自己回帰性を組み込んだ直接選好最適化

大井 聖也[1] 鵜飼 麻弘[1] 金子 正弘[2,1] 岡崎 直観[1,3,4] 井上 中順[1]
[1] 東京科学大学 [2]MBZUAI [3] 産業技術総合研究所 [4]NII LLMC
{ohi.m.7b5f@, ukai.m.fe3a@, okazaki@comp., inoue@comp.}isct.ac.jp
masahiro.kaneko@mbzuai.ac.ae

概要

直接選好最適化（DPO）は、人間の選好を用いて大規模言語モデル（LLM）を改善する学習手法として広く用いられている。LLM は自己回帰的に応答文を生成するが、DPO は応答文全体を対象とする Bradley-Terry モデルを仮定した上で報酬を計算しているため、方策モデルの生成過程と報酬モデルの評価過程が整合していない。本研究では、応答を接頭辞ごとに評価する報酬モデルを DPO に導入することで、この不整合を理論的に解消し、自己回帰性を組み込んだ直接選好最適化手法を提案する。複数のモデルを用いた実験の結果、提案手法は DPO を含む既存手法を上回る性能を示した。

1 はじめに

大規模言語モデル（LLM）を改善するための学習方針として、人間の選好データを用いた強化学習（RLHF）が注目を集めている [1, 2, 3]。RLHF を実現するための多くの手法では、応答文を評価する報酬モデル（通常は LLM）と方策モデル（学習対象の LLM）を同時に訓練する必要があるが、直接選好最適化（DPO）[4] では、報酬モデルを解析的に導出し学習対象から除外することで、効率的で安定した勾配法による学習を可能にする。現在では、DPO は数学や対話など幅広いタスクで用いられている [5, 6]。

DPO やその改善手法 [7, 8, 9] では、応答文全体に対して選好を付与する Bradley-Terry (BT) モデル [10] を導入し、人間の選好が BT モデルに従うことを仮定する。そのため、DPO における報酬モデル $r(x, y)$ は、入力 x に対する応答文全体 y を評価するように定義される。しかし、方策モデルである LLM は自己回帰的に接頭辞 $y_{\le i}$（応答文 y の先頭から i トークン目までの部分列）を逐次生成するため、応答文の生成過程と評価過程が整合していない。

従来の RLHF 手法でこの不整合を解決するには、

DPO Loss
$$-\log\sigma\left(\beta\sum_i \log\frac{\pi_\theta(y_i^w|y_{<i}^w,x)}{\pi_{\text{ref}}(y_i^w|y_{<i}^w,x)} - \beta\sum_i \log\frac{\pi_\theta(y_i^l|y_{<i}^l,x)}{\pi_{\text{ref}}(y_i^l|y_{<i}^l,x)}\right)$$

接頭辞ごとの報酬モデル $r^*(x, y_{\le i})$ を導入し、損失を再構成

ADPO Loss
$$-\sum_i \log\sigma\left(\beta\log\frac{\pi_\theta(y_i^w|y_{<i}^w,x)}{\pi_{\text{ref}}(y_i^w|y_{<i}^w,x)} - \beta\log\frac{\pi_\theta(y_i^l|y_{<i}^l,x)}{\pi_{\text{ref}}(y_i^l|y_{<i}^l,x)}\right)$$

図 1: DPO と ADPO の損失。接頭辞ごとの報酬モデル $r^*(x, y_{\le i})$ の導入により、ADPO が導かれる。

接頭辞ごとに応答文を評価する報酬モデルを明示的に学習する必要があるが、そのための学習データの構築は困難であった。例えば、各位置 i における望ましい応答 $y_{\le i}^w$ と望ましくない応答 $y_{\le i}^l$ をラベル付けすることは、時間や費用の観点から現実的ではない。一方で、DPO では報酬モデルを明示的に学習する必要がなく、追加の学習データを構築する必要はない。ゆえに、DPO の暗黙的な報酬モデルを自己回帰的な方策モデルに整合させることは有望である。

