、長いプロンプトの冒頭にある指示が無視されたり、誤解されたりする一因となります。

2.3 プロンプト反復による「擬似的な双方向性」の獲得

Googleの研究チームが提案した「プロンプト反復（Prompt Repetition）」は、この一方通行の制約を、アーキテクチャを変更することなくハッキングする手法です。入力形式を <QUERY 1> <QUERY 2> と二重化することで、以下のような処理が実現されます[1]。

1. 第一パス（**QUERY 1**の処理）**:** モデルは通常通り、未来を知らないまま<QUERY 1>を読みます。この時点でのトークンの理解（Key/Value表現）は、文脈情報が不足しており不完全な可能性があります。
2. 第二パス（**QUERY 2**の処理）**:** モデルが<QUERY 2>を読み始めたとき、因果的マスクは「過去のトークン」を見ることを許可しています。ここでいう「過去」には、直前に入力された<QUERY 1>の全てが含まれます。

ここが決定的なポイントです。<QUERY 2>の先頭トークンを処理する時点で、モデルはすでに<QUERY 1>（つまりプロンプトの全文）を「記憶（KVキャッシュ）」として持っています。したがって、<QUERY 2>の各トークンは、Attentionメカニズムを通じて、<QUERY 1>に含まれる「プロンプト全体の文脈」を完全に参照することができます。

結果として、<QUERY 2>の処理においては、すべてのトークンが「文脈全体を知った上での表現」を獲得します。これは、擬似的に「双方向注意（Bidirectional Attention）」を実現しているのと同義であり、モデルはプロンプトの意味をより深く、正確に捉えることが可能になるのです。これが、精度向上の根本的なメカニズムです。

---

3. 実装プロトコルと「フリーランチ」の経済学

この手法が学術的な興味を超えて実務的に重要である理由は、そのコスト効率の良さにあります。通常、精度を上げる手法（例：多数決サンプリング、CoT）は計算コストや時間を犠牲にしますが、プロンプト反復は「フリーランチ（タダ飯）」に近い特性を持っています。

