人工知能学会 インタラクティブ
情報アクセスと可視化マイニング研究会(第36回)
SIG-AM-36-01

タスク指示と発話系列の表現に注目した LLM の「チャット対話分離」プロンプト最適化

Prompt Optimization for LLM-Based Dialogue Disentanglement Focusing on Task Instructions and Utterance-Sequence Representations

高田尚輝 [1]　　森辰則 [1]
Naoki Takada[1]　Tatsunori Mori[1]

[1] 横浜国立大学大学院環境情報学府
[1] Graduate School of Environment and Information Sciences, Yokohama National University

Abstract: Dialogue disentanglement separates intertwined utterances in multi-user online chats into coherent dialogue, enabling reliable downstream processing. While large language models (LLMs) are promising for this task, prompt-based approaches can be unstable across models and open-weight LLMs still lag behind non-LLM methods. This paper proposes an automatic prompt optimization for LLM-based dialogue disentanglement. We decompose a prompt into three components—task instructions, utterance-sequence representation, and output-format constraints—and optimize them using GEPA, an optimization method for compound AI systems. Experiments on benchmark datasets show that the optimized prompts improve dialogue disentanglement accuracy over the original prompts and can surpass carefully hand-crafted prompts.

1　はじめに

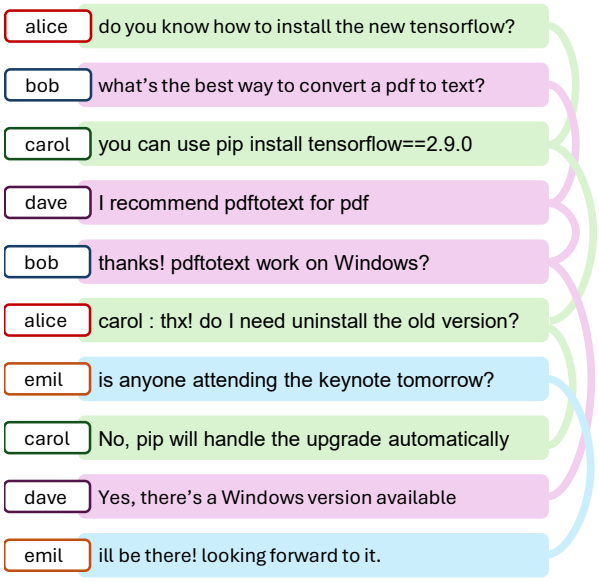

図 1　分離後の対話を色ごとに示す

多数の参加者によって構成される Slack や Discord 等のオンラインチャットにおいて，複数の対話が同時に進行し交錯する現象が生じる．この交錯はユーザがチャット中の文脈を追跡する難易度を上げる．加えて，機械的な解析においても無関係な発話がノイズとして混入し，処理の精度低下を招く．この課題に対し，対話分離 (Dialogue Disentanglement) が提案された [2]．対話分離は図 1 に示すように，対話が交錯している発話系列を応答関係で結びついた一貫性のある対話クラスタへ分割するタスクである．本技術は，対話状態追跡 [3] や応答生成 [4] といった下流タスクの重要な基盤となる．従来，対話分離は教師あり機械学習による発話間ペアワイズ分類として扱われてきた [2]．チャットの各発話について，新しい対話の始まりであるか，もしくはそれ以前のどの発話と応答関係があるのか判定することで対話構造を明らかにする．

大規模言語モデル (LLM) は高度な文脈理解力と推論力を有しており，対話分離への応用が期待される．その初の試みは既存の非 LLM 手法より著しく劣る結果であった [8]．しかし，プロンプトを工夫することでその精度を向上させることができると報告されている．対話単位の判定，後続文脈の付与を導入することで既存手法を凌駕

する精度を達成し，対話分離への LLM 応用の可能性が示された [20]．一方で，対話単位の判定と後続文脈の付与は LLM によって有効性が異なり，精度を低下させる場合もある．そのため，LLM による対話分離の精度を向上させる汎用的手法は未だ存在しない．さらに，LLM が 30B 程度のモデルになると，既存の非 LLM ベースの手法よりも大きく性能が劣ってしまい，対話分離への LLM 応用には改善の余地が残されている．

対話分離は人間にとっても難易度が高い．人手のアノテーションでは多くの揺れが生じ，複数人によるアノテーション結果のすり合わせが必要とされている [5]．そのため，人手で詳細なタスク指示を考えることは困難である．加えて，LLM はプロンプト中での情報の提示形式や出力形式の指示によって性能が大きく変動する [17, 18]．対話分離においてもその傾向は顕著であり，発話系列の表現や出力形式の指示によって精度が大きく変わってしまう (後述する第 5 章，特に表 1 を参照)．そのため，形式の差も考慮したプロンプト設計が重要となる．以上のような背景から，最適な対話分離プロンプトの設計と，その検証を人手で行うことは難しい．そこで本研究では，対話分離における自動プロンプト最適化手法を提案する．プロンプトを次の 3 つの要素に分け，それぞれを最適化する．