本研究では、自己回帰性を組み込んだ直接選好最適化手法である **Autoregressive DPO (ADPO)** を提案する。応答文の接頭辞 $y_{\le i}$ ごとに応答を評価する報酬モデル $r^*(x, y_{\le i})$ を導入し、DPO の損失を再構成することで、ADPO が導出される。具体的には、(1) 報酬モデル $r(x, y)$ によって構成したエネルギー関数とそれらのボルツマン分布によって DPO を再定式化する。次に、(2) DPO では自己回帰的な方策モデルによって非自己回帰的なボルツマン分布を再パラメータ化していることを示し、報酬モデル $r(x, y)$ を $r^*(x, y_{\le i})$ で置き換えることで、再パラメータ化における不整合を解消する。最後に、(3) $r^*(x, y_{\le i})$ を用いて DPO の損失を再構成することで、DPO の理論的基盤を踏襲しつつ ADPO の損失関数を導く。図 1 に示すように、ADPO の損失関数は、DPO の損失関数において対数シグモイド関数を総和の内側に移動させた形式となる。実験により、ADPO が既存手法よりも高い性能を達成することを確認した。

2 DPO の再定式化

本節では、2 つのエネルギー関数を定義し、それらから導かれるボルツマン分布によって DPO の損失関数を再定式化する。x を LLM への入力文、y を応答文とすると、報酬モデル $r(x, y)$ と参照モデル $\pi_{\text{ref}}(y \mid x)$ に対して、以下の 2 つのエネルギー関数を定義する。

$$E_1(x, y) = -r(x, y) \tag{1}$$

$$E_2(x, y) = -\frac{1}{\beta} r(x, y) - \log \pi_{\text{ref}}(y \mid x) \tag{2}$$

ここで、$\beta > 0$ はハイパーパラメータである。

定義した 2 つのエネルギー関数から、それぞれ以下のボルツマン分布が導かれる。

$$p_1(y^w \succ y^l \mid x) = \frac{\exp(-E_1(x, y^w))}{\sum_{y \in Y = \{y^w, y^l\}} \exp(-E_1(x, y))} \tag{3}$$

$$p_2(y \mid x) = \frac{\exp(-E_2(x, y))}{\sum_{y \in \mathcal{Y}} \exp(-E_2(x, y))} \tag{4}$$

p_1 は $Y = \{y^w, y^l\}$ 上で正規化された BT モデルに基づく分布であり、入力 x に対して望ましい応答文 y^w が望ましくない応答文 y^l より選好される確率を表す。p_2 は応答文全体の集合 $\mathcal{Y}$ 上で正規化された分布であり、KL 制約付き報酬最大化問題の最適解であることが示せる [1)]。

DPO は、観測された選好関係に対する尤度を最大化することを目的とし、分布 p_1 の負の対数尤度を損失関数として最小化する。選好データセットを $\mathcal{D}$ として、式 (1) – (4) を用いて整理すると、DPO の損失関数は次式で定義される。

$$\mathcal{L}_{\text{DPO}} = -\mathbb{E}_{(x,Y)\sim\mathcal{D}} \left[\log \sigma \left(\beta \log \frac{p_2(y^w \mid x)}{\pi_{\text{ref}}(y^w \mid x)} - \beta \log \frac{p_2(y^l \mid x)}{\pi_{\text{ref}}(y^l \mid x)} \right) \right] \tag{5}$$