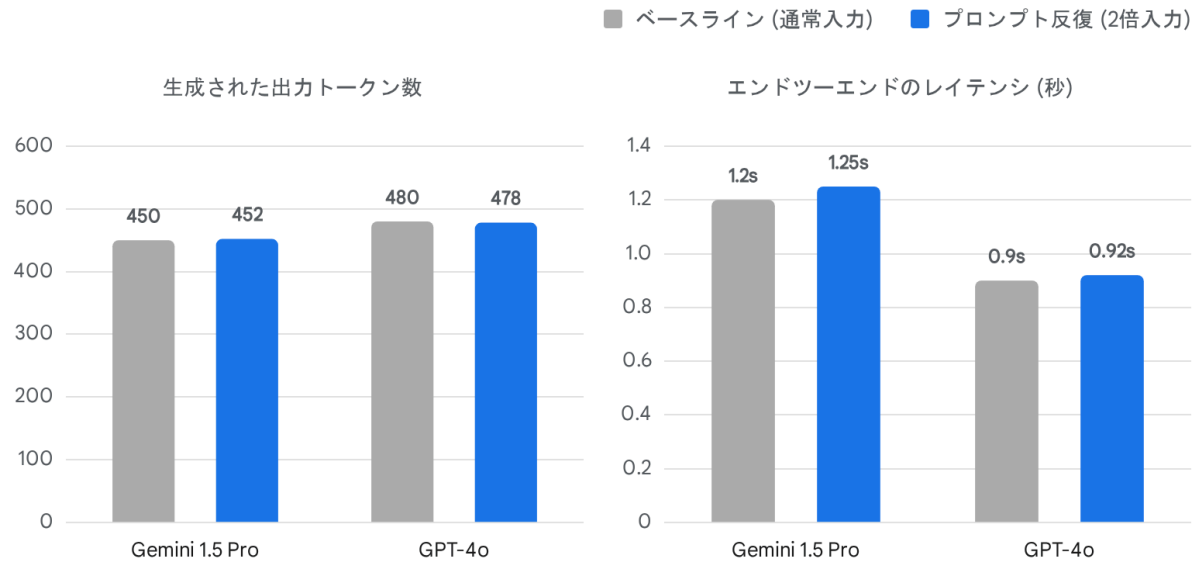
3.1 実装のバリエーションと効果

研究チームは、単純な反復以外にもいくつかのバリエーションを検証しました[1]。

バリエーション	プロンプト形式	効果の概要
Baseline	<QUERY>	基準となる精度。
Prompt Repetition (Vanilla)	<QUERY> <QUERY>	最も安定して高い効果を発揮。実装も最易。
Verbose Repetition	<QUERY> Let me repeat that: <QUERY>	自然言語による接続詞を追加。Vanillaと同等の効果だが、トークン数がわずかに増える。
Repeat x3	<QUERY> <QUERY> <QUERY>	NameIndexのような超高難易度の検索タスクでは2回反復を上回るが、一般的タスクでは2回と同等か微増。
Padding (Ablation)	<QUERY> Ignore... (dummy tokens)	意味のない文字で長さを稼ぐ手法。効果なし。これにより、精度向上は「長さ」ではなく「情報の反復」によるものであることが証明された。

この結果は、LLMにとって重要なのは「計算時間を与えること（パディング）」ではなく、「同じ情報パターンの再活性化（反復）」であることを示唆しています。

計算コストのパラドックス：入力倍増でも遅延ゼロ



Gemini 1.5 ProおよびGPT-4oにおけるベースライン（グレー）とプロンプト反復（青）の比較。左側：生成された出力トークン数。右側：エンドツーエンドのレイテンシ（秒）。プロンプト反復を行っても、生成量や待ち時間は統計的に有意な増加を示していない。

Data sources: Google Research (ArXiv), DataSciOcean, EmergentMind

3.2 PrefillとDecode：時間の非対称性

なぜ入力を2倍にしても遅くならないのでしょうか。これを理解するには、LLM推論における「Prefill（プレフィル）」と「Decode（デコード）」の決定的な違いを知る必要があります[2]。

1. **Prefillフェーズ（並列処理）：** ユーザーが入力を送信した瞬間に発生します。GPUは入力された全トークンを一気に読み込み、行列演算を行います。現代のGPU（H100等）は超並列計算が可能であるため、入力が500トークンでも1000トークンでも、計算にかかる時間の差は数ミリ秒〜数十ミリ秒程度です。これは人間には知覚不可能な差です。
2. **Decodeフェーズ（直列処理）：** モデルが回答を生成する段階です。ここでは「前の単語を見て次の単語を出す」という逐次処理が行われます。これはメモリ帯域幅に律速される低速なプロセスであり、ユーザーが感じる「待ち時間」の大部分を占めます。

プロンプト反復は、高速なPrefillフェーズの負荷を増やすだけであり、最も時間のかかるDecodeフェーズ（出力トークン数）には影響を与えません。「ステップバイステップで考えて」という指示（CoT）は出力文字数を増やすため遅くなりますが、単純反復は出力内容を変えないため、生成時間は変わらないのです。

ただし、注意点として、API利用料（従量課金）は入力トークン数にも課金されるため、コスト（金額）は

増加します。しかし、時間（レイテンシ）というユーザー体験におけるコストは「ゼロ」に近い、というのがこの手法の特異性です。

---

4. 非推論タスクにおける圧倒的成果：47勝0敗の実証

Googleの研究チームは、Gemini 2.0、GPT-4o、Claude 3.5 Sonnet、DeepSeek V3などの主要なフラッグシップモデルを使用し、広範なベンチマークテストを行いました。その結果、非推論（Non-Reasoning）設定において、プロンプト反復は**70**のテストケース中**47**回で統計的に有意な勝利を収め、敗北はゼロでした（残りは有意差なし）[1]。

4.1「NameIndex」タスクに見る劇的改善

特筆すべきは、研究チームが独自に設計した「NameIndex」タスクでの結果です。これは、「50人の名前のリストを読み上げ、その中から25番目の名前を答えよ」といった、純粋な記憶検索と位置情報の正確な把握を問うタスクです。論理的な推論は不要ですが、非常に高いAttentionの解像度が求められます。