タスク指示: 対話分離の定義を説明する．どのようなタスクであるのか，何を根拠に判定を行うべきか説明する．

発話系列表現: 発話番号，発話時間，発話者，テキストメッセージの組を与える．形式変換 (例:JSON で表現) やラベル付けなどを行い，LLM にとって理解しやすい形式で発話系列を提示する．

出力形式指示: 対話分離の判定結果を，LLM が判定に成功しやすい形式で出力する方法を指示する．

本研究の最適化対象は複数要素から成るプロンプトである．そこで，複数プロンプトの自動最適化において最先端の成果を上げている GEPA[15] を利用した手法を提案する．本研究の貢献は設計の難易度が高い対話分離プロンプトを自動最適化する手法を提案したことにある．最適化前のプロンプトを大幅に上回る対話分離精度を達成し，Takada らの研究 [20] で人手の試行錯誤によって作られたプロンプトの結果を上回る精度を達成した．

2 関連研究

2.1 対話分離

対話分離は，長らく 2 段階の処理プロセスとして定式化されてきた [2, 6, 7]．第 1 段階では，指定範囲内の全発話ペアに対して応答関係の度合いを示すスコアを付与する．第 2 段階では特定された関係を統合し，全体的な対話構造を構築する．初期の機械学習モデルは人手の特徴量設計に依存しており，語彙の重複，時間差，ユーザ言及などが利用された [1, 2]．その後，大規模なアノテーション付きコーパス [5] の登場により，エンドツーエンドのニューラルモデル開発が促進された．さらに，BERT 等の事前学習済み言語モデルの導入により，文脈情報の活用が進み精度が向上した [6]．しかし，これらの手法は会話を単なる発話系列として扱う傾向があり，対話構造の考慮が不十分であった．この課題に対し，話者の特性やユーザへの言及といった対話特有の特徴を組み込むアプローチが登場した [7]．DiHRL[8] では，階層的学習損失や easy-first デコードアルゴリズムの統合，大域的な会話特性を捉えることでさらに精度を向上させている．

LLM は自然言語処理において多くの成果を上げているが，対話分離への適用は十分に探求されていない．初の試み [8] は，既存の非 LLM ベースの手法に大きく劣った．この実験で使われたタスク指示は説明性に富むものではなく，「応答関係のある発話を特定し，その発話インデックスを出力せよ．近い発話同士は応答関係を持ちやすい．」のみであり，発話系列表現と出力形式指示においても，簡単な用例を試すに留まっていた．これを受け，Takada らの研究 [20] では，発話系列表現と出力形式指示の改善によって，既存手法を大きく上回る LLM ベースの手法を提案した．発話系列を JSON 形式で表現し，過去の判定から明らかになった対話構造を示す手法 (DLA)，判定対象の発話以降に続く発話を参考情報として挿入する手法 (SC) が提案されている．しかし，一部のクローズモデルでは高精度である一方，30B 程度の LLM では精度が著しく低下してしまう．パラメータの少ないオープンモデルでも高精度な分離ができることは，個人情報を含むチャットデータを扱う上で有用である．本研究はその実現に向け，LLM の対話分離精度を向上させる問題に取り組む．

精度向上を目指すにあたり，より説明性に富んだタスク指示の作成が考えられる．判定の基準が記述されており，どのような発話であっても正しく，一意に判定することができることが望ましい．タスク指示の原案として，人手のアノテーション時に使われた指示書を参考にすることが考えられる．しかし，これまでに多くの対話分離アノテーションが行われた [5, 19] 一方で，詳細に定義された指示書は設計されていない．唯一参照可能であるものも極めて簡単な内容に限る [5]．この指示書は複数のアノテータによる意見統合が前提に作られており，一意にアノテーション可能な高品質なものではない．アノテーション実験では判断の一致度を示すカッパ係数が 0.75 未満であり，判断に揺れが生じていた．対話分離アノテーションでは，複数の応答先が考えられる場合や多様な話

題展開，アノテーション時の個人差があり，高品質な指示書を作成することは難しい．同様に，LLM に与える対話分離のタスク指示作成は困難であり，発話系列表現や出力形式指示まで含めた最適プロンプトの設計は困難を極める．そこで本研究では，自動プロンプト最適化を利用し，最適プロンプト設計を目指す．

