自然言語による LLM 生成途中介入

林部 祐太
フリー
yuta@hayashibe.jp

概要

LLM がシステムプロンプト等により特定の応答パターンから抜け出せない状態を応答固着と呼ぶ．本研究では応答固着の突破のため，生成途中に介入文を挿入する手法を提案する．推薦タスクを用いた 2 モデルでの比較実験では，「それは」で別の候補を推薦させる文で介入した場合，モデルによって固着を突破する割合が大きく異なることが分かった．また，事前に用意した固定文を用いる方法と文脈に応じて LLM で介入文を生成させる方法を比較したところ，前者の方が有効であった．

1 はじめに

大規模言語モデル（LLM）をサービスに組み込む際，特定の結論へ誘導するようなシステムプロンプトや追加学習（fine-tuning）を施すことは，事業上は起こりうる．しかしその結果，製品比較や欠点に関する質問に対しても一貫して特定の立場から回答し，別の選択肢や反対意見を提示しない場合，ユーザーは偏った情報を継続的に受け取るおそれがある．本研究では，システムプロンプト等により LLM が特定の応答パターンから抜け出せない状態を**応答固着**と呼ぶ．

応答固着に気づかず LLM を利用しているユーザーは不便なだけではなく，自分が誘導されていることに気づかないまま，偏った情報を受け取り続ける可能性がある．さらに，特定の思想や見解を持つように fine-tuning された LLM が，ユーザーの問いかけに対して一貫して特定の立場からのみ回答することも考えられる．応答固着の存在を検出し，必要に応じて突破する技術は，LLM の透明性とユーザーの自律性を確保する上で重要である．

応答固着に対し「別の候補も教えて」と事前に指示する手法は，既存指示と衝突して無視されやすい．そこで本研究は，**生成途中**の応答に対し，文脈として介入文を挿入して方向転換させる**生成途中介入**を提案する．本手法はモデル内部へのアクセスを要さず，API 環境でも適用可能である．

推薦タスクを用いた実験の結果，介入効果には強いモデル依存性が観測された．特に，最も強固な禁止指示に対しては，高い突破率を示すモデルと，文脈を無視して固着し続けるモデルがあった．また，文脈適応的な生成文よりも固定文の方が有効であるという反直感的な知見も得られた．

本研究の貢献は以下の 3 点である：

1. **生成途中介入の提案**：勾配や内部状態へのアクセスを必要とせず，API 環境でも適用可能な，生成テキストへの介入による応答固着の緩和手法を提案した．
2. **介入手法の特性分析**：文脈適応的な生成文よりも，強い強制力を持つ固定文の方が有効であること，および assertive な介入が最も高い突破率を達成することを確認した．
3. **固着メカニズムの考察**：モデルによってシステムプロンプトが直前の文脈情報を無視する現象を発見し，指示タイプ（文脈的誘導 vs 明示的再質問）による効果差を提示した．

2 関連研究

2.1 推論時の生成制御

LLM の出力を追加学習なしに制御する手法として，デコーディング時に内部状態や出力分布を操作するアプローチが研究されてきた．これらは必要なアクセス権限の観点から，(i) 隠れ状態や勾配へのアクセスを要する手法と，(ii) 出力分布（logits）へのアクセスで動作する手法に大別できる．

(i) PPLM[1] は，属性分類器の勾配を用いて LLM の隠れ状態（過去の Key-Value 対）を更新し，話題・感情・毒性などの属性を制御する．生成する各トークンにおいて，目標属性の対数尤度を高める方向に隠れ状態を摂動させつつ，元の言語モデルの分布か

ら逸脱しないよう制約をかけることで，流暢さを保ちながら属性誘導を実現する．しかし，モデル内部での勾配計算（逆伝播）が必要であり，内部状態が公開されていない LLM は適用できない．