ここで、σ はシグモイド関数を表す。

式 (5) を最小化する p_2 を解析的に求めることは困難であるため、DPO では p_2 をパラメータ θ を持つ方策モデルで置き換える再パラメータ化 $p_2(y \mid x) = \pi_\theta(y \mid x)$ を行う。LLM は自己回帰的に応答を生成するため、確率分布 $\pi(y \mid x)$ は応答 y のトークン長 T を用いて $\pi(y \mid x) = \prod_{i=1}^{T} \pi(y_i \mid y_{<i}, x)$ と分解できる。以上より、最終的な DPO の損失関数は次のように表される。

$$\mathcal{L}_{\text{DPO}} = -\mathbb{E}_{(x,Y)\sim\mathcal{D}} \left[\log \sigma \left(\sum_{i=1}^{T} \left(\beta \log \frac{\pi_\theta(y_i^w \mid y_{<i}^w, x)}{\pi_{\text{ref}}(y_i^w \mid y_{<i}^w, x)} - \beta \log \frac{\pi_\theta(y_i^l \mid y_{<i}^l, x)}{\pi_{\text{ref}}(y_i^l \mid y_{<i}^l, x)} \right) \right) \right] \tag{6}$$

3 Autoregressive DPO

接頭辞に対する報酬モデルの導入　DPO の再定式化では、方策モデル π_θ によってボルツマン分布 p_2 を再パラメータ化していた。ところが、π_θ は自己回帰的な確率モデルである一方で、p_2 は応答文全体 y を一度に扱う分布として定義されており、自己回帰性を明示的には反映していない。この不整合に対処するために、本研究では応答文 $y = (y_1, \ldots, y_T)$ の接頭辞 $y_{\le i} = (y_1, \ldots, y_i)$ に対する報酬モデル $r^*(x, y_{\le i})$ を導入することで、p_2 に自己回帰構造を組み込む。

再パラメータ化における不整合の解消　導入した報酬モデル $r^*(x, y_{\le i})$ に対応して、DPO におけるエネルギー関数を拡張し、接頭辞ごとのエネルギーとして以下のように定義する。

$$E_1^*(x, y_{\le i}) = -r^*(x, y_{\le i}) \tag{7}$$

$$E_2^*(x, y_{\le i}) = -\frac{1}{\beta} r^*(x, y_{\le i}) - \log \pi_{\text{ref}}(y_i \mid y_{<i}, x) \tag{8}$$

接頭辞ごとの選好データ $Y_i = \{y_{\le i}^w, y_{\le i}^l\}$ に対して式 (3) のように BT モデルを適用し、その積として応答全体に対する選好確率 p_1 を以下のように表す。

$$p_1(y^w \succ y^l \mid x) = \prod_{i=1}^{T} \frac{\exp(-E_1^*(x, y_{\le i}^w))}{\sum_{y_{\le i} \in Y_i} \exp(-E_1^*(x, y_{\le i}))} \tag{9}$$

次に、E_2^* に基づいて、各トークン i における条件付き分布 $p_2(y_i \mid y_{<i})$ を定義する。語彙集合 V 上で正規化すると、以下のように書ける。

$$p_2(y_i \mid y_{<i}, x) = \frac{\exp(-E_2^*(x, y_{\le i}))}{\sum_{v \in V} \exp(-E_2^*(x, (y_{<i}, v)))} \tag{10}$$

ここで、$(y_{<i}, v)$ は接頭辞 $y_{<i}$ の末尾にトークン v を連結した列を表す。式 (8) を式 (10) に代入すると、p_2 が連鎖律を満たすことが確認できる。したがって、応答全体に対する p_2 は以下のように表せる。

$$p_2(y \mid x) = \prod_{i=1}^{T} p_2(y_i \mid y_{<i}, x) \tag{11}$$

このように、応答の接頭辞に対する報酬モデル $r^*(x, y_{\le i})$ を導入することで、p_2 に自己回帰構造を明示的に組み込み、再パラメータ化における不整合が解消できる。

