SQL Agent とマルチモーダル LLM を用いた 文書内の表に対する質問応答システム

青柳 直人　森田 祐介
みずほ第一フィナンシャルテクノロジー株式会社
{naoto-aoyagi, yusuke-morita}@fintec.co.jp

概要

多くの文書は表が含まれているため，質問応答において文書内の表に対する質問に正確に回答することは重要である．本研究では，回答の根拠となる表の検索と表に対する推論を必要とする質問応答タスクに取り組んだ．まず，従来手法の比較実験を行い，SQL Agent の回答性能が良いこと，SQL Agent に表画像を入力に加えた手法 (SQL Agent-V) が表記揺れに対して頑健である可能性が示唆された．表記揺れに対する頑健性を向上させるために，LangGraph を利用して SQL Agent と SQL Agent-V を質問に応じて動的に切り替える手法を提案し，表記揺れを発生させたデータセットで実験を行うことで提案手法の有効性を確認した．

1　はじめに

大規模言語モデル (LLM) の発展に伴い，LLM に学習データに含まれないテキストをプロンプトに代入することで，外部知識を踏まえた質疑応答を行う， Retrieval-Augmented Generation(RAG) [1, 2] に関する研究が進んでいる．RAG において表に対する質問に正確に回答することは重要である．表から抽出した文章は構造化情報を失っており，そのままプロンプトに代入し LLM に入力すると，通常の推論と比較して回答精度が落ちてしまう [3].

近年，言語モデルを用いて表に対する推論を行う手法が提案されている．研究の方向性は大きく二つある．一つは，特殊な埋め込み層や注意機構を言語モデルに組み込み，表のセルや一部を復元することでモデルを事前学習する方法である [4, 5, 6]．もう一つは，事前学習を実施せずに既存の LLM を用いて推論を行う方法である [7, 8, 9] ．Wang ら [9] は，列の追加，行の選択，グループ化，ソート等の操作を段階的にしていくことで、表に対する理解を深めてクエリに回答する手法である Chain-of-Table を提案した．各操作は Python の関数としてあらかじめ実装されており，どのような順番でどの操作を実行するかの選択を LLM で行うことで最終的な回答を得る．ただし，表全体をプロンプトに入れる必要があるため，一定以上大きい表では Chain-of-Table を実行することが困難であることには注意が必要である．このように，既存の LLM を用いた手法の多くは表を文字列として入力するため，表のサイズが大きい場合は使用するトークン数が多くなり，回答精度の悪化や LLM の最大トークン数を超えてしまう可能性がある．

自然言語で表されたクエリを SQL クエリに変換する Text-to-SQL の技術も盛んに研究されている [10, 11, 12]．Li ら [13] は，複数の異なる LLM が出力した SQL クエリの実行結果を多数決に用いて最終回答を得ることで回答精度を向上させた．LLM を使用してアプリケーションを構築するためのオープンソースライブラリである LangChain[1)]では，Text-to-SQL の技術を用いて SQL データベースを操作する SQL クエリを生成し，自ら実行を行う SQL Agent が提供されている．SQL Agent には，実行結果を受けとって，多段階でデータを加工する SQL クエリを生成したり，トレースバックをキャッチして SQL クエリの生成を自律的にやり直したりすることができるという利点がある．SQL Agent は，回答の根拠となる表が予め特定できている場合では，繰り返し SQL を実行することで回答を得ることができる．しかし，表の候補が大量にあり，回答に必要な表を事前に特定する必要があるケースにおいては，SQL Agent 単体では機能しない．

また，テキストと画像データの両方を入力とするマルチモーダル型の LLM を用いることにより，表構造を持つ画像に対し，直接質問回答を生成させ

1） https://www.langchain.com

ることも可能である．Singh ら [14] は，数学，視覚データの分析，コード生成などの構造化推論タスクにおいて，マルチモーダル型の LLM である GPT4-V の評価を行い，コード生成において，表を画像化してテーブルのスキーマとともに入力にすることによって，画像を入力しない場合より精度が向上することを示した．入力として表が与えられている場合に質問応答を行う研究は存在するが，対象となる表を特定する必要がある場合の質問応答タスクにおいて，表の画像も入力に追加したマルチモーダル型の LLM を利用して SQL Agent で回答生成を行った例は，我々の知る限り存在しない．

そこで，本研究では回答の根拠となる表の検索と表に対する推論を必要とする質問応答タスクにおいて，SQL Agent と SQL Agent の入力に表の画像を追加したモデル (SQL Agent-V とする) を組み合わせて回答を生成する手法を提案する．表から文字列を抽出し LLM に入力するのではなく，LLM に質問の根拠となる情報の抽出や集計を行う SQL クエリを記述させ，それを実行してから質疑応答のプロンプトに代入して回答させる方法を採用することで，計算などの複雑な処理を必要とする質問に対する精度が向上することを確認した．また，SQL クエリを記述させる際に，マルチモーダル型の LLM を利用して表の画像を入力に含めることで，質問内の単語と表中の単語で表記揺れが存在する場合において回答精度が向上することを確認した．表画像がノイズとなり不正解になってしまう質問もあるため，通常の SQL Agent と SQL Agent-V を質問に応じて動的に切り替える手法を提案し，実験を行うことでその有効性を確認した．

2 提案手法

本研究では，与えられた質問に対して，回答の根拠が含まれる表を検索により特定し，その後特定した表に対する質疑応答を行うタスクについて実験を行った．

2.1 データセット

実験では，回答の根拠となる表の検索と表に対する複雑な推論を必要とする質疑応答タスクを提供する Open-WikiTable[15] を用いた．このデータセットは，Wikipedia 内の表，表に対する質問，質問に答えるための SQL クエリ，答えから構成される．21,676 の表の中から質問に関連する表を特定し，質問の回答を生成する．質問は Train/Valid/Test に分かれているが，今回は主に Test の 6,602 件の中から，明らかに正解が誤っているサンプルと Azure OpenAI Service のコンテンツ フィルターにかかるサンプルを除外した 6,578 件を使用した．実験の一部では Train，Valid のデータセットも補助的に使用した．

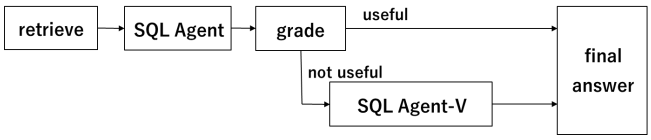

図 1 提案手法のフロー

2.2 Vector Store と画像 DB の作成

まず，表を最大 100 単語ごとに分割した後，各行の間に [SEP] トークンを挟んだ文字列シーケンスをテキスト埋め込みモデルでベクトル化し，テキストと共に chunk として Vector Store に格納しておく．次に，それぞれの chunk に対応する表を pandas Dataframe で読込み，dataframe_image[2] を用いて画像化し，chunk とメタデータで紐づけて DB に保存した．

2.3 回答に必要な表の検索

質問を元に表検索を行う際は，同様に質問文をベクトル化し，質問文の埋め込みベクトルと Vector Store に格納された chunk の埋め込みベクトルのコサイン類似度を計算し，上位 k 個の chunk に紐づく表を質問応答に必要な根拠を含む表として取得する．

2.4 特定した表に対する質疑応答

図 1 に流れを示す．まず，質問に関連する表を埋め込みベクトルのコサイン類似度の大きい上位 k 件の抽出する (retrieve)．類似度が上位の表から順に SQL Agent で SQL クエリを生成して実行することで質問に対する回答を作成する．SQL クエリの実行結果が空の場合は，類似度が次に大きい表で同様の処理を行う．空でない場合は，質問と LLM による回答を LLM に入力して，回答が質問に関連しているか否かを判定する (grade)．回答判定に用いたプロンプトは付録 A.1 に掲載する．プロンプトは汎用的なものではないので，使用する LLM やデータセットに応じてカスタマイズする必要がある．関連していると判定された場合 (useful) は SQL Agent の出力を回答とし，そうでない場合 (not useful) は SQL

2) https://github.com/dexplo/dataframe_image

Agent-V で解く．SQL Agent-V では，検索で特定した chunk とメタデータで紐づけられた表画像を画像 DB から特定し，マルチモーダルモデルにテキストと共に入力する．SQL Agent-V でも，SQL Agent で回答を生成したときと同様に，類似度が上位の表から順に入力して回答を生成する．本手法の実装には，LLM エージェントのステップをグラフ化して状態管理を行うためのツールである LangGraph[3] を使用した．

2.5 使用モデル

LLM は，Azure OpenAI の gpt-4 (0613)，gpt-4o (2024-05-13) を用いた．gpt-4o はマルチモーダル型の LLM で表画像を入力する際に使用し，それ以外の場合は gpt-4 を使用した．また，埋め込みベクトルの計算には Azure OpenAI の text-embedding-ada002 モデルを用いた．

2.6 評価指標

表に対する質問応答では，生成された回答と正解データが完全に一致しているか否かを判定する exact match accuracy (EM) を評価指標にすることが多い [15, 16]．EM は機械的に計算できるが，生成された回答の内容は合っているが，表記が正解データと異なる場合に誤りと判定されてしまう．例えば，正解が”0”で生成された回答が”Zero”の場合，内容は合っているが回答は誤りとされる．LLM による回答では特に表記がデータセット内のものと変わってしまうことが考えられる．今回は，EM による評価に加えて生成した回答と正解の意味合いが同じであるかを LLM を使って判定させる correctness (CRT) を評価指標とする．すべての実験において評価時に使用する LLM は gpt-4 (0613) とした．CRT 算出時に使用するプロンプトは付録 A.2 に記載した．

3 従来手法の比較

通常の RAG，表を html に変換し構造化情報を保持したまま LLM に入力する手法（RAG(html)），RAG に画像を入力に含める手法（RAG+Vision），Chain-of-Table，SQL Agent，SQL Agent-V の 6 手法の精度を比較する．類似度検索では 5 つの表を検索によって抽出し ($k = 5$)，その中に回答の根拠となる表が含まれているか示す retrieval accuracy はいずれの場合も同一で，93.42% となった．表 1 に各手法の評価指標を示す．データセットの中で，平均，合計，最大値，最小値，カウントなどの質問の回答にあたり計算を要する質問を計算問題（calculation），計算を要さず表の中から条件に応じて抽出する質問を非計算問題（non-calculation）として，分けた場合の精度も算出した．

表 1 従来手法の比較結果

	all	non-calculation	calculation
	CRT (EM)	CRT (EM)	CRT (EM)
RAG	63.55 (47.77)	72.15 (54.97)	45.51 (32.66)
RAG (html)	65.17 (51.82)	74.35 (60.21)	45.93 (34.26)
RAG+Vision	66.05 (32.35)	**77.48** (40.58)	42.12 (15.11)
Chain-of-Table	67.54 (52.14)	76.94 (60.27)	47.86 (35.11)
SQL Agent	**73.12** (**61.78**)	75.86 (**64.02**)	**67.39** (**57.08**)
SQL Agent-V	72.03 (59.50)	76.35 (62.97)	62.96 (52.24)

非計算問題では．通常の RAG と比較して，RAG(html) と RAG+Vision の CRT が高い結果となった．このことから，構造化情報を保持して入力すること，画像を入力に追加することの有効性が確認できた．また，計算問題では RAG+Vision の精度が落ちており，複雑な処理では画像が逆にノイズになっている可能性が示唆された．

CRT については，非計算問題では RAG+Vision，全体と計算問題では SQL Agent の精度が最も高い結果となった．特に計算問題では，SQL Agent が Chain-of-Table と比較して約 20 %高い結果となった．このことから，計算を要する質問に対しては，SQL Agent などで表に対する計算をプログラムで実行することが効果的であると考えられる．EM では，全体 (all)，非計算問題 (non-calculation)，計算問題 (calculation) のいずれにおいても SQL Agent が最も高い値となった．ただし今回使用したデータセットの正解は SQL クエリの実行結果で作成されており，他の手法と比較して SQL Agent で作成した回答は正解との表記揺れが発生せず，相対的に有利になっている可能性があることに留意されたい．

SQL Agent もしくは SQL Agent-V それぞれで正解した質問数は図 2）のようになった．SQL Agent で不正解かつ SQL Agent-V で正解の質問 530 件を詳しく見ていくと，質問内と表で単語の表記が異なるようなケースが多く見られた．例えば，質問が”What is the date of the athlete in the CARIFTA Games Women’s Under 20 Shot Put event?”のサンプルについて，表から”Event”カラムが”Shot Put”の日付を抽出する必要があるが，表内の表記が”Shot put”と大文字小文字の表記が質問と異なり，表記揺れが生じている．SQL

3） https://www.langchain.com/langgraph

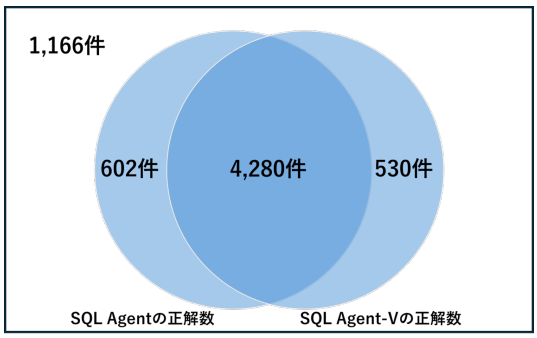

図 2　各手法で正解した質問数

Agent では，条件抽出で使用すべき”Shot put”を質問内での表記のまま”Shot Put”としてしまうため，回答を得ることができない．SQL Agent-V を使用することで，画像情報を使って，表中の記述と質問との間に生じる表記揺れを踏まえた SQL クエリの生成を行うことができる．このことから，SQL Agent に画像を入力に加えることで，表記揺れに対して頑健なモデルになることが示唆される．一方，画像を入力することがノイズとなり不正解になってしまう質問もあるため，SQL Agent と SQL Agent-V を質問に応じて動的に切り替えることができれば，さらなる精度向上が見込まれる．

4　実験

従来手法の比較から，SQL Agent が他のモデルと比較して性能が良いこと，SQL Agent-V にすることで表記揺れに強くなる可能性があることが示唆された．ここでは，表記揺れの起きているケースを対象として，SQL Agent-V の頑健性の検証と SQL Agent と SQL Agent-V を組み合わせた提案手法の有効性の検証を行う．

表記揺れに対する頑健性を調べるために図 3 のように質問内の日付に着目して，質問内と表内の日付の表記を異なるものにすることで意図的に表記揺れを発生させたデータセットを新たに作成した．まず，Train/Valid/Test すべてのデータセットから，質問内に日付を含むことに加えて，正解 SQL クエリの WHERE 句に日付を含むサンプル 3,914 件を抽出した．次に，質問内の日付表記を表内の日付表記と異なるものとする目的で，LLM を用いて質問の日付表記の変換を実施した．変換に使用したプロンプトは付録 A.3 に示す．以上の手続きで作成したデータセットを用いて実験を行うことで，表記揺れがある場合に各手法がどの程度影響を受けるかを検証した．

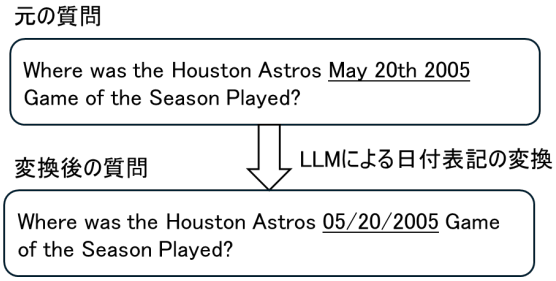

図 3　LLM による日付表記の変換

表 2　実験結果

	date not converted CRT (EM)	date converted CRT (EM)
SQL Agent	76.80 (59.17)	72.43 (55.72)
SQL Agent-V	80.97 (61.80)	78.31 (59.79)
Mixed	**81.55** (**63.06**)	**79.31** (**60.63**)

類似度検索では 5 つの表を検索によって抽出し (k = 5)，retrieval accuracy は 93.71% となった．表 2 に実験結果の評価指標を示す．日付表記を変換したデータセット (date converted) と変換しなかったデータセット (date not converted) の 2 種類で実験を実施した．両データセットについて提案手法 (Mixed) の CRT と EM が最も高い値となり，提案手法の有効性が確認できた．提案手法は，SQL Agent-V では画像がノイズとなる質問に関しても，SQL Agent を採用することで回答精度が向上したと考えられる．同じ手法で日付表記変換の有無を比較すると，SQL Agent の CRT は変換により 4.37% 下がった．これに対して，SQL Agent-V の CRT 下落は 2.66%，提案手法の CRT 下落は 2.34% と比較的抑えられている．このことから，表の画像を入力に加えマルチモーダルに処理したことが表記揺れに対する頑健性の向上に寄与したと考えられる．

5　まとめ

本研究では，回答の根拠となる表の検索と表に対する推論を必要とする質問応答タスクに取り組んだ．従来手法の比較実験の結果，SQL Agent が他モデルと比較して回答精度が高いこと，SQL Agent-V が表記揺れに対して頑健である可能性を示した．質問内の単語と表中の単語で表記揺れが存在する場合において，SQL Agent-V は画像がノイズとなり不正解になってしまう質問もあるため，通常の SQL Agent と SQL Agent-V を質問に応じて動的に切り替える手法を提案した．LLM によって表記揺れを発生させたデータセットを新たに作成して実験を行い，提案手法の有効性を確認した．

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, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. **Advances in Neural Information Processing Systems**, Vol. 33, pp. 9459–9474, 2020.

[2] Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. In-context retrieval-augmented language models. **Transactions of the Association for Computational Linguistics**, Vol. 11, pp. 1316–1331, 2023.

[3] Peng Li, Yeye He, Dror Yashar, Weiwei Cui, Song Ge, Haidong Zhang, Danielle Rifinski Fainman, Dongmei Zhang, and Surajit Chaudhuri. Table-gpt: Table fine-tuned gpt for diverse table tasks. **Proