Fusion-in-LLM: 質問応答タスクにおける RAG

太刀岡 勇気
デンソーアイティーラボラトリ
tachioka.yuki@core.d-itlab.co.jp

概要

質問応答タスクにおいて、大規模言語モデル (LLM) の知識を補うために検索拡張生成 (RAG) がよく使われるが、Retriever により得られた複数の異なるコンテキストを活用する方法に課題がある。一般的にはコンテキストを逐次的に入力しながら前の答えを書き直すかどうかを LLM 自身に判定させる (逐次 RAG)。一方で fusion-in-decoder と呼ばれる方法では得られたコンテキストをすべて結合してデコーダーに入力し、デコーダーに回答を生成させる。ここでは LLM を並列に用いて質問応答を行う方法 (fusion-in-LLM) を提案し、AI 王を用いたクイズのための質問応答タスクの実験により、逐次 RAG よりも fusion-in-LLM のほうが質問応答タスクに回答する性能が高いことを示す。

1 はじめに

大規模言語モデル (large language model: LLM) に内在にする知識だけでは答えられない質問に回答するためには、検索拡張生成 (retrieval-augmented generation: RAG)[1] が有効である。質問応答タスクに利用する際 [2] には、Retriever により得られた複数の異なるコンテキストを活用する方法に課題がある。例えば、コンテキストには回答の参考になるものとならないものがあるため、回答の参考にならないものにより偽の回答が生成されるという過大がある。一般的な RAG には、逐次的に上位のコンテキストから順番に入力し、前のコンテキストの答えをもとにして新しいコンテキストに基づいて答えを書き直すかどうかを LLM 自身に判定させる逐次 RAG がよく用いられる。しかしながらこの方法を質問応答タスクに適用すると、top-1 のコンテキストのみを入力した場合に比べて、top-2,3 を用いると回答の正解率が大きく低下することが分かった。

一方で fusion-in-decoder (FiD) と呼ばれる方法 [3] では、得られたコンテキストをすべて結合して decoder に入力し、デコーダーに回答を生成させる。ここではこの方法と同様に LLM を並列に用いて質問応答を行う方法 fusion-in-LLM の方法を提案する。逐次的に書き直すのではなく、各コンテキストに対応する回答を一つのコンテキストに基づき生成し、回答を後から多数決を取ったり、LLM 自身に適切な回答を選ばせたりする。AI 王を用いたクイズのための質問応答タスクの実験により、本報告では一般的に使われている逐次 RAG よりも fusion-in-LLM のほうが質問応答タスクに回答する性能が高いことを示す。

2 質問応答タスクのための検索拡張生成 (逐次型)

2.1 Dense passage retrieval

LLM で質問応答タスクを解くために、根拠となるであろう関連パッセージを dense passage retrieval (DPR) を用いて取得する。質問文 q と根拠となる文書群 (ここでは wikipedia)p をそれぞれ、質問エンコーダー E_q および文書エンコーダー E_p により固定次元の埋め込みベクトル $E_q(q)$ と $E_p(p)$ を得る。その後、Faiss を用いて関連パッセージを類似度スコア順に top-n 個抽出する。これをコンテキストとして、LLM に質問文とともに入力する。関連パッセージを取得する手続きはほぼ共通なので、この際に異なるコンテキストが複数得られることから、これらを LLM にどう入力するかが問題となる。

2.2 検索拡張生成 (RAG)

図 1 に示すように得られたコンテキストと質問 (query) をプロンプトとして与える。その際に答えのみを簡潔に回答するように指示する[1)]。

1) この指示はあまり守られず、長い文章が出力されることもあるので、初めの改行記号が出力されるまでを回答として扱った。

あなたはクイズ番組の回答者です。
コンテキスト情報を参考にして、query に 20 文字以内で答えだけを単語で簡潔に回答してください。
Context: ...
Query: ...
Answer:

図 1　RAG を行うためのプロンプト.

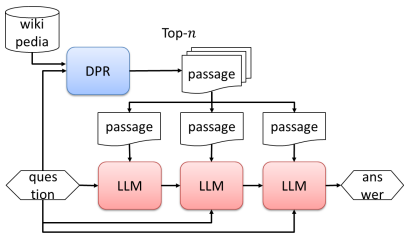

図 2　LLM の直列接続による逐次 RAG.

