メモと圧縮を用いた効率的なメモリモジュールを持つ対話システムの構築

谷口伊織 [1]　南泰浩 [1]
[1] 電気通信大学大学院 情報理工学研究科 情報・ネットワーク工学専攻
t2331097@edu.cc.uec.ac.jp　minami.yasuhiro@is.uec.ac.jp

概要

過去情報の整合性を維持しながら応答を生成する「長期一貫性」能力は，人間らしい対話システム構築において重要な要素の一つである．これを可能にする手法として，人間の長期記憶を模したメモリモジュールを用いる手法がある [1]．最も単純な方法は，対話履歴をメモリに保存し，次の応答生成時にその対話履歴をユーザの応答とともにプロンプト入力に加えることだが，入力可能なシーケンス長の制限や V-RAM 消費量の増大など，様々な問題が発生する．本研究では，これらの問題を解決するため，長期一貫性を持ったオープンドメイン対話が可能な Memochat[2] と対話圧縮モジュールを組み合わせて，高効率なメモリモジュールを持った対話システムを提案する．対話履歴を保存する際，トピックごとに対話履歴を分割し，それらの要約と対話，トピック名を三つ組にしたメモを生成する．さらに，単純な対話圧縮モデルを用いて対話内容を短縮してメモリに保存し，後の応答生成の際にそれを利用することで，モデルへの入力トークン数を抑えつつ，過去記憶に忠実な応答生成を可能にする．公開されたデータセット及び [2] で採用されたデータセットを基に訓練用データセットを再構築し，メモ生成，検索，長期一貫性を持った応答能力の向上を図る．また，メモ生成時に分割した対話履歴内の助詞，助動詞を除去するモジュールで対話を圧縮することで，メモリモジュールの記憶効率性を上げる．最後に，対話ターンが長い対話データセットを用いて長期一貫性及びメモリ効率を検証し，その有効性を確認した．

1　はじめに

人間のように自然な応答を行う対話システムの研究は以前から盛んにおこなわれ，大規模言語モデル (LLMs: Large Language Models) の登場により，さらに加速している．より人間らしい応答生成の方法として，対話履歴をプロンプト入力に加え，過去に交わした対話やその中で出現したユーザ，システムの情報を活用することが挙げられる．しかし，現在主流の Transformer モデルを用いた対話システムでは，入力可能なシーケンス長の制限や KV キャッシュによる V-RAM 消費量の増大などの問題で，今までの全ての対話履歴をプロンプト入力として追加するのは現実的ではない．

この問題を解決するアイデアの一つに，人間の長期記憶を模したメモリモジュールを用いる手法がある．対話システムがユーザから得られた情報やシステム自身の経験を蓄積し，現在の対話文脈に即した情報を検索してプロンプト入力に追加することで，効率よく合理的で一貫性のある振る舞いが可能になる [1]．例えば，MemoryBank[3] は対話とその要約をベクトルにした記憶保存，直近の対話文脈のベクトルによる記憶検索，「エビングハウスの忘却曲線」に基づいた記憶更新メカニズムを対話システムに実装した．また，Memochat[2] は対話履歴をトピックごとに分割，要約しメモとして保存，また，それを用いた記憶検索，応答生成を特別なモジュールを作らず LLM を Fine-Tuning することで可能にした．

この手法の課題は記憶の保存方法である．情報を柔軟かつ理解しやすい形で保存できる最も直接的な手法は，自然言語で直接メモリに記述する方法だが，豊富な意味表現をもち情報量が多い分，メモリ必要量も多い．また，自然言語の冗長性も非効率な記憶保存の一因となる．

本研究では，[2] に対話履歴を圧縮するモジュールを組み合わせることで，記憶効率のより高いメモリモジュールを有した対話システムの実装手法を提案する．さらに，先行研究は英語だけの実装なのに対し，性能が未知数である日本語での実装，適用を目指す．

2 提案手法

図 1 に提案するシステムの概要図を示す．3 つの機能があり，1. 記憶メモの生成，2. 記憶メモの検索，3. 記憶メモを参考にした応答生成である．

まず 1. では，ユーザと LLM ベースのシステムが対話し，対話が一定の長さに達したとき，それまでの対話履歴が対話システムに入力され，システムがそれをトピック毎に分割し，トピック名と分割した対話の要約を生成する．その後分割されたサブ対話は圧縮モデルによって圧縮され，トピック，要約と共に保存される．

次に 2. では，ユーザからの応答とトピック名をもとに，関連する任意の数のトピック・要約ペアを LLM が推論し，関連トピック群とする．

最後の 3. は，2. でヒットした関連トピック群と直近の対話履歴，ユーザからの応答を入力とし，システム応答を生成する．

これらのタスクは特別に設計したモジュールを用いるのではなく，対話システムとして使用する LLM を Instruction-tuning してタスク能力を向上させ，LLM が全てのタスクを実行する．

2.1 インストラクションデータセットの構築

先に説明したタスク能力向上のために，インストラクションデータセットを作成した．各タスクの詳細を説明する．メモ作成では，タスク説明と対話履歴が入力として与えられる．タスク内容は，対話履歴から考えられるトピックの生成，それらに基づいた対話分割，分割した対話の要約である．理想的な出力は，トピック名，トピック毎に分割した対話の開始行，終了行，分割対話の要約をまとめた JSON データである．メモ検索は，LLM が現在の対話文脈から関連するトピック．要約ペアを選ぶタスクの説明と対話文脈，これまで記録したメモのトピック・要約ペア群を入力として与える．ただし，ユーザが新しいトピックの対話を開始する可能性を考慮し，トピック・要約ペア群に加えて，"NOTO"(関連トピックなし) という選択肢も追加する．最終的に LLM が推論した関連トピック候補のインデックスが出力される．メモを活用した応答生成は，メモ検索で取得した関連メモと直近の対話履歴，直前のユーザ対話，それらを考慮した応答を生成させるタスク説明を入力として，LLM はタスク説明に則った応答を出力する．

以上の 3 タスクのデータセットを構築するのに，DialogSum[4], ja_conv_wikipedia_orion14B_100K[1)], 日本語日常対話コーパス [5] を使用した．DialogSum は 1 つのトピック名と，それに関した日常対話，人手で書かれた要約を 1 データとし，計 13,000 個のデータセットである．ja_conv_wikipedia_orion14B_100K は日本語版 Wikipedia データセットを使用し，Orion14B-Chat[2)] で生成された 100,423 個のマルチターン対話データセットである．日本語日常対話コーパスは，5 つのトピックの日常対話を扱ったマルチターン対話データセットで，5,261 個ある．メモ作成データセットは日本語日常対話コーパスや DialogSum の異なるトピックのデータを 2 つ組み合わせ，1 つの対話データとし，それぞれを分割対話とみなす．そしてトピック，要約，分割対話の開始行，終了行を JSON 形式でまとめ，教師データとした．ただし，日本語日常対話コーパスは対話の要約とトピック名がないため，1340 個を人手でアノテーションした．メモ検索データセットは，まず DialogSum から LLM が選択すべき正解のトピックと要約のペアと，ダミーのトピックと要約のペアを用意し，そこに"NOTO"を加えて LLM の選択候補となるトピック・要約ペア群を作成する．次に正解とするペアに対応する対話とダミー対話を組み合わせて，クエリ文とし，LLM はそれに関連するトピック．要約ペアのインデックスを推論する．最後に，正解のインデックスを教師データとした．ダミー対話は日本語日常対話コーパス，ja_conv_wikipedia_orion14B_100K から選択した．応答生成データセットは，初めに ja_conv_wikipedia_orion14B_100K のデータを 4 つ組み合わせて長いターンの対話を疑似的に生成する．次に DialogSum から 3 つデータを用意し，それらをメモ検索で得られたメモと見立てる．その中から教師データにする応答作成のために使用するものを一つ選び，その対話内のことが答えとなるような質問と答えを GPT-4o で生成する．最後に生成した質問を直前のユーザ対話とし，生成した答えをメモを活用した応答 (教師データ) とした．

最終的に，(メモ作成):(メモ検索):(メモを活用した応答生成)=1970:1482:1977，計 5429 個のデータセットを作成した．そして，それぞれ 150 個をテストデータ，40 個を検証データとした．

1） https://huggingface.co/datasets/shi3z/ja_conv_wikipedia_orion14B_100K
2） https://github.com/OrionStarAI/Orion?tab=readme-ov-file

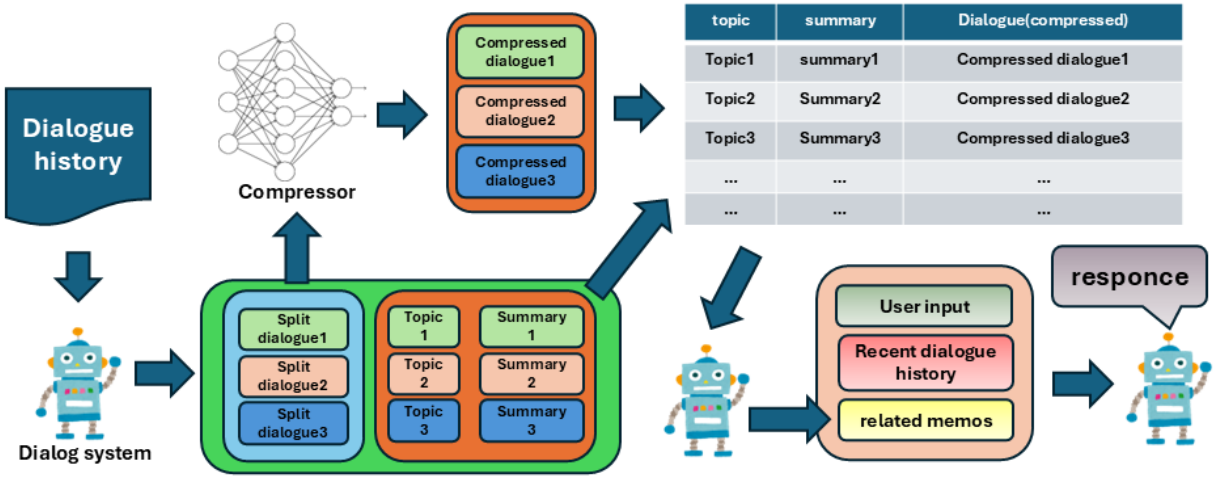

図 1　対話システムの概要図

2.2　対話圧縮モジュールの構築

本研究では二つのモジュールを作成した．

1) At Random. 対話履歴を LLM のトークナイザーでエンコードし，その中のトークンをランダムで削除する．

2) Rm Particle and Auxverb. Mecab[6] を用いて対話履歴を形態素解析し，その中の助詞，助動詞を一定の割合削除する．形態素解析辞書は UniDic を用いる．

3　実験

3.1　実験設定

検証データセット． 2.1 で作成した検証データセット以外に，対話システムの長期一貫性を検証するためのデータセットとして，[2] で用いられた MT-Bench+を日本語訳したものを使用する．

ベースライン． 日本語への適用を目指すため，日本語に特化した LLM である Llama-3.1-Swallow-8B/70B-Instruct-v0.1,Swallow-13b-instruct-v0.1[7, 8] を使用した．Fine-Tuning には LoRA[9] を用い，70B モデルはパラメータを 4bit に量子化して行った [10]．詳細なハイパーパラメータは A.1 に示す．また，実験中の LLM のコンテキストウィンドウを疑似的に 4k に設定して実験した．

評価指標． メモ作成タスクは，トピック，要約生成，対話分割の 3 つを評価した．トピックと要約は precision, recall, F1-Score で評価した．トピックは検証データと完全マッチした時のみ True Positive とし，その要約と検証データの要約を BERTScore[11] で評価した．対話分割は与えられた対話が取りこぼしなく分割できているかを見た．メモ検索タスクは，生成されたトピックオプションと正解のトピックオプションを用いて F1-Score で評価した．メモを活用した応答生成タスクは，生成された応答と検証データの応答との BERTScore を算出した．

長期一貫性. [2] を参考に，MT-Bench+を基に作成された平均 28.6 ターンの対話を GPT-4o に 1̃00 点の間で評価させた．評価するタスクは，対話履歴の用語に関連した質問 (Retrospection)，過去の対話で出てきた知識を基にしたタスク (Continuation)，複数のトピックの内容を組み合わせた質問 (Conjunction) の 3 つである．

圧縮モデルを用いた長距離対話. 圧縮割合を 25%，50%，75%(Rm Particle and Auxverb は 100%も測る) とし，gpt-4o と提案したメモ機構を組み合わせてそれぞれ長期一貫性を調べた．