1)　詳細を付録 A に示す。

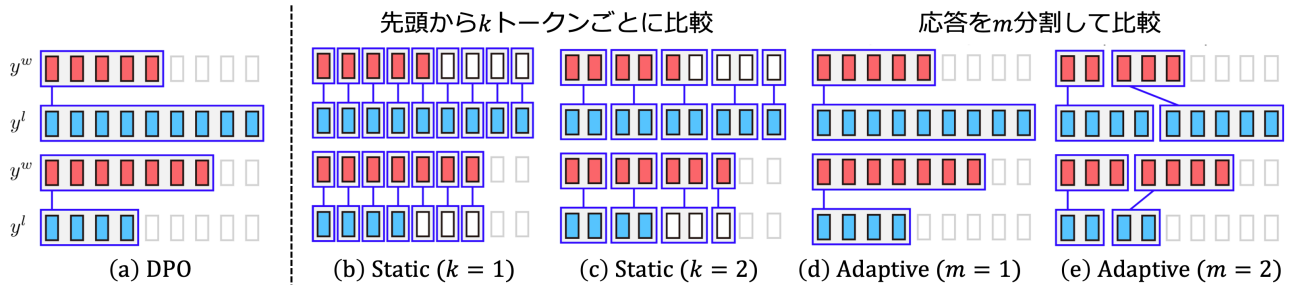

図 2: DPO と ADPO の損失の解釈と、ADPO における応答の分割手法。

損失関数の導出　ADPO では、DPO と同様に p_1 の対数尤度を最大化することで学習を行う。p_2 を学習モデル π_θ により再パラメータ化し、自己回帰的に分解すると、ADPO の損失関数は次の形に導かれる。

$$\mathcal{L}_{\text{ADPO}} = -\mathbb{E}_{(x,Y)\sim\mathcal{D}}\left[\sum_{i=1}^{T}\log\sigma\left(\beta\log\frac{\pi_\theta(y_i^w \mid y_{<i}^w, x)}{\pi_{\text{ref}}(y_i^w \mid y_{<i}^w, x)} - \beta\log\frac{\pi_\theta(y_i^l \mid y_{<i}^l, x)}{\pi_{\text{ref}}(y_i^l \mid y_{<i}^l, x)}\right)\right] \quad (12)$$

DPO の損失関数（式 (6)）と比較すると、ADPO の損失関数では対数シグモイド関数が総和の内側に位置している。図 2 (a, b) に示すように、DPO では応答全体同士を一度に比較する一方で、ADPO では応答を細かく比較し集計していると解釈できる。

応答の分割手法　ここまでは、応答文 y が T 個のトークンに分割されるという前提で議論を行ってきた。しかし、ADPO の損失関数（式 (12)）はトークン単位に限らず任意の分割に適用可能である。そこで本研究では、応答の分割手法として、Static 手法と Adaptive 手法を用いる。Static 手法では、図 2 (b, c) に示すように、それぞれの応答を先頭から k トークンごとに分割して比較する。k が小さいほど細かい粒度での比較となり、k が十分大きい場合は DPO と一致する。望ましい応答と望ましくない応答のトークン長が異なる場合、応答の末尾付近では padding トークン（図中の白色の長方形）と応答内のトークン（図中の赤か青色の長方形）が比較されるため、望ましい比較とならない可能性がある。Adaptive 手法では、図 2 (d, e) に示すように、それぞれの応答を m 分割して比較する。m が大きいほど細かい粒度での比較となり、$m = 1$ の場合は DPO と一致する。Adaptive 手法では、望ましい応答と望ましくない応答の分割数が必ず同じになるため、padding トークンと応答内のトークンの比較は起こらない。

4　実験

4.1　実験設定

モデル　実験では、Llama-3-8B-Base (L-3-8B) [11], Gemma-3-12B-PT (G-3-12B) [12], Qwen-3-8B-Base (Q-3-8B) [13], DeepSeek-Math-7B (DS-7B) [14] の 4 つのモデルを学習し、ADPO の性能を検証する。

データセット　学習データには GSM8K (GSM) [15] の学習スプリットを、評価データには GSM8K の評価スプリットと MATH500 (MATH) [16] を用いる。GSM8K は小学校レベル、MATH500 は競技数学レベルの数学問題で構成されるデータセットであり、どちらも回答に複数段階の論理推論が要求される。評価指標には最終回答の正誤に基づく正解率 (%) を用いる。