2.3　RAG のための LLM の低ランク近似 (LoRA) ファインチューニング

コンテキストを有効に活用し、正しい答えを導きだすための方法を LLM にファインチューニングする。そのために、学習データを用いてコンテキストと回答のペアを生成させ、コンテキストは適合していたものの回答が間違っていたものを収集する。これを図 1 の ### Answer のあとに正解を付加して、これを再現するように LoRA により LLM をファインチューニングする。

2.4　逐次 RAG(リファインプロンプト)

図 2 に最もよく使われている逐次 RAG を示す。DPR で得られた複数のコンテキストを扱うために、llama-index[2] 等の汎用フレームワークで標準的に使われている方法で、コンテキストを top-$\{1, 2, \ldots, n\}$ と n 個順番に入力し、前の回答を書き換えるかどうかの判断をプロンプトで LLM にさせる方法である。

これを実現するプロンプトを図 3 に示す。前のコンテキストに基づく回答 (original answer) を新しいコンテキスト (new context) を参考にすることで書き直すかどうかを、LLM に判断させている。よくわからない場合には、original answer を繰り返すように指示している。

3　Fusion-in-LLM

fusion-in-decoder の方法 [3] では、質問と各関連パッセージを連結したものをエンコーダーでベクトル化して、それを n 件連結したものを decoder に入力する。すなわち並列にコンテキストを接続し、decoder により正解を選ばせることを行う。これと同様に LLM に対しても、関連パッセージごとに質問応答させ、のちに回答を選ぶ方法が考えられる。図 4 のように、コンテキストを別々に複数の LLM に入力し、出力された回答の多数決を取って最終的な回答を決定する。

2) https://www.llamaindex.ai/

あなたはクイズ番組の回答者です。
コンテキスト情報を参考にして、query に 20 文字以内で答えだけを単語で簡潔に回答してください。
ほかのコンテキスト情報を参考にした original answer を new context を使用して書き直すことができます。
new context が役に立たない場合やよくわからない場合は、original answer を繰り返してください。
New context: ...
Query: ...
Original answer: ...
Answer:

図 3　答えを書き直す逐次 RAG を行うためのリファインプロンプト.

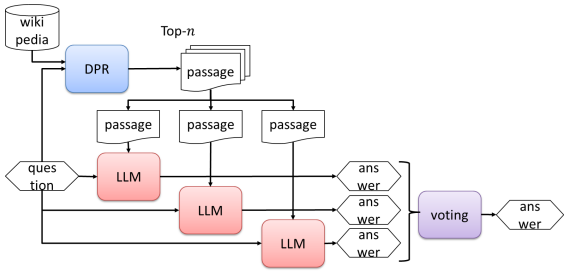

図 4　Fusion-in-LLM.

3.1　質問文との比較による不適切回答の除外

質問応答の前提として、回答が質問文にそのまま含まれているものは明らかに不適切な回答である。例えば質問「映画『ウエスト・サイド物語』に登場する 2 つの少年グループといえば、シャーク団と何団?」に対する回答として「シャーク団」は適切でないことは明らかである。そのため答えが質問文にそのまま含まれている場合は不適切回答として除外する。

あなたはクイズ番組の回答者です。query に対して適切と思う回答を choices の中から一つ選んで回答してください。

```
Query:...
Choices: ...
### Answer:
```

図 5　答えを選択肢から選択させるプロンプト.

3.2　多数決による回答の選択

LLM により top-n までのコンテキストに対応する最大 n 個の回答[3]を多数決する。個数が同じ場合には上位の結果のものを採用する。

3.3　LLM による回答の選択

LLM に得られた回答を選択肢として提示し適切なものを選択させる。これを実現するプロンプトを図 5 に示す。

4　AI 王での実験

4.1　実験の設定

質問応答タスクとして AI のクイズへの回答能力を試す AI 王[4]のタスクを用いた。AI 王は学習 (22335 件)・検証 (1000 件)・評価 (1000 件) の 3 つのセットに分けられている。検証・評価セットは様々なジャンルから集められた質問 1000 件からなる。このうち評価セットの正解は公開されていないため、学習セットでモデルを学習し、検証セットで性能を評価した。

参照テキストは wikipedia を対象として dense passage retrieval (DPR) [3] により、関連パッセージの検索および抽出を行う。モデルの構築は AI 王で配布されている FiD のベースラインシステム[5]と同様にした。

まず前段として top-30 までの関連パッセージを DPR により抽出する。これを後段の RAG のコンテキストとして、LLM に入力する。この部分に関しては著者らが transfer-2 (RAG タスク)[4] のベースラインコードとして提供している[6]。LLM は llama-2 をベースに日本語の様々なタスクでインストラクトチューニングした 70 億パラメータの日本語 LLM[5][7]を用いた。

表 1　質問応答タスクの性能.

w/o RAG	RAG	sequential RAG		LoRA+RAG
	top-1	top-2	top-3	top-1
19.1	34.0	27.6	23.0	40.5

4.2　結果と考察

表 1 に結果を示す。llama-2 をベースにした 70 億パラメータの日本語 LLM に質問文のみを与えた場合の正解率は 19.1%で、LLM に内在する知識だけでは解けず RAG が必須の課題である。これに対して RAG を行い top-1 のコンテキストを与えると 34.0%まで正解率が向上した。

複数コンテキストを利用するために逐次 RAG を行ったところ、top-2、top-3 と増やすほど性能が低下し、34.0%の正解率が 23.0%まで低下した。正解の回答まで誤って修正してしまったためである。少なくともこのタスクにおいては逐次 RAG は有効でないことが分かった。