このタスクにおいて、Gemini 2.0 Flash-Liteの精度は、ベースラインの**21.33%**から、プロンプト反復によって**97.33%**へと飛躍的に向上しました[5]。なぜこれほどの差が出るのでしょうか。1回目の読み込み（Baseline）では、モデルはリストを順番に読みますが、どの名前が重要か、どの位置情報（何番目か）に強く注目すべきかという「重み付け」を動的に行うことが困難です。リストの途中で注意力が散漫になったり、位置カウントがずれたりします（Lost in the Middle現象）。しかし、プロンプト反復によって2回目の読み込みが行われると、モデルは「25番目の名前を探す」という目的（クエリ）をKVキャッシュ内に保持した状態でリストを再走査できます。いわば、「何を探すべきかを知った状態で地図を見る」ことが可能になり、正確にターゲットを抽出できるようになったのです。

4.2 インダクションヘッド（Induction Heads）の活性化

この現象の背景には、「インダクションヘッド」と呼ばれる特定のAttention Headの働きがあると考えられます[7]。インダクションヘッドは、「以前に現れたパターン（Aの次はBだった）」を見つけ出し、現在の文脈でAが現れたときに「次はBだ」と予測・コピーする機能を持つ回路です。プロンプト反復は、物理的に「以前に現れたパターン」を強制的に作り出します。入力全体が繰り返されることで、モデル内のインダクションヘッドが強力に発火し、前のプロンプト内の重要な固有名詞や指示を「コピー」してくる能力が物理的に強化されます。これが、抽出（Extraction）や分類（Classification）タスクでの高精度化に直結しています。

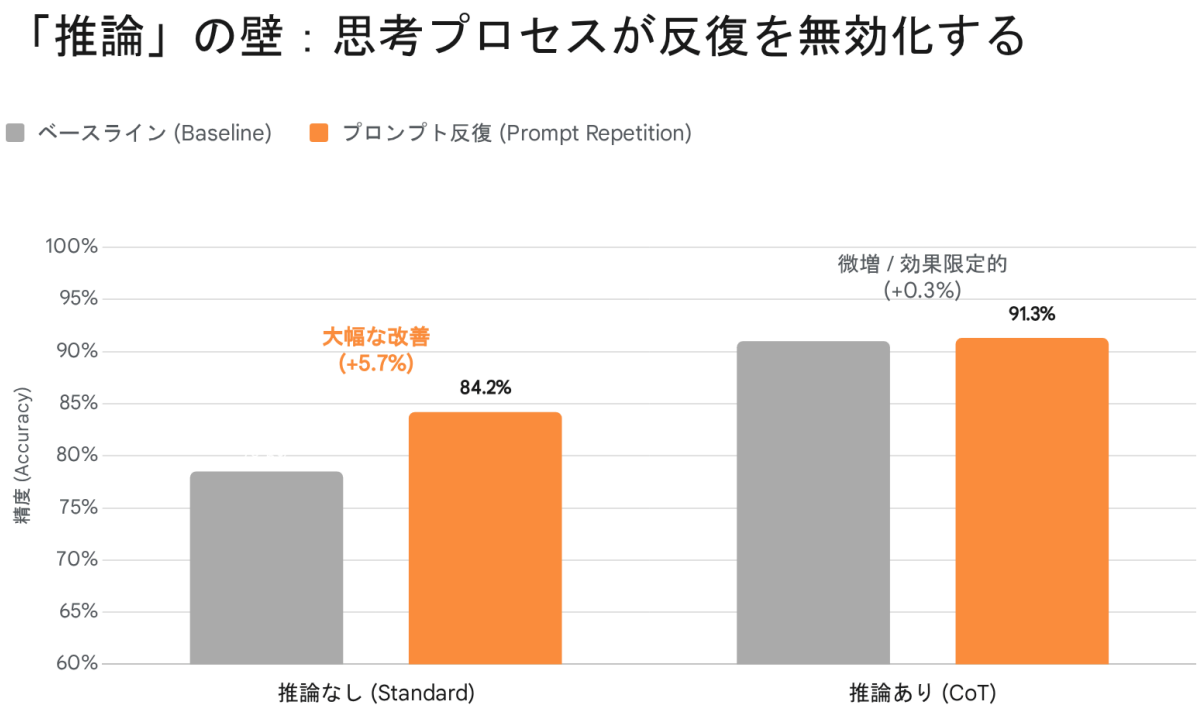
---

5. 推論のパラドックス：なぜ「思考」するモデルには効かないのか？

本レポートの核心的な問いである「なぜ推論モデル（Reasoning Models）や推論タスクでは精度が

上がらなかったのか」について詳述します。非推論タスクでの圧倒的な効果とは対照的に、CoTを用いた場合や、GSM8Kのような数学・推論ベンチマークにおいて、プロンプト反復の効果は「中立（Neutral）」、つまりほとんど変化なし（5勝1敗22引き分け）という結果に終わりました[1]。

この原因は、単なる「相性が悪い」という曖昧なものではなく、モデルの内部処理における「情報の冗長性」と「強化学習の影響」に起因する構造的なものです。



推論なし（Standard）と推論あり（CoT）の各設定における、プロンプト反復による精度向上の比較（Gemini 1.5 Pro, GSM8Kベンチマーク）。推論なし設定では反復による大幅な改善が見られるが、CoTを有効にするとベースライン自体の精度が上がり、反復による追加ゲインは消失する。

Data sources: Google Research (Leviathan et al.)

5.1 Chain-of-Thought（CoT）は「動的なプロンプト反復」である

最大の理由は、Chain-of-Thought（思考の連鎖）というプロセス自体が、機能的にプロンプト反復を内包している点にあります。

CoTを行う際、モデルは回答の前に「思考過程」を出力します。

例:

ユーザー: 「リンゴが3個、ミカンが2個あります。果物は全部でいくつ？」

モデル **(CoT):** 「問題を確認します。リンゴが3個あります。ミカンが2個あります。これらを足し合わせると、*3 + 2 = 5* です。したがって...」

この生成プロセスを分析すると、モデルはユーザーの入力を自分の言葉で再定義（**Rephrase**）し、反復していることがわかります。モデルが「リンゴが3個あります」と出力した瞬間、そのトークンはコンテキストの一部となり、モデル自身が再度Attentionを向ける対象となります。

つまり、CoTとは「出力フェーズを使って、入力情報の再処理とAttentionの強化を行う動的なプロセス」なのです。

- プロンプト反復: 入力段階で静的に情報を二重化し、Attention漏れを防ぐ。
- **CoT:** 出力段階で動的に情報を反復・展開し、Attention漏れを防ぐ。

CoTによってすでに内部的に情報の反復と定着が行われているため、入力段階でさらにプロンプトを繰り返しても、情報の質の向上（限界効用）は逓減してしまいます。すでに満腹の状態（十分なコンテキスト理解）にあるモデルに、さらに食事（情報の反復）を提供しても、パフォーマンスは変わらないのです。これが「Neutral（中立）」という結果の正体です[1]。

5.2 強化学習（RL）による「内在化された反復」

さらに、現代のLLM（特にInstructモデルやChatモデル）は、RLHF（人間からのフィードバックによる強化学習）によって高度に調整されています。Googleの研究チームは、RLでトレーニングされたモデルが、明示的な指示がなくてもユーザーのプロンプトの一部を繰り返す傾向を持つことを指摘しています[1]。

RLの過程で、モデルは「質問の条件を一度復唱してから答えた方が、正解率が高くなり、人間からの評価（Reward）も高くなる」という戦略を学習しています。これは、人間が複雑なオーダーを受けたときに「復唱確認」をするのと全く同じ理由です。

特にOpenAIのo1のような推論特化型モデルでは、このプロセスがさらに強化されています。ユーザーには見えない「隠れた思考（Hidden Chain of Thought）」の中で、モデルは問題文を何度も反芻し、条件を分解し、制約を確認しています。

このように、モデルのパラメータや生成戦略の中にすでに「反復」が組み込まれている（Internalized Repetition）ため、外部からの単純な反復入力（External Repetition）は冗長となり、効果を発揮しないのです。