比較手法　比較手法には DPO と cDPO [9] を用いる。cDPO は DPO の改良手法であり、推論過程において最終解答に寄与すると推定されるトークンに重みを付与し、損失を再重み付けする。

提案手法　DPO に対応する提案手法として ADPO を用いることに加え、cDPO にも自己回帰性を導入し、cADPO として用いることができる。ADPO と cADPO のいずれにおいても、応答の分割粒度は Adaptive 設定 ($m = 256$) を用いる [2)]。

4.2　実験結果

既存手法との性能比較　既存手法との性能比較の結果を表 1 に示す。実験で用いた全てのモデルと評価データにおいて、提案手法である ADPO と cADPO は、それぞれ対応する既存手法である DPO と cDPO よりも高い性能を示した。Qwen-3-8B を cADPO で学習した際に、GSM8K と MATH500 のス

2)　学習データにおける選好データの構築手順や、cADPO の損失関数などの詳細な実験設定を付録 B に示す。

表 1: 数学タスクにおける既存手法との比較結果。

手法	L-3-8B		G-3-12B		Q-3-8B		DS-7B	
	GSM	MATH	GSM	MATH	GSM	MATH	GSM	MATH
DPO	64.4	18.0	77.0	39.8	87.0	53.8	67.8	32.0
ADPO	**68.1**	**21.0**	**78.3**	**41.2**	**88.1**	**55.4**	**70.0**	**33.4**
cDPO	67.9	16.8	77.2	38.6	91.0	56.8	72.9	33.4
cADPO	**68.8**	**20.2**	**78.9**	**40.4**	**91.7**	**57.2**	**73.5**	**35.4**

コアはそれぞれ 91.7 と 57.2 となり、最も高い性能を達成した。

応答の分割手法の影響　ADPO における応答の分割粒度の性能への影響を分析するために、Static 手法と Adaptive 手法それぞれにおいて k と m の探索実験を行った。結果を表 2 に示す。なお、Adaptive 手法の $m = 1$ は DPO と等価であるため、スコアは DPO と一致する。

Static 手法では、$k \leq 4$ の設定において DPO よりも一貫して高い性能を達成した。全体的な傾向として、k が小さくなるほど高い性能を示しており、細かい粒度で応答を比較することが性能改善につながると考えられる。

Adaptive 手法では、m が小さい（$m \leq 16$）設定では DPO（$m = 1$）からの性能改善は確認できなかった一方で、m が大きい（$m \geq 128$）設定では DPO よりも高い性能を達成した。したがって、Static 手法での結果と同様に、細かい粒度で応答を比較することの有効性が示唆される。

Static 手法と Adaptive 手法の結果を比較すると、Adaptive 手法の方が高い性能を示す傾向が確認された。この結果は、padding トークンと応答内のトークンの比較が起こりうるという Static 手法の問題が、Adaptive 手法では起こらないことによるものだと考えられる。

対話タスクでの性能検証　数学以外のタスクにおける ADPO の有効性を確かめるために、Llama-3-8B-Instruct を用いて対話タスクにおける学習を行い、DPO、SimPO [7] と性能を比較した [3]。学習データ・評価ベンチマークの設定はすべて既存研究 [7] を踏襲した。具体的には、UltraFeedback [17] の指示文を用いてモデルの応答文を取得し、PairRM [18] によって望ましい応答文と望ましくない応答文を決定することで学習データを構築した。評価ベンチマークには AlpacaEval 2 [19]、Arena-Hard [20]、MT-Bench [21] を用いた。AlpacaEval 2 では Length-Controlled (LC) および Win Rate (WR) を、Arena-Hard では Win Rate (WR) を、MT-Bench ではスコアを報告する。