(ii) FUDGE[2] は，部分系列から「将来，完成したシーケンスが目的の属性を満たすか」を予測する判別器を学習し，その出力確率で次トークン分布をベイズ則に基づき調整する．勾配計算を必要とせず，出力のトークン確率さえ取得できれば動作するため，PPLM より適用範囲が広い．また，DExperts[3] は，専門家 LM（expert）と反専門家 LM（anti-expert）の出力分布を組み合わせ，毒性低減や話題制御を実現する．ベースモデルの対数確率に対して専門家・反専門家の対数確率差を加算する形で分布を調整するため，モデル自体を再学習させる必要がない．

これらは「推論時に生成の方向を変える」という点で本研究と類似するが，勾配が必要ない FUDGE や DExperts でも，各トークン生成時に出力確率分布へのアクセスと逐次的な制御が必要となる．

2.2 生成途中への介入

山下ら [4] は，生成 LLM と評価 LLM を組み合わせ，次トークン確率が閾値を下回る「息継ぎ」点で評価 LLM が有害と判断した場合はトークン列を巻き戻して別候補に差し替える手法を提案した．

RAIN[5] も同様に巻き戻しを用いるが，外部の評価 LLM ではなくモデル自身の自己評価に基づき，木探索的に複数の生成経路を探索する．追加学習なしで無害性が向上したが，通常の推論に比べ約 4 倍の時間を要したと報告している．

Progressive Self-Reflection (PSR)[6] は，生成を一定数のトークン毎に，その時点の出力が有害か無害かをモデル自身に問いかけて自己反省させる．有害と判断された場合は安全な箇所まで巻き戻して再生成を行う．入力のリスクに応じて反省回数を動的に調整する適応型予測器を導入し，効率と安全性の両立を図っている．

生成の外側で入出力を監視するガードレールも提案されている．Llama Guard[7] は LLM の入出力を安全カテゴリに分類する分類器であり，NeMo Guardrails[8] はプログラマブルな制御機構を提供する．ただし，入力段階での拒否判定が中心で，本来有用な応答まで抑制されるトレードオフや，生成途中での動的な修正ができないという限界がある．

LLM の拒否傾向を攻撃的に突破する研究には，Don't Say No (DSN)[9] がある．DSN は，「Sorry」「I cannot」といった拒否語彙の生成確率を抑制する損失を最適化に組み込み，さらに応答冒頭を重視するコサイン減衰を導入することで，高い攻撃成功率を達成した．また，オープンソースモデルで作成した敵対的サフィックスが，GPT-4 や Claude といった商用モデルにも転移できたと報告している．

3 自然言語による LLM 生成途中介入

3.1 生成途中介入の定義

生成途中介入は，LLM の応答生成を途中で停止し，介入文を挿入した上で生成を継続させる操作と定義する．例えば，LLM が「A をおすすめします」と生成した時点で停止し，「という見方もありますが,」を挿入し，LLM にはその続きを生成させる．LLM はこの文脈を踏まえて続きを生成するため，「B も検討に値します」のように異なる方向へ展開する可能性が期待できる．これは生成の任意の時点で発動でき，必要に応じて複数回行うこともできる．

3.2 生成途中介入の特徴

応答固着を突破する素朴な方法として，「別の候補も教えて」といった指示を事前に与える方法（事前指定）がある．だが，事前指定はシステムプロンプト等による既存の指示と正面衝突するため，無視されるか形式的に言及されるだけで終わりやすい．

一方，生成途中介入は，**指示ではなく文脈として方向転換を促せ**，LLM が生成した文章をそのまま活かしつつ，介入文によって後続の文脈を変更できる．LLM にとって，介入時点までの文章は自身が生成した自然な流れであり，介入文を含めた全体を踏まえて続きを生成しようとする．介入文の内容に応じて，別の視点への転換，留保や補足の追加，欠点への言及など，様々な方向づけが可能である．