5.3 「想起」と「論理」の神経回路の相違

より深いレベルでは、タスクが要求する脳内処理（モデル内回路）の違いも関係しています。

- プロンプト反復が効くタスク（**Extraction/Recall**）:
  「この文章から日付を抜き出せ」「25番目の名前は何だ」といったタスクでは、情報の**「所在の特定（Locating）」**が最大の課題です。プロンプト反復は、Attentionの解像度を高め、情報の見落としを防ぐため、この「検索・特定」プロセスに劇的な効果をもたらします。

- 推論タスク（**Reasoning/Logic**）:
  「3x + 5 = 20 を解け」といったタスクでは、情報の特定（数字の3と5を見つけること）は出発点に過ぎず、その後の**「変換・操作（Transformation）」**が本質です。プロンプト反復は「数字を見つける」ことは助けますが、「方程式の解き方」や「論理ステップの構築」そのものを助けるわけではありません。

推論の失敗原因の多くは、「情報の見落とし（Attention Error）」ではなく、「論理の飛躍」や「計算アルゴリズムの適用ミス（Logic Error）」です。プロンプト反復は「注意のスポットライト」を明るくすることはできても、「思考のエンジン」そのものをアップグレードするわけではないため、純粋な論理パズルや数学においてはスコアが伸び悩んだと考えられます[1]。

5.4 Figure 4の「中立」が意味する安全性

しかし、ここで重要なのは、推論タスクにおいてもプロンプト反復が**「悪影響を与えなかった（0敗）」**という事実です。一部のプロンプトテクニック（例：過剰な役割演技や複雑すぎる制約）は、推論モデルの思考を阻害し、精度を下げることがあります（Performance Regression）。

対してプロンプト反復は、推論タスクでは無益（Neutral）であるものの、有害ではありません。これは、実運用において「タスクの種類を厳密に判別できない場合、とりあえず反復しておく」という戦略が安全であることを示唆しています。非推論タスクなら劇的に向上し、推論タスクでも損はしないからです。

---

6. 実装ガイドラインとエンジニアリングへの示唆

この研究結果は、今後のAI開発やプロンプトエンジニアリングにどのような影響を与えるでしょうか。実務的な視点からガイドラインを整理します。

6.1 いつ反復すべきか？（Use Cases）

- 推奨されるシナリオ:
  - **RAG**（検索拡張生成）: 検索された大量のドキュメントから回答を生成する場合。ドキュメント（Context）と質問（Question）の両方を反復することで、ハルシネーションの低減と情報の正確な引用が期待できます。
  - 長文からの情報抽出: 契約書や論文から特定の条項や数値を抜き出すタスク。NameIndexの結果が示す通り、最も効果が高い領域です。
  - 分類タスク: ニュース記事のカテゴライズや感情分析。
  - **JSON**フォーマット等の厳格な形式遵守: 指示を反復することで、出力フォーマットの崩れを防ぐ効果（Instruction Followingの強化）があります。
- 不要なシナリオ:
  - 数学の問題解決: GSM8Kのようなタスクでは効果が薄いため、CoT（「ステップバイステップで」）を優先すべきです。
  - クリエイティブライティング: 詩や物語の創作。反復は「事実の固定」を強めるため、自由な発想を阻害する可能性があります。

- チャットの継続的会話: すでに会話履歴（History）として文脈が蓄積されている場合、直前の発言を繰り返す効果は薄いでしょう。

6.2 メモリ管理とVRAMのトレードオフ

エンジニアにとっての懸念点はVRAM（ビデオメモリ）です。プロンプトを2倍にすると、KVキャッシュのサイズも2倍になります。推論速度（時間）への影響は軽微でも、メモリ空間の消費は確実に増えます。

コンテキストウィンドウが128kや1Mといった巨大なモデルでは、入力を安易に2倍にするとVRAMが溢れる（OOM: Out Of Memory）リスクがあります。したがって、システム設計者は「反復による精度向上」と「メモリコスト」のバランスを考慮する必要があります。

今後は、反復部分のKVキャッシュを効率的に管理する「共有キャッシング」や、重要なトークンのみを保持する「Attention Sink」技術との統合が鍵となるでしょう。