2.2 自動プロンプト最適化

LLM の性能を引き出すためには適切なプロンプト設計が求められる．しかし，手動設計は多大な試行錯誤を要し，最適プロンプトの設計は難しい．この問題を解決するべく，自動プロンプト最適化の研究が進められている．初期の研究として，Prompt Tuning[10] や Prefix-Tuning[11] が提案された．これらは自然言語の指示文を探索する代わりに，入力に付与する学習可能な連続ベクトルを最適化する手法である．LLM 本体の重みを固定し，追加した少数のベクトルのみを勾配法で更新する．ファインチューニングよりもコストを抑えられる一方，得られるプロンプトがベクトルであり，人間が内容を読んで解釈することはできない．さらに，ベクトルの次元や注入位置が LLM に依存するため，他の LLM への移植が難しいという課題があった．

これに対し，APE[12] や OPRO[13] といった自然言語プロンプトを最適化する手法は，LLM 自身を最適化器としてプロンプトの探索を行う．これらは最適プロンプトの可読性と汎用性を向上させた．EvoPrompt[14] のように，進化計算的手法を導入することで膨大な探索空間を考慮した最適化手法も提案されている．しかし，その多くは単一タスクの最適化に留まり，複数のエージェントや推論ステップが相互作用する複合システムを最適化する能力には限界があった．近年では，複合システムの最適化に向けて，MIPROv2[16] のようなベイズ最適化やブートストラップ探索を応用した手法も提案されている．本研究では，上記をさらに発展させた，複合システム全体の挙動に対する内省とパレート最適化を導入した GEPA[15] を応用する．GEPA を選んだ理由は 3 つある．第 1 に，複合システムを最適化対象としている点である．第 2 に，自然言語プロンプトを最適化対象としており，利用する LLM が変更された場合でも一定の性能を維持できる点である．第 3 に，先述した 2 点を満たす最先端の精度を誇る手法である点である．しかしながら，GEPA のシステムをそのまま利用することはできない．GEPA の最適化対象はプロンプト中のタスク指示のみであり，発話系列表現や出力形式指示まで扱うことができないからである．よって本研究では GEPA を応用し，発話系列表現と出力形式指示も含めた最適化手法を提案する．

3 LLM による対話分離

3.1 対話分離の定義

チャットデータ中の全発話系列を会話 C と呼び，C から分離された対話の集合を D とする．ここで，各 $d_j \in D$ は，特定の話題や目的を共有し，応答関係で結ばれた発話からなる対話クラスタである．これに基づき，タスクは次のように定式化される．入力は時系列順に並んだ n 個の発話からなる会話 $C = (u_1, u_2, \ldots, u_n)$ である．各発話 $u_i \in C$ は $u_i = (t_i, s_i, m_i)$ で表される．ここで t_i は発話時刻，s_i は発話者，m_i はテキストメッセージである．目的は，C を互いに素な対話クラスタ集合 $D = \{d_1, d_2, \ldots, d_p\}$ に分割することである．この分割は次の 2 つを満たす，C の厳密な分割でなければならない．第 1 は網羅性であり，$C = \bigcup_{d_j \in D} d_j$ が成立することである．第 2 は排他性であり，$\forall j \neq k, d_j \cap d_k = \emptyset$ が成立することである．

3.2 LLM による判定・分離手法

まず，判定対象となる発話 u_{target}(以降対象発話と呼ぶ) を 1 件選び，それより前の発話と共に LLM に与え，新しい対話の始まりであるか，もしくはどの発話と応答関係があるのか，判定させる．ただし，応答先はただ 1 つの発話であることが前提である．C の全発話を判定し，次に，得られた応答関係をもとに先頭の発話から対話クラスタを作る．u_{target} が新しい対話であった場合，u_{target} ただ 1 つを要素とする対話クラスタを作る．u_{target} に応答関係があった場合は，応答先の発話が所属する対話クラスタへ u_{target} を加える．全発話で上記の処理を行い，終了すると，対話クラスタ集合 D を得ることができる．この方法は従来の非 LLM ベースの手法 [2] で用いられており，本研究では判定器を LLM に置き換えて使用する．

上記の方法以外に，u_{target} 以前の発話系列を過去の判定から明らかになった対話構造で表現し，応答先の候補を対話単位で設定する手法 (DLA) や，補足情報として後続文脈を付与する手法 (SC) が提案されている [20]．これらは LLM ごとに精度向上の寄与が異なり，精度が低下する場合もあった．本研究では対話分離プロンプトの自動最適化手法を提案することを目標とし，前段落で述べたように，DLA も SC も導入しない手法を採用する．