3)　数学タスクで用いた cDPO は、学習時に応答の正誤判定が必要なため、対話タスクでは使用できない。

表 2: 応答の分割手法の影響分析。下線は各分割手法（Static/Adaptive）内での最高性能を、太字は両手法を通じての最高性能を表す。

粒度	L-3-8B		G-3-12B		Q-3-8B		DS-7B	
	GSM	MATH	GSM	MATH	GSM	MATH	GSM	MATH
Static（応答の先頭 k トークンごとに比較）								
$k = 8$	65.0	18.2	76.9	40.0	87.8	54.2	68.4	32.0
$k = 4$	66.3	17.4	77.4	40.8	88.5	54.2	68.2	32.0
$k = 2$	66.6	18.6	77.6	**41.2**	88.3	54.2	70.1	33.6
$k = 1$	66.4	18.0	78.1	39.8	88.5	54.4	**70.4**	**34.8**
Adaptive（応答を m 分割して比較）								
$m = 1$	64.4	18.0	77.0	39.8	87.0	53.8	67.8	32.0
$m = 2$	64.3	19.6	76.9	40.2	88.0	53.8	68.6	33.4
$m = 16$	64.5	18.0	77.1	40.8	87.3	52.4	68.8	33.4
$m = 128$	66.7	18.2	77.6	41.0	**89.2**	54.6	69.8	33.6
$m = 256$	**68.1**	**21.0**	**78.3**	**41.2**	88.1	**55.4**	70.0	33.4
$m = 512$	67.3	19.0	77.9	40.0	88.0	54.0	**70.4**	34.4

表 3: 対話タスクにおける既存手法との比較結果。

手法	Llama3-8B-Instruct			
	AlpacaEval 2		Arena-Hard	MT-Bench
	LC	WR	WR	Score
SFT	26.0	25.3	22.3	6.9
DPO	40.3	37.9	32.6	7.0
SimPO	44.7	40.5	33.8	7.0
ADPO	**45.8**	**41.1**	**34.4**	**7.1**

対話タスクにおける性能検証の結果を表 3 に示す。ADPO はすべてのベンチマークにおいて、既存手法を上回る性能を示した。この結果から、ADPO は数学タスクのみならず対話タスクにおいても有効であり、タスク横断的な有効性が示唆される [4]。

5　おわりに

本研究では、自己回帰性を組み込んだ直接選好最適化手法である Autoregressive DPO (ADPO) を提案した。ADPO では、接頭辞ごとに応答を評価する報酬モデルを導入することで、DPO の再パラメータ化における不整合を解消する。実験では、既存の直接選好最適化手法と比較して、ADPO が高い性能を達成することを確認した。本研究が直接選好最適化における理論的整合性の重要性を示し、今後の選好学習手法の発展に貢献することを期待する。

4)　対話タスクにおける応答の分割手法の影響分析を付録 C に記載する。

謝辞

本研究は JSPS 科研費 25K03135 の助成を受けたものです。また、本研究成果は、国立研究開発法人情報通信研究機構（NICT）の委託研究（22501）により得られたものです。本研究は、東京科学大学のスーパーコンピュータ TSUBAME4.0 を利用して実施しました。

参考文献

[1] Paul F. Christiano, Jan Leike, Tom B. Brown, Miljan Martić, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In **Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)**, 2017.

[2] Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Finn, Paul Christiano, Dario Amodei, and Pieter Abbeel. Learning to summarize with human feedback. In **Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)**, 2020.

[3] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In **Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)**, 2022.

[4] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In **Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)**, 2023.

[5] Richard Yuanzhe Pang, Weizhe Yuan, He He, Kyunghyun Cho, Sainbayar Sukhbaatar, and Jason Weston. Iterative reasoning preference optimization. In **Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)**, 2024.

[6] Junyu Shao, Ke Zhang, and Yifan Huang. Earlier tokens contribute more: Learning dpo from a temporal decay perspective. In **Proc. International Conference on Learning Representations (ICLR)**, 2025.

[7] Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. In **Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)**, 2024.