3.2　メモ関連能力の評価結果

表 1 に Fine-Tuning したモデルの評価結果を示す．

能力の向上が見られたのは，トピック毎の対話分割と要約，メモ検索タスクだった．特に 70B モデルの能力向上が顕著にみられる．一方，トピック生成と応答生成は有意差がみられなかった．特に応答生成は同じ文を何度も繰り返す事象が見られたので，対策として，応答生成のデータセットを増やしたり，各タスクのデータセットの割合を考えたりする必要がある．また，一般的にモデルのパラメータ数に比例して性能が上がるが，8B より 13B のほうが結果が悪いことがしばしば見られた．これは Swallow13B が LLaMA2 ベースのため，基のモデル

表 1　メモ関連能力の評価結果

		Memo Generating							Memo	Chat
		topic			summary			split	Retrieval	with Memo
Setting	Model	Rec.	Pre.	F1	Rec.	Pre.	F1	%	F1	F1
baseline	Swallow-8B	0.066	0.109	0.082	0.519	0.677	0.586	0.275	0.310	0.747
	Swallow-13B	0.100	0.0612	0.0759	0.616	0.508	0.556	0.125	0.135	0.662
	Swallow-70B	0.123	**0.147**	**0.134**	0.527	0.648	0.578	0.800	0.244	**0.794**
LoRA(r=8)	Swallow-8B	0.0898	0.0593	0.0717	0.648	0.530	0.582	0.425	0.517	0.732
	Swallow-13B	0.089	0.058	0.070	0.666	0.528	0.588	0.175	0.147	0.658
	Swallow-70B	**0.129**	0.141	**0.134**	0.772	**0.753**	0.758	**0.975**	0.629	0.747
LoRA(r=32)	Swallow-8B	0.0581	0.0328	0.0420	0.642	0.531	0.579	0.425	0.327	0.732
	Swallow-13B	0.0689	0.0357	0.0470	0.517	0.616	0.560	0.000	0.000	0.668
	Swallow-70B	0.0581	0.0609	0.0595	**0.838**	0.752	**0.792**	**0.975**	**0.635**	0.764
LoRA(r=128)	Swallow-8B	0.0357	0.0243	0.0289	0.707	0.596	0.646	0.275	0.527	0.746
	Swallow-13B	0.0357	0.0128	0.0188	0.695	0.501	0.575	0.100	0.096	0.668
	Swallow-70B	0.0581	0.0588	0.0584	0.771	0.695	0.730	**0.975**	0.604	0.765

の性能差が表れた結果だと考える．

3.3　長期一貫性の評価

表 2　対話システムの長期一貫性の検証結果

Model	Retro.	Cont.	Conj.	Ave.
gpt-4o(4k)	48.05	**76.38**	**69.44**	**64.62**
gpt-3.5-turbo(4k)	43.61	57.22	50.55	50.46
gpt-4o(4k)+memo	**58.05**	73.33	58.61	63.33
gpt-3.5-turbo+memo	51.38	54.44	43.88	49.90
Swallow8B+LoRA(r=8)	26.38	56.66	45.00	42.68
Swallow13B+LoRA(r=8)	25.00	45.00	38.33	36.11
Swallow70B+LoRA(r=8)	25.83	38.61	24.16	29.53

結果を表 2 に示す．メモ機構を導入することで，Retrospection は向上したが，その他のタスク性能は上がらなかった．特に Conjunction タスクのポイントが他と比較して大幅に減っているため，関連メモを複数持ってくるのが難しい，もしくは複数のメモを入力することでモデルのコンテキストウィンドウを圧迫し，うまく応答を生成できなかったことが考えられる．また，メモ関連の能力が最も高かった 70B モデルが最も性能が悪いという結果になった．

3.4　対話圧縮によるメモリモジュールの性能の変化

表 3,4 に圧縮モジュールで対話を圧縮して記憶した場合の性能比較を示す．

表 3　圧縮モジュールによる性能の変化 (At Random)

cmpression ratio / %	Retro.	Cont.	Conj.	Ave.
25	**51.11**	69.72	59.72	60.18
50	56.94	71.11	**68.33**	**65.46**
75	48.33	**73.44**	56.38	59.38