3.3 判定のスコア化

GEPA による最適化を行うためには LLM の判定を 1 件ごとにスコア化する必要がある．そのため，判定の成

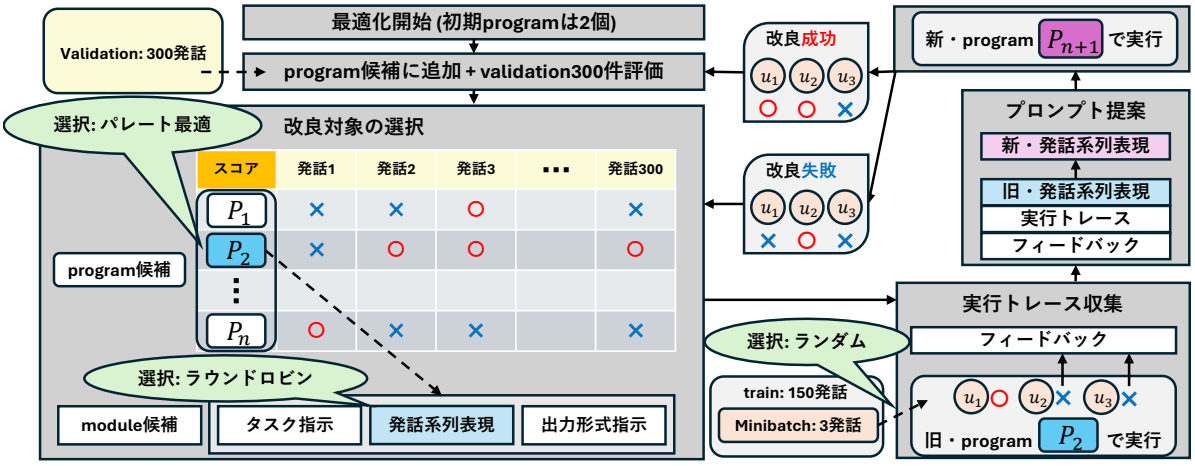

図 2　GEPA による対話分離プロンプト最適化 (改善対象として program 2 の発話系列表現 module が選ばれた例．プロンプト提案する module は発話系列表現に限らず，候補からラウンドロビンで選択される．)

否を 2 値で評価し，成功を 1，失敗を 0 としてスコア化した．判定の成否は次のように定義される．まず，正解データ上で u_{target} が属する対話クラスタを d_{gold} と表す．LLM による判定結果は u_{target} が (i)「新しい対話の始まりである」または，(ii)「ある先行発話と応答関係を持つ」のいずれかである．各判定の成功条件は，(i)「u_{target} が d_{gold} の先頭の発話である」(ii)「d_{gold} 内に，特定された先行発話が含まれる」である．ただし，(ii) の場合は，特定された先行発話が u_{target} よりも前の発話を指定する必要がある．上記の条件を満たさない結果は失敗とみなす．

4　自動プロンプト最適化

4.1　GEPA

1 章で述べたように，対話分離プロンプトを，タスク指示，発話系列表現，出力形式指示の 3 要素に分けて最適化する．この各要素を module と呼び，3 つの module の組を使用して対話分離を行うシステムを program と呼ぶこととする．GEPA[15] を応用して対話分離プロンプトを最適化する手法を図 2 に示す．最適化の過程では，まず，初期 program 群を候補プールに追加し，それらを 300 件の validation データで評価してスコアを算出する．次に，パレート最適な program が改善候補として選ばれる．その過程は 3 段階に行われ，まず各 program において，300 件の validation データにおける各タスクの成否を確認する．次に，ある program の成功したタスクが他の program によって全て成功され，総スコアでも劣っていた場合，その program を候補から除く．最後に，残った候補から総スコアに比例した確率で program が選択される．program が選ばれると，次は，どの module を改善するかをラウンドロビンによって決める．改善対象が決まったら，train データからランダムに抽出した minibatch 3 件を使用し，旧・program で対話分離の判定を行い，実行トレースを収集する．実行トレースには LLM の入出力，推論過程が含まれる．判定に失敗した場合は LLM に失敗理由を生成させ，フィードバックを収集する．フィードバックの生成方法は GEPA で定められておらず，利用者に委ねられている．本研究では，失敗した判定ごとに入出力と真の解を与え，失敗理由を生成させた．minibatch 上でのトレース収集が完了したら，改善を試みる．対象 module の旧・プロンプトと実行トレース，フィードバックが含まれた改善指示を LLM に与え，新・プロンプトを得る．旧，新・プロンプトは対話分離プロンプト全文ではなく，選択された module の要素のみである．新・プロンプトを反映した新・program を minibatch 上で評価し，旧・program の精度を超えた場合に改善成功である．新・program を program 候補に加え，300 件の validation データで評価する．改善失敗の場合は新・program を棄却し，再び改善対象を選びなおす．このサイクルを決められた回数だけ繰り返し，スコアの高い program が最適 program となる．使用したプロンプトやデータセットの詳細は github[*1] にて公開する．

GEPA[15] の改善指示は「実行トレースを注意深く見て新しい指示を作りなさい」である．本研究はこの改善指示を修正し，さらに，module に外部機能を追加することで対話分離プロンプトの最適化を行った．次節ではその詳細を述べる．