さらにコンテキストの有効な活用法を LLM に教示するために、まずは「質問文+コンテキスト」→「回答」のペアを学習セットのうちからランダムに選択した 8000 件に対して top-1 のみ生成した。表 2 に示すように、DPR が正解/不正解 (DPR が適合コンテキストを提供しているかいなか) と LLM の正解/不正解で分類した。両者ともに正解のものが 1846 件あった。DPR が適合文書を提供した 2616 件に対しては LLM の正答率は 70%であった。これを引き上げるために残りの 30%に対応する、コンテキストは適合していたものの LLM が不正解となった 774 問を対象として、2.3 節に示す方法で LLM の低ランク近似したうえで finetuning を行った。これにより正解率は 40.5%まで向上した。コンテキストの活用法を教示することは性能向上に有効である。

これに対して、提案の fusion-in-LLM で多数決を取った場合の正解率を、表 3 に示す。top-3 までの回答の多数決を取ったものは 44.5%まで正解率が向上した。複数のコンテキストを質問応答タスクで使う場合には逐次 RAG よりも提案の fusion-in-LLM が有

3)　回答なしや前節での不適切回答の除外があるので、実際には n 個より少なくなることもある。
4)　https://sites.google.com/view/project-aio/home
5)　https://github.com/ntcirtransfer/transfer2/tree/main/RAG/AIO3_FiD_baseline。AI 王の第 4 回は早押しに特化しているため、第 3 回のシステムをベースラインとして採用した。
6)　https://github.com/ntcirtransfer/transfer2/tree/main/RAG/transfer2_llm_baseline
7)　https://huggingface.co/elyza/ELYZA-japanese-Llama-2-7b-fast-instruct

表 2　DPR と LLM の正解/不正解での分類.

		LLM(+RAG)		
		correct	wrong	total
DPR	correct	1842	774	2616
	wrong	488	4896	5384
total		2330	5670	8000

表 3　Fusion-in-LLM による正解率.

top-n	3	5	7	10	20
多数決	44.5	46.5	46.3	45.3	42.3
LLM による選択	44.3	46.1	46.2	45.7	43.4

効であることが示された。

図 6 には多数決による正解率とオラクル (top-n までの回答のうちに正解が一つでも含まれていた割合) の正解率の比較を示す。top-5~7 のときに正解率は 46.3%~46.7%で最高であるが、オラクルの場合と多数決の結果の最良の場合にはいまだに大きな開きがある。

5　まとめと今後の課題

LLM に不足する知識を補うために有効な RAG で複数コンテキストを活用する方法を検討した。方法に課題がある。一般的にはコンテキストを逐次的に入力しながら RAG を行うが、この方法を繰り返すと正解率が大きく低下することが分かった。一方で fusion-in-decoder を参考として、LLM を並列に用いて質問応答を行う fusion-in-LLM を提案した。AI 王を用いたクイズのための質問応答タスクの実験により、逐次 RAG よりも fusion-in-LLM のほうが top-5~7 程度のコンテキストを利用した場合に、質問応答タスクに回答する性能が高いことを示した。ただし、オラクルで結果を選択した場合には 30 位程度の結果であっても性能の向上がみられ、下位のコンテキストが有効な場合も多い。これに関しては回答の選択法を改善することで正解率を改善できる見込みがあるが、それは今後の課題である。

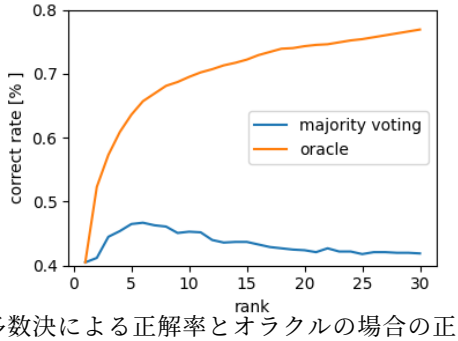

図 6　多数決による正解率とオラクルの場合の正解率の比較.

参考文献

[1] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In **Proceedings of the 34th International Conference on Neural Information Processing Systems**, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc.

[2] Zizhong Wei, Dengrong Huang, Jichen Zhang, Chen Song, Sijia Zhang, Jianing Zhang, Zhaochuan Li, Kai Jiang, Rui Li, and Qiang Duan. GARAG: A general adaptive question-answering system based on RAG. In **Proceedings of the 2024 International Conference on Cloud Computing and Big Data**, ICCBD '24, p. 442–447, New York, NY, USA, 2024. Association for Computing Machinery.

[3] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-Tau Yih. Dense passage retrieval for open-domain question answering. In **Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)**, pp. 6769–6781, Online, November 2020. Association for Computational Linguistics.

[4] Hideo Joho, Atsushi Keyaki, Yuuki Tachioka, and Shuhei Yamamoto. Building test collections for Japanese dense information retrieval technologies and beyond. In **Proceedings of The First Workshop on Evaluation Methodologies, Testbeds and Community for Information Access Research (EMTCIR '24)**, 12 2024.

[5] Akira Sasaki, Masato Hirakawa, Shintaro Horie, and Tomoaki Nakamura. ELYZA-japanese-llama-2-7b. https://huggingface.co/elyza/ELYZA-japanese-Llama-2-7b, 2023.