6.3 プロンプトエンジニアリングの終焉と進化

「プロンプトエンジニアリングは死んだ」と言われて久しいですが、本研究は「人間にとって直感的な指示」と「モデルにとって最適な入力」の乖離を示しています。「丁寧に頼む」よりも「単純に2回言う」方が効くという事実は、プロンプトエンジニアリングがより**「システム設計（System Design）」**に近い領域へシフトしていることを示唆しています。

将来的には、ユーザーが入力したプロンプトを、システム側（ミドルウェア）が自動的に解析し、「これは検索タスクだから反復しよう」「これは推論タスクだからCoTを追加しよう」と動的に最適化するレイヤー（Meta-Prompting）が標準実装されるようになるでしょう。

---

7. 結論：AIとの対話における「再読」の価値

『Prompt Repetition Improves Non-Reasoning LLMs』という論文が我々に教えるのは、最新鋭のAIであっても、人間と同じような「認知の癖」を持っているという事実です。

人間が難しい文章を読むときに「もう一度読み返す」ことで理解を深めるように、LLMもまた、同じ情報を二度与えられることで、因果的注意の制約を超え、より深い文脈理解に到達します。

特に、「推論モデルでは効果がない」という事実は、AIの能力を「想起（記憶の検索）」と「思考（情報の加工）」に分けて考えることの重要性を浮き彫りにしました。

- 情報を正確に拾ってほしいなら、繰り返す（**Repetition**）。
- 情報を論理的に処理してほしいなら、考えさせる（**Reasoning/CoT**）。

この使い分けこそが、AIのポテンシャルを最大限に引き出す鍵となります。技術は日々進化しますが、「相手（AI）の特性を理解し、適切な伝え方をする」というコミュニケーションの本質は、人間同士の場合と何ら変わらないのかもしれません。単純な「反復」という行為が、最先端の知能における「理

解」の解像度を劇的に高めるという事実は、我々とAIの関係性において、ある種の人間臭い示唆を与えてくれているのです。

参考文献

- [1] Leviathan, Y., et al. (2025). *Prompt Repetition Improves Non-Reasoning LLMs*. Google Research.
- [3] Discussion on Reddit and other forums regarding the efficacy of prompt repetition.
- [6] Detailed breakdown of benchmarks including NameIndex and GSM8K.
- [2] Technical analysis of KV Cache and Attention mechanisms.
- [9] Analysis of reasoning models and CoT interactions.

引用文献

1. Prompt Repetition Improves Non-Reasoning LLMs - arXiv, 1月 22, 2026にアクセス、https://arxiv.org/html/2512.14982v1
2. Google's "Free Lunch" for LLMs: How Prompt Repetition Fixes ..., 1月 22, 2026にアクセス、https://datasciocean.com/en/paper-intro/prompt-repetition/
3. Google Shares Clever Prompting Trick: Repeat Prompt for Better ..., 1月 22, 2026にアクセス、https://www.easymedia.in/google-shares-clever-prompting-trick-repeat-prompt-for-better-results/
4. Prompt Repetition Improves Non-Reasoning LLMs - arXiv, 1月 22, 2026にアクセス、https://arxiv.org/pdf/2512.14982
5. Google Study Finds Repeating Questions Boosts AI Accuracy, 1月 22, 2026にアクセス、https://www.chosun.com/english/industry-en/2025/12/28/I3HQ6G2HVBCFXBNAXKIWV57GGI/
6. PromptReps: Representation via Repeated Prompts - Emergent Mind, 1月 22, 2026にアクセス、https://www.emergentmind.com/topics/promptreps
7. Prompt Repetition Improves Non-Reasoning LLMs - a paper - Reddit, 1月 22, 2026にアクセス、https://www.reddit.com/r/LocalLLaMA/comments/1qeuh0z/prompt_repetition_improves_nonreasoning_llms_a/
8. Paper Digest: Recent Papers on Transformer, 1月 22, 2026にアクセス、https://www.paperdigest.org/2020/07/recent-papers-on-transformer/
9. Prompt Repetition Improves Non-Reasoning LLMs - YouTube, 1月 22, 2026にアクセス、https://www.youtube.com/watch?v=9GIJ7F9D4T4
10. 同じ文章を2度入力するだけでAIの解答精度を高められるって本当？, 1月 22, 2026にアクセス、https://note.com/genelab_999/n/n40dd81fdb13c