[*1] https://github.com/haniwara/sigam2026

4.2 タスク指示

元の改善指示に加え,「出力形式の指示を作ってはいけない」という条件を加えた．この条件が無い場合，得られるプロンプトに出力形式の指示が含まれ，module ごとに分離して改善にすることができないためである．外部機能の追加は無い．最適化過程では出力されたタスク指示の改善案を加工することなくプロンプトに挿入して評価する．

4.3 発話系列表現

新しい発話系列表現の案を得ることを目的とし，元の改善指示を「対話分離を行う LLM にとって理解しやすい発話系列の表現方法を提案せよ」に差し替えた．得られた表現案を任意の発話系列で使うため，発話の形式変換コードを作成する外部機能を設けた．コードは LLM で作る．表現の案を LLM に与え，事前に用意した形式から表現案の形式に変換するコードの作成を依頼する．新・program の評価では，発話系列を形式変換コードで変換し，プロンプトに挿入する．どのような発話系列も提案された表現での挿入を目指す．

4.4 出力形式指示

改善指示を「対話分離を行う LLM にとって判定が正確になる出力形式指示を提案せよ」に差し替えた．提案された出力形式に対応するため，出力解釈コードを作成する外部機能を設けた．コードは LLM で作る．新しい出力形式指示を事前に用意した形式へ変換するコードを作成するよう依頼する．新・program の評価では，新形式で出力される判定を得られたコードで変換し，評価する．

5 評価実験

5.1 実験目的

提案手法を 2 つの観点から評価する．第 1 に，LLM の対話分離精度を向上させることができるか評価する．人手設計のプロンプトと最適プロンプトの精度を比較する．第 2 に，小さなモデルがプロンプト最適化によって高性能なモデルにどこまで迫ることができるか評価する．

5.2 データセット

本研究では，対話分離の標準的なベンチマークである Ubuntu IRC データセット [5] を使用した．本データは，Ubuntu OS に関する技術的な課題解決を目的とした実際のチャットログであり，複数の参加者による交錯した対話が含まれている．データセットは train，dev，test の 3 つに分割されている．このうち dev および test は，複数のアノテータによる検証を経た高品質な正解データであり．dev は 2500 件，test は 5000 件の発話を含む．プロンプト最適化には dev を使い，改善前後での性能差の検証，および，先行研究との比較では test を使用した．

5.3 比較手法

プロンプトが異なる四つの手法について評価する．1 つ目は最適化の初期 program の 1 つで Seed 1 と呼ぶ．タスク指示は DiHRL[8] で試験的に作られた簡単なものである，発話系列表現は，発話の各要素を空白で区切るのみである．出力形式指示は，判定結果を発話番号のみで扱い，新しい対話の始まりである場合は自身の番号，応答先がある場合は応答先の番号の出力を指示する．2 つ目はもう 1 つの初期 program で Seed 2 と呼ぶ．Seed 1 から出力形式指示のみ変更し，新しい対話の始まりであるかを true/false で表すキーと，応答先の発話番号を示すキーの，2 キーでの出力を指示する．3 つ目は Takada らの研究 [20] で Baseline と呼ばれるプロンプトの再現であり，同じく Baseline と呼ぶ．タスク指示は Seed 1 と同じで，発話系列表現は各要素のラベル付き JSON 形式，出力形式指示は Seed2 と同じである．4 つ目は最適化された program を使用する方法であり，Optimum と呼ぶ．Optimum は Seed 1 をタスク指示，発話系列表現，出力形式指示の順に 1 回ずつ改善して得られた。

Baseline は Seed 1, 2 と比べて判定の精度が高い (表 1 参照) ため，2 キーでの出力と JSON 形式の発話表現は精度が向上する要因と考えられる．しかし，Seed 1 のみで始めた最適化では JSON 形式の発話表現が早期に提案される一方，2 キー出力が提案されることは稀であった．初期の改善を円滑に進めるため，Seed 2 を用意した．

5.4 評価指標

対話分離の研究で広く使われている評価指標を採用し，3 つの観点から評価する．第 1 に，クラスタリングの全体的な整合性を評価する，Variation of Information (VI)[9]，Adjusted Rand Index (ARI)[22]，Normalized Mutual Information(NMI)[21]，One-to-One accuracy(1-1)[2]，および Shen-F1 (S-F)[23]．第 2 に，発話 3 件ごとの局所的なクラスタリング精度を測る Local_3[2]．第 3 に，対話クラスタの完全一致した数を評価する，適合率 (P)，再現率 (R)，および F1 スコアを用いた [5]．

表 1　使用 LLM とプロンプトの違いによる成功率の変化

LLM	手法	成功率 (%)
Qwen3-30B	Seed1	80.28
Qwen3-30B	Seed2	81.60
Qwen3-30B	Baseline	91.92
GPT5.2	Baseline	96.12