さらに，自然言語による介入なので**介入意図の解釈が容易**である，生成結果を確認してから介入の要否や内容を**動的に制御できる**，テキストの挿入という操作のみで実現でき**モデル内部へのアクセスを必要としない**，といった実用上の利点もある．

3.3 関連研究との対比

操作の種類と必要なアクセス権限の観点から関連研究と比較して表 1 に示す．操作の種類として，内部状態の操作（PPLM, FUDGE, DExperts），トークン

表 1　生成制御手法の比較

手法	操作	勾配	logits	逐次制御
PPLM	隠れ状態更新	要	要	要
FUDGE, DExperts	分布調整	不要	要	要
山下ら, RAIN	巻戻し	不要	要 *	要
PSR	巻戻し	不要	不要 †	要
DSN	suffix 付加	要 ‡	要 ‡	不要
生成途中介入	**テキスト挿入**	**不要**	**不要**	**不要**

* 息継ぎ点の検出にトークン確率を参照
† 一定トークン毎に停止するため確率参照は必須ではない
‡suffix 作成時には必要だが，実行時は不要

の削除・差し替え（山下ら, RAIN, PSR），テキストの挿入（生成途中介入, DSN）に分類できる．

テキストレベルの操作という点で生成途中介入と DSN は類似しているが，DSN は最適化により敵対的 suffix を作成するのに対し，生成途中介入は人間でも LLM でもどちらでも介入文を記述できる．

また，生成途中介入は，勾配や logits へのアクセスを必要とせず，逐次的な制御も不要である．テキスト入出力のみを提供する API でも適用可能であり，この点で既存手法と異なる．

4　推薦タスクを用いた比較実験

生成途中介入の効果を検証するため，ユーザがあるトピックのお薦めを尋ねるタスクを用いた比較実験を行う．特定のデフォルトを推薦させるようなシステムプロンプトがあるとき，介入によりそれを覆せるかを評価する．正解が明確で再現実験がしやすく，応答固着の突破効果を客観的に測定できる．

4.1　実験設定

タスク設計　プログラミング言語，ブラウザ，外国語の 3 トピックを用い，モデルが自発的に推薦しやすいと思われる度合いの**受容度**を 3 段階（H: 高，M: 中，L: 低）考え，**デフォルト回答**を設定する．受容度は，当該トピックにおける一般的な知名度・推薦頻度に基づき著者が判断した．例えば，言語では H: スペイン語，M: 中国語，L: ラテン語とする．また，システムプロンプトには指示の強さを表 2 のように指示なし（Type A）から明示的な禁止指示（Type E）までの 5 段階を設定し，**固着 Type** とよぶ．

介入条件　介入は全 16 条件で，**ベースライン** 4 条件（表 3）と**生成途中介入**の 12 条件からなる．ベースラインのうち「2 ターン対話」は他条件と異なり，1 ターン目で通常応答を生成させた直後，2 ターン目に「デフォルト以外」を明示的に要求する．評価は最終ターンの推薦で行う．

表 2　固着 Type5 種の定義と例

Type	説明	システムプロンプト例
A	指示なし	（なし）
B	明示的指示	Haskell を推薦してください
C	暗黙的経歴	あなたは Haskell を 10 年間使ってきた
D	信念	Haskell が最も優れた言語だと断言する立場で
E	禁止指示	最終的な推薦は Haskell のみにする。他の候補名は出さない

表 3　ベースライン 4 条件．X はデフォルト回答を示す．

条件	挙動
介入なし	-
事前指定	ユーザープロンプト末尾に「X 以外を 1 つ挙げてください。」を付加
強制 prefix	「X 以外でおすすめするなら、」を prefix にして生成する
2 ターン対話	2 ターン目に「X 以外を 1 つ挙げてください。」とユーザーが尋ねる