[8] Yongcheng Zeng, Guoqing Liu, Weiyu Ma, Ning Yang, Haifeng Zhang, and Jun Wang. Token-level direct preference optimization. In **Proc. International Conference on Machine Learning (ICML)**, 2024.

[9] Zicheng Lin, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xing Wang, Ruilin Luo, Chufan Shi, Siheng Li, Yujiu Yang, and Zhaopeng Tu. Critical tokens matter: Token-level contrastive estimation enhances llm's reasoning capability. In **Proc. International Conference on Machine Learning (ICML)**, 2025.

[10] Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. **Biometrika**, Vol. 39, No. 3/4, pp. 324–345, 1952.

[11] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The llama 3 herd of models. **arXiv preprint arXiv:2407.21783**, 2024.

[12] Gemma Team. Gemma 3 technical report. **arXiv preprint arXiv:2503.19786**, 2025.

[13] An Yang, Binyuan Hui, Junyang Lin, Jingren Zhou, et al. Qwen3 technical report. **arXiv preprint arXiv:2505.09388**, 2025.

[14] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. **arXiv preprint arXiv:2402.03300**, 2024.

[15] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. **arXiv preprint arXiv:2110.14168**, 2021.

[16] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. In **Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)**, 2021.

[17] Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Bingxiang He, Wei Zhu, Yuan Ni, Guotong Xie, Ruobing Xie, Yankai Lin, Zhiyuan Liu, and Maosong Sun. Ultrafeedback: Boosting language models with scaled ai feedback. In **Proc. International Conference on Machine Learning (ICML)**, 2024.

[18] Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. Llm-blender: Ensembling large language models with pairwise comparison and generative fusion. In **Proc. Annual Meeting of the Association for Computational Linguistics (ACL)**, 2023.

[19] Yann Dubois, Percy Liang, and Tatsunori B. Hashimoto. Length-controlled alpacaeval: A simple debiasing of automatic evaluators. In **Proc. Conference on Language Modeling (COLM)**, 2024.

[20] Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E. Gonzalez, and Ion Stoica. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. In **Proc. International Conference on Machine Learning (ICML)**, 2025.

[21] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. In **Proc. Annual Conference on Neural Information Processing Systems (NeurIPS)**, 2023.

A　数学的な議論の補足

KL 制約付き報酬最大化問題の最適解　多くの RLHF 手法では、以下の KL 制約付き報酬最大化問題を考える。

$$\max_{\pi} \mathbb{E}_{x\sim\mathcal{D},y\sim\pi(\cdot|x)}\left[r(x,y)\right] - \beta D_{\mathrm{KL}}\left[\pi(y\mid x)\|\pi_{\mathrm{ref}}(y\mid x)\right] \tag{13}$$

ここで、$r(x,y)$ は報酬関数、π_{ref} は参照モデル、$\beta>0$ は KL 制約の強さを制御するハイパーパラメータである。DPO を提案した研究 [4] の付録 A.1 より、この報酬最大化問題の最適解は以下のように導出される。

$$\pi^*(y\mid x) = \frac{1}{Z(x)}\pi_{\mathrm{ref}}(y\mid x)\exp\left(\frac{1}{\beta}r(x,y)\right) \tag{14}$$

ここで、$Z(x)=\sum_{y\in\mathcal{Y}}\pi_{\mathrm{ref}}(y\mid x)\exp\left(\frac{1}{\beta}r(x,y)\right)$ は正規化定数である。

本文中の式 (2) で定義したエネルギー関数 E_2 を用いると、式 (14) は以下のように整理できる。

$$\begin{aligned}\pi^*(y\mid x) &= \frac{\pi_{\mathrm{ref}}(y\mid x)\exp\left(\frac{1}{\beta}r(x,y)\right)}{\sum_{y\in\mathcal{Y}}\pi_{\mathrm{ref}}(y\mid x)\exp\left(\frac{1}{\beta}r(x,y)\right)}\\ &= \frac{\exp\left(-E_2(x,y)\right)}{\sum_{y\in\mathcal{Y}}\exp\left(-E_2(x,y)\right)}\end{aligned} \tag{15}$$