5.5　ハイパーパラメータ

データセット構築 (5.6 節) を除き，使用する LLM は Qwen3-30B-A3B-Thinking-2507 に限る．リクエスト時のパラメータは (i) 応答関係の判定と (ii) プロンプト改善によって異なる．(i) では LLM での対話分離における再現性確保のため，temperature: 0 とした．(ii) では GEPA と同様に改善提案に多様性を持たせるため，temperature: 0.6，top-p: 0.95，top-k: 20 とした．

応答関係の判定には，候補となる先行発話の範囲を 50 件とした．GEPA のバージョンは v0.023 を使用し，学習データの件数は GEPA[15] に従い，train150 件，validation300 件，minibatch の選択 3 件とした．merge は汎用的な有効性が無いため使用しなかった．

5.6　プロンプト最適化用データセット構築

プロンプト最適化の学習データとして 450 件のタスクを集めたデータセットを構築した．IRC データの dev セット 2500 件を使い，LLM が判定可能な発話のうち難易度が高いもので構成されている．構築方法は次の通りである．まず，使用する LLM とプロンプトを変えながら，表 1 のように判定結果を収集する．次に，いずれの手法でも判定に失敗した 63 件の発話を LLM が判定不可能と候補から除外する．IRC データにはアノテーションミスと思われるものや，対応可能な発話範囲 50 件を超えた応答関係を特定しなければならない場合があるためである．最後に，判定成功率の高い手法で失敗した発話は難易度が高いとみなし，難易度の高い発話でデータセットを構築した．なお，train と validation で難易度は均一化した．

6　実験結果

6.1　ベンチマーク評価

IRC データの test セット 5000 発話に対する各手法の精度を表 2 に示す．上段に非 LLM ベースの最先端手法，中段に高性能なクローズモデルによる分離，下段に本研究の結果を示す．Optimum は Seed と比較して大幅な精度向上を達成した．Baseline と比較しても，全ての評価指標において精度が向上した．この結果は，本研究で提案した自動プロンプト最適化手法の有効性を示す．しかしながら，表の上段にある非 LLM ベース手法の DiHRL[8] や，クローズモデルによる分離 [20] には劣る結果となった．

6.2　最適化の限界

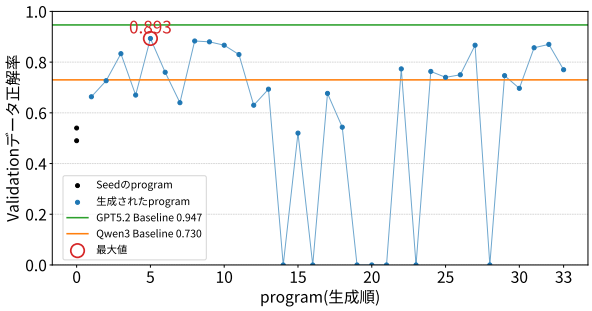

図 3　生成された program の最適化過程

最適化の過程で生成された program とそのスコアについて図 3 に示す．5 個目に生成された program のスコアが最高値となり，それ以降に生成される program では改善が見られなかった．Qwen3-30B の Baseline からの改善は達成したが，GPT5.2 の Baseline スコア以上に改善されることは無かった．最適化回数を増やすことで改善される可能性がある一方，どの program の改善も正解率 0.88 付近に収束し，改善が進行していない．さらに，改善された program は多くの train データで正解してしまい，失敗から内省する事例も少なくなっていた．よって本研究では，最適化の限界に達したとみなし，5 個目の program を最適プロンプトとして結論付けた．

6.3　誤判定の分析

Optimum で IRC test セットを判定した結果のうち，誤判定した事例を定性的に分析した．誤判定は以下の 5 つに分類される．1 つ目は “hi” や “hmm” などの挨拶やリアクションである．同じ人が連続して発話したものだが，間に関係の無い発話が挟まれていると，これらを別の対話であると判定してしまう．2 つ目はサーバーログやシステムメッセージである．応答関係を持たない発話であるが，テキストメッセージに発話者の名前が含まれているため，応答関係があると判定してしまう．3 つ目は “Wine” や “hoary” など，ubuntu OS の専門用語を含む発話である．内容の関連性から応答関係を特定しなけれ

表 2　対話分離精度の比較: 上段に非 LLM ベースの最先端手法，中段にクローズモデルを使った最適化無しプロンプト，下段にオープンモデルを使った最適化前プロンプト (Seed1, Seed2), 人手作成されたプロンプト (Baseline), 最適化プロンプト (Optimum) による結果を示す．各段の最大値を下線，全結果における最大値を太字で表す．