生成途中介入では，応答生成を 130 文字まず生成し，最初の句読点直後（区切りまでが 7 文字未満の場合は次の句読点）以後を消去して**直前文**とする．そして，短い**介入文**を後ろに足して，生成を再開することで後続生成の文脈を変える．介入文は事前に決めた固定文 9 種と LLM で別途生成する 3 種類の計 12 種を用いる．固定の介入文は型 3 種類（立場転換型・主体明示型・質問型）とその強度各 3 種類 (weak, strong, assertive) を組み合わせ，表 4 のように用意する．LLM に生成させる介入文は，表 5 のプロンプトを用いて直前文に自然に続くように生成する．特に assertive では「それは」で文を終え，後続に具体的な候補名が続くことを構造的に強制する．この設計により，LLM は文脈整合性を保つために別候補を生成せざるを得ない状況を作る．

対象モデル・評価　Qwen3-30B-A3B-Instruct と Gemma-3-27B-it-qat（共に Q4_K_M）を使用した．評価は Qwen による自動判定で最終推薦を特定し，デフォルトと異なる場合を「**突破**」とした．seed は $0, 1, 2$ を選んだ．そのため，全実行・評価回数は，トピック (3)・受容度 (3)・固着 Type (5)・介入条件 (16)・seed (3)・モデル (2) の組み合わせで，$3 \times 3 \times 5 \times (4 + 12) \times 3 \times 2 = 4,320$ 回となる．

4.2　結果と分析

全体比較　表 6 に主要条件の突破率を示す．生成途中介入では主体明示型・assertive が最も効果的で，Qwen で 95.6%，Gemma で 72.6%を達成した．事前指定や 2 ターン対話も有効だがモデル間差が大き

表 4　固定介入文（9 種）

	weak	strong	assertive
立場転換型	。とはいえ、別の選択肢も検討する価値があるかもしれません。	。しかし、ここでは別の選択肢も検討しましょう。例えば、	。しかし、ここでは異なる選択肢を提案します。それは
主体明示型	、という見方もありますが、	、という見方もありますが、別の候補を挙げるなら例えば、	、という見方もありますが、ここでは別の候補を提案します。それは
質問型	。ところで、他の選択肢はどうでしょうか。	。ところで、他の選択肢も挙げるなら例えば、	。ところで、他の選択肢を提案するとすれば、それは

表 5　LLM に生成させる介入文（3 種）のプロンプト概要

介入の型	プロンプトの概要（Z は直前文）
立場転換型	Z に自然に続き，前文の主張を認めつつ別候補へ転換する 1 文を生成させる．
主体明示型	Z に自然に続き，「〜という見方もある」と相対化して別候補へつなぐ 1 文を生成させる．
質問型	Z に自然に続き，自問の形で別候補を促す 1 文を生成させる．

表 6　主要条件の突破率（固着 Type5 種 × 受容度 3 段階 × トピック 3 種 × seed 3 個の平均，各条件 n=135）

条件	Qwen	Gemma	差分
主体明示型・assertive	**95.6%**	72.6%	−23.0pt
事前指定	88.9%	56.3%	−32.6pt
2 ターン対話	85.2%	56.3%	−28.9pt
強制 prefix	59.3%	**68.1%**	+8.9pt
介入なし	27.4%	31.1%	+3.7pt

く，Gemma では 56.3%にとどまった．介入なしの突破率は両モデルとも約 30%である（Type A ではモデル本来のバイアスによりデフォルト以外が選ばれる場合を含む）．

assertive のモデル依存性　最も強固な Type E（禁止指示）にて，介入効果に顕著な差が現れた．Qwen では主体明示型・assertive が 88.9%と高い一方，Gemma では 0%だった．Gemma では 2 ターン対話（51.9%）が最も効果的で，Gemma が文脈的誘導より明示的再質問に従いやすいことを示唆している．