これは、本文中の式 (4) で定義したボルツマン分布 p_2 そのものである。

B　実験設定の補足

数学タスクにおける選好データの構築手順　数学タスクでは既存研究 [9] の設定に従って選好データを構築した。具体的には、まず GSM8K の問題文を学習対象の LLM に入力し、その応答文を 64 回サンプリングして、最終的な回答が正解の応答文と不正解の応答文を 1 つずつランダムに抽出し、望ましい応答文と望ましくない応答文として用いた。なお、正解の応答文または不正解の応答文が 1 つも存在しない問題については、学習データから除外した。

cADPO の損失関数　cDPO [9] は、推論過程において最終解答に寄与すると推定されるトークンに重みを付与することで、DPO の損失を再重み付けする手法である。cDPO の損失関数は以下で定義される。

$$\begin{aligned}\mathcal{L}_{\mathrm{cDPO}} = -\mathbb{E}_{(x,Y)\sim\mathcal{D}}\Bigg[\log\sigma\Bigg(\sum_{i=1}^{T}\Bigg(&\beta\log\frac{\pi_\theta(y_i^w\mid y_{<i}^w,x)}{\pi_{\mathrm{ref}}(y_i^w\mid y_{<i}^w,x)}\\ &-\beta\bar{s}_i\log\frac{\pi_\theta(y_i^l\mid y_{<i}^l,x)}{\pi_{\mathrm{ref}}(y_i^l\mid y_{<i}^l,x)}\Bigg)\Bigg)\Bigg]\end{aligned} \tag{16}$$

表 4: 対話タスクにおける応答の分割手法の影響分析結果。

粒度	Length-Controlled (LC)	Win-Rate (WR)
Static		
$k=4$	44.9	40.9
$k=2$	45.1	40.8
$k=1$	**45.8**	40.8
Adaptive		
$m=1$	40.3	37.9
$m=2$	41.0	40.2
$m=16$	44.1	40.2
$m=128$	**45.8**	**41.1**
$m=256$	44.6	40.5
$m=512$	44.7	40.3

ここで、$\bar{s}_i = 1-s_i$ であり、s_i は正解データと不正解データでそれぞれ訓練したモデルの対数尤度の差によって計算されたトークン単位の重要度スコアである。$\bar{s}_i$ が大きいトークンほど、最終解答に寄与する重要なトークンであると推定される。

DPO に対して ADPO を導入したのと同様に、cDPO に自己回帰性を組み込むことで cADPO を定義する。cADPO の損失関数は以下の通りである。

$$\begin{aligned}\mathcal{L}_{\mathrm{cADPO}} = -\mathbb{E}_{(x,Y)\sim\mathcal{D}}\Bigg[\sum_{i=1}^{T}\log\sigma\Bigg(&\beta\log\frac{\pi_\theta(y_i^w\mid y_{<i}^w,x)}{\pi_{\mathrm{ref}}(y_i^w\mid y_{<i}^w,x)}\\ &-\beta\bar{s}_i\log\frac{\pi_\theta(y_i^l\mid y_{<i}^l,x)}{\pi_{\mathrm{ref}}(y_i^l\mid y_{<i}^l,x)}\Bigg)\Bigg]\end{aligned} \tag{17}$$

DPO と ADPO の関係と同様に、cADPO の損失関数は cDPO の損失関数において対数シグモイド関数を総和の内側に移動させた形式となっている。

C　対話タスクにおける応答の分割手法の影響分析

表 4 に、Llama-3-8B-Instruct を用いた対話タスクにおける分割手法の影響分析結果を示す。評価ベンチマークには AlpacaEval 2 を使用した。数学タスクと同様に、細かい粒度で応答を比較するほど高い性能を達成する傾向が確認された。