Method		VI	ARI	1-1	NMI	Local_3	S-F	P	R	F1
Elsner [1]		82.10	—	51.40	—	—	—	12.10	21.50	15.50
DiaBERT [6]		93.20	72.80	79.70	—	—	—	42.10	47.90	44.80
Struct [7]		94.60	76.80	84.20	—	—	—	51.80	51.80	51.70
DiHRL [8]		94.23	81.10	84.20	91.85	95.64	87.50	47.97	49.86	48.90
GPT4.1	DLA+SC[20]	95.39	82.22	86.34	96.65	95.14	90.55	46.38	48.73	47.53
Gemini2.5pro	DLA+SC[20]	**97.16**	**92.23**	**90.78**	**97.88**	**97.62**	**92.02**	**58.81**	**63.94**	**61.27**
Qwen3-30B	Seed1	82.97	37.95	56.68	83.83	84.89	63.37	12.03	18.03	14.43
Qwen3-30B	Seed2	83.31	38.98	56.14	84.36	84.68	62.97	11.26	16.34	13.33
Qwen3-30B	Baseline	93.00	70.02	78.46	93.62	94.32	82.81	38.30	40.56	39.40
Qwen3-30B	Optimum	94.12	75.87	82.26	95.51	95.39	84.80	42.22	42.82	42.52

ばならない状況であるが，適切な応答先を見つけることができなかった．4 つ目は抽象的な意見や感想で，応答先が複数考えられる場合である．応答関係は対話ごとに特定の発話者によって構成される．発話者が誰と話しているのか特定し，応答先を判定しなければならない状況で誤判定が起きた．参加者の名前を言及している発話であっても，判定に失敗してしまう場合も存在した．また，対象発話より後ろの発話を参照することができれば，文脈を推測して判定が容易になるような場合も存在した．

7　考察

GEPA[15] では長期の改善プロセスを経て最適解にたどり着く一方，本研究では早期に出た最適解以降に改善が起きなかった．これには次の 2 つが原因として考えられる．第 1 に，多様な事例を考慮したトレースを作成できない点である．話題や参加者の交流，話し口調はチャット内に多様に存在する．その全てに対応できることが理想である一方，改善時には 3 つのトレースしか参考にすることができない．参照するトレースの量が不十分である可能性がある．しかし，トレース 3 件を含んだ改善指示プロンプトは既に 2 万トークンを超えている．単にトレースを増やすだけでは改善指示プロンプトが肥大してしまい，LLM がトレースを十分に考慮することができないと考えられる．トレースを増やす場合は，要約したトレースを挿入するなどしてトークンを減らす必要がある．第 2 に，最適化用データセットの難易度が低い点である．判定難易度の高い発話を特定してデータセットを構築したものの，図 3 ではその半数以上が Seed によって正解されてしまう．高品質な対話分離データセットを大量に用意し，難易度の高い発話を集めることで更なる改善を促す可能性があると考えられる．

8　おわりに

LLM を使った対話分離において自動プロンプト最適化を導入し，その有効性を示した．プロンプトの要素をタスク指示，発話系列表現，出力形式指示に分解し，各要素を最適化した．30B のオープンモデルで最適プロンプトを使った結果は既存の非 LLM ベース手法や最適化の無いクローズモデルの結果には及ばなかったが，その精度を向上させることができた．本研究が，LLM による対話分離プロンプトの最適化に用いられることを期待する．

9　限界と課題

入力可能なトークン数には上限があり，範囲外にある長距離の応答関係は捕捉できない．使用したデータセットはプログラミングの話題に特化しており，他ドメインでの会話内容について有効性は未検証である．また，本チャットデータは公開されており，LLM の事前学習データに含まれている可能性も否定できない．

GEPA のハイパーパラメータ調整，DLA や SC[20] を導入したプロンプトの最適化によってさらに精度向上する可能性がある．

謝辞

本研究の一部は，ＮＥＤＯ（国立研究開発法人新エネルギー・産業技術総合開発機構）の委託事業「経済安全保障重要技術育成プログラム／先進的サイバー防御機能・分析能力強化」（JPNP24003）によるものである．また，本研究の一部は JSPS 科研費 JP24K15084, JP23H00491 の助成を受けたものである．

参考文献

[1] Elsner, M., Charniak, E.: You talking to me? A corpus and algorithm for conversation disentanglement, *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 834–842 (2008)

[2] Elsner, M., Charniak, E.: Disentangling chat, *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 117–126 (2010)

[3] Ouyang, Y., Chen, M., Dai, X., Zhao, Y., Huang, S., Chen, J.: Dialogue state tracking with explicit slot connection modeling, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 34–40 (2020)

[4] Cai, X., Fu, Y., Zhao, H., Jiang, W., Pu, S.: Memory Graph with Message Rehearsal for Multi-Turn Dialogue Generation, *Proceedings of the 31st ACM International Conference on Information & Knowledge Management (CIKM)*, pp. 108–117 (2022)

[5] Kummerfeld, J. K., Gouravajhala, S. R., Peper, J. J., Athreya, V., Gunasekara, C., Ganhotra, J., Patel, S. S., Polymenakos, L. C., Lasecki, W.: A large-scale corpus for conversation disentanglement, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 3846–3856 (2019)

[6] Li, T., Gu, J.-C., Zhu, X., Liu, Q., Ling, Z.-H., Su, Z., Wei, S.: DialBERT: A hierarchical pre-trained model for conversation disentanglement, *arXiv preprint arXiv:2004.03760* (2020)

[7] Ma, X., Zhang, Z., Zhao, H.: Structural characterization for dialogue disentanglement, *arXiv preprint arXiv:2110.08018* (2021)

[8] Li, B., Fei, H., Li, F., Wu, S., Liao, L., Wei, Y., Chua, T.-S., Ji, D.: Revisiting conversation discourse for dialogue disentanglement, *ACM Transactions on Information Systems*, Vol. 43, No. 1, pp. 1–34 (2025)

[9] Meilā, M.: Comparing clusterings by the variation of information, *Learning Theory and Kernel Machines (COLT/Kernel 2003)*, pp. 173–187 (2003)

[10] Lester, B., Al-Rfou, R., Constant, N.: The Power of Scale for Parameter-Efficient Prompt Tuning, *arXiv preprint arXiv:2104.08691* (2021)

[11] Li, X. L., Liang, P.: Prefix-Tuning: Optimizing Continuous Prompts for Generation, *arXiv preprint arXiv:2101.00190* (2021)

[12] Zhou, Y., Muresanu, A. I., Han, Z., Paster, K., Pitis, S., Chan, H., Ba, J.: Large Language Models are Human-Level Prompt Engineers, *Proceedings of the Eleventh International Conference on Learning Representations (ICLR 2023)* (2023)

[13] Yang, C., Wang, X., Lu, Y., Liu, H., Le, Q. V., Zhou, D., Chen, X.: Large Language Models as Optimizers, *Proceedings of the Twelfth International Conference on Learning Representations (ICLR 2024)* (2024)

[14] Guo, Q., Wang, R., Guo, J., Li, B., Song, K., Tan, X., Liu, G., Bian, J., Yang, Y.: Connecting Large Language Models with Evolutionary Algorithms Yields Powerful Prompt Optimizers, *Proceedings of the Twelfth International Conference on Learning Representations (ICLR 2024)* (2024)

[15] Agrawal, L. A., Tan, S., Soylu, D., Ziems, N., Khare, R., Opsahl-Ong, K., Singhvi, A., Shandilya, H., Ryan, M. J., Jiang, M., Potts, C., Sen, K., Dimakis, A., Stoica, I., Klein, D., Zaharia, M., Khattab, O.: GEPA: Reflective Prompt Evolution Can Outperform Reinforcement Learning, *Proceedings of the Fourteenth International Conference on Learning Representations (ICLR 2026)* (2026)

[16] Opsahl-Ong, K., Ryan, M. J., Purtell, J., Broman, D., Potts, C., Zaharia, M., Khattab, O.: Optimizing Instructions and Demonstrations for Multi-Stage Language Model Programs, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 9340–9366 (2024)

[17] Sclar, M., Choi, Y., Tsvetkov, Y., Suhr, A.: Quantifying Language Models' Sensitivity to Spurious Features in Prompt Design or: How I learned to start worrying about prompt formatting, *The Twelfth International Conference on Learning Representations*, (2024)

[18] Tam, Z. R., Wu, C.-K., Tsai, Y.-L., Lin, C.-Y., Lee, H.-Y., Chen, Y.-N.: Let Me Speak Freely? A Study On The Impact Of Format Restrictions On Large Language Model Performance, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pp. 1218–1236 (2024)

[19] Chatterjee, P., Damevski, K., Kraft, N. A., Pollock, L.: Software-related slack chats with disentangled conversations, *Proceedings of the 17th International Conference on Mining Software Repositories*, pp. 588–592 (2020)

[20] Takada, N., Mori, T.: Rethinking Dialogue Disentanglement for LLMs via Dialogue-Level Assignment and Subsequent Context, *Proceedings of the 10th Linguistic and Cognitive Approaches To Dialog Agents Workshop (LaCATODA) co-located with the 40th AAAI Conference on Artificial Intelligence (AAAI)* (2026)

[21] McDaid, A. F., Greene, D., Hurley, N.: Normalized mutual information to evaluate overlapping community finding algorithms, *arXiv preprint arXiv:1110.2515* (2011)

[22] Hubert, L., Arabie, P.: Comparing partitions, *Journal of Classification*, Vol. 2, No. 1, pp. 193–218 (1985)

[23] Shen, D., Yang, Q., Sun, J.-T., Chen, Z.: Thread detection in dynamic text message streams, *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pp. 35–42 (2006)