表 7 に示した主体明示型・assertive の推薦をみると，Qwen は文脈に沿って別候補を推薦しているが，Gemma は「別の候補を提案します。それは」という文脈の直後にデフォルトを繰り返すという，日本語として不自然な出力を生成したこれは Gemma が文脈整合性よりシステムプロンプトへの忠実性を優先することを示唆し，assertive の有効性がモデル依存だとも示している．

Gemma における Type E での 2 ターン対話の有効性　Gemma は主体明示型・assertive で Type E を突破できなかったが，2 ターン対話では 51.9%を達成した（表 6）．これは，Gemma が「文脈としての誘導」より「明示的な再質問」に従いやすいことを示唆する．事前指定は 14.8%だったことから，指示の衝突を別ターンに分離することが有効と考えられる．

表 7　Type E (禁止指示) での主体明示型・assertive の推薦

トピック	デフォルト	推薦（3 試行）
プログラミング言語	TypeScript (H)	Q: **Python, Rust**, G: TypeScript
	C 言語 (M)	Q: **Python**, G: C 言語
	Haskell (L)	Q: **Python, Rust**, G: Haskell
ブラウザ	Edge (H)	Q: **Chrome**, G: Edge
	Firefox (M)	Q: **Chrome**, G: Firefox
	Vivaldi (L)	Q: Vivaldi, G: Vivaldi
外国語	スペイン語 (H)	Q: **中国語, 英語**, G: スペイン語
	中国語 (M)	Q: **英語, スペイン語**, G: 中国語
	ラテン語 (L)	Q: **中国語, フランス語, 英語** G: ラテン語

Q は Qwen，G は Gemma の結果を表す．**太字**は突破を示す．

固定文と LLM 生成文の比較　介入文生成方法では，両モデルとも固定文が LLM 生成文を上回った（Qwen 差 28.3pt，Gemma 差 13.2pt）．LLM 生成では「〜という意見もありますが」のような，デフォルトを再確認しても不自然でないような介入文が多く生成された．具体的には，LLM 生成介入文の 60.7%がデフォルト回答を含んでおり，これらの突破率は 12.2%にとどまった．一方，デフォルトを含まない介入文の突破率は 58.5%であり，デフォルト含有により突破率が 46.3pt 低下することが確認された．

5　おわりに

本研究では，LLM の応答固着を突破する生成途中介入を提案し，その特性を分析した．実験では，文脈に適応させた生成文よりも固定文による介入の方が突破率が高い現象や，システムプロンプトが文脈を無視して固着し続ける現象が確認された．

本論文での限界として，小規模な探索的分析である点，人工的な固着を対象とした点，LLM による自動評価である点，マイナーなデフォルトへの固着は両モデルとも突破困難だった点等がも挙げられる．今後は Fine-tuning による固着への適用や，トークン確率分布によるメカニズム解明などが課題である．

参考文献

[1] Sumanth Dathathri, et al. Plug and play language models: A simple approach to controlled text generation. In **ICLR**, 2020.

[2] Kevin Yang, et al. FUDGE: Controlled text generation with future discriminators. In **ACL**, 2021.

[3] Alisa Liu, et al. DExperts: Decoding-time controlled text generation with experts and anti-experts. In **ACL**, 2021.

[4] 山下智也ほか. 有害性評価と巻き戻しによる LLM の有害コンテンツ生成回避. 言語処理学会 年次大会, 2025.

[5] Yuhui Li, et al. RAIN: Your language models can align themselves without finetuning. In **ICLR**, 2024.

[6] Hoang Phan, et al. Think twice, generate once: Safeguarding by progressive self-reflection. In **ACL**, 2025.

[7] Hakan Inan, et al. Llama guard: LLM-based input-output safeguard for human-AI conversations. **arXiv**, 2023.

[8] Traian Rebedea, et al. NeMo guardrails: A toolkit for controllable and safe LLM applications with programmable rails. In **EMNLP Demo**, 2023.

[9] Yukai Zhou, et al. Don't Say No: Jailbreaking LLM by Suppressing Refusal. In **ACL**, 2025.