どちらの手法も非圧縮時の結果とあまり変わらないか上がっており，圧縮の有効性が示された．特に Conjunction の性能が上がっており，これはトークンを削除した分タスクに影響する重要なトークンをさらに入力することが可能になったためだと考えられる．また，ランダムでなく，助詞・助動詞だけを削除対象にしたほうが性能が高いことも分かった．

表 4　圧縮モジュールによる性能の変化 (Rm Particle and Auxverb)

cmpression ratio / %	Retro.	Cont.	Conj.	Ave.
25	54.16	**77.50**	**74.44**	**68.70**
50	52.22	70.27	69.44	63.98
75	**55.55**	70.55	61.11	62.40
100	54.16	75.27	65.0	64.81

4　おわりに

本研究では，対話システムの長期一貫性能力向上のため，LLM が対話履歴をメモとして構造化する能力，メモ検索能力，引き出した記憶を活用した応答生成能力を instruct-tuning によって付与し，性能を評価した．また，対話履歴を圧縮することで，メモを効率よく保存し，制限されたシーケンス長でも効率よい応答生成が可能になり，性能が未知数だった日本語への適用の可能性も示せた．今後の展望を述べる．まずメモリモジュールについて，リストに追加する操作でメモを保存するため，対話が長期化するとメモもそれに応じて膨れ上がる．これを抑制するために，類似メモをマージしたり，古いメモを削除したり，保存方法を効率化していく．また，タスク能力をさらに向上させるためデータセットを増強する必要がある．特にトピック生成の能力を向上させるためにトピック生成に特化したデータを増やしていく．次に圧縮モデルについて，単純な操作のみで実現しているので，今後，高性能な LLM から対話圧縮タスクを蒸留したモデルを作成していく．

参考文献

[1] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Jirong Wen. A survey on large language model based autonomous agents. **Frontiers of Computer Science**, 18(6), March 2024.

[2] Junru Lu, Siyu An, Mingbao Lin, Gabriele Pergola, Yulan He, Di Yin, Xing Sun, and Yunsheng Wu. Memochat: Tuning llms to use memos for consistent long-range open-domain conversation, 2023.

[3] Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory, 2023.

[4] Yulong Chen, Yang Liu, Liang Chen, and Yue Zhang. DialogSum: A real-life scenario dialogue summarization dataset. In **Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021**, pages 5062–5074, Online, August 2021. Association for Computational Linguistics.

[5] 赤間 怜奈, 磯部 順子, 鈴木 潤, and 乾 健太郎. 日本語日常対話コーパスの構築. In **言語処理学会 第 29 回年次大会 発表論文集**, pages 108–113, 2023.

[6] Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. Applying conditional random fields to Japanese morphological analysis. In Dekang Lin and Dekai Wu, editors, **Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing**, pages 230–237, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[7] Kazuki Fujii, Taishi Nakamura, Mengsay Loem, Hiroki Iida, Masanari Ohi, Kakeru Hattori, Hirai Shota, Sakae Mizuki, Rio Yokota, and Naoaki Okazaki. Continual pre-training for cross-lingual llm adaptation: Enhancing japanese language capabilities. In **Proceedings of the First Conference on Language Modeling**, COLM, page (to appear), University of Pennsylvania, USA, October 2024.

[8] Naoaki Okazaki, Kakeru Hattori, Hirai Shota, Hiroki Iida, Masanari Ohi, Kazuki Fujii, Taishi Nakamura, Mengsay Loem, Rio Yokota, and Sakae Mizuki. Building a large japanese web corpus for large language models. In **Proceedings of the First Conference on Language Modeling**, COLM, page (to appear), University of Pennsylvania, USA, October 2024.

[9] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. **CoRR**, abs/2106.09685, 2021.

[10] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms, 2023.

[11] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert, 2020.

A Appendix

A.1 訓練の詳細設定

表 5 に Swallow の LoRA-Tuning 時の詳細な訓練設定を示す.

表 5 ハイパーパラメータ設定

	Swallow-8b	Swallow-13B	Swallow-70B
Batch	8	8	1
Accumulation	8	8	16
Epoch	4		
lr	2.0e-5		
max_seq_len	2048		
LoRA-r	8,32,128		
LoRA-α	16		
LoRA-target-module	gate_proj,w_proj,down_proj,v_proj,k_proj,up_proj,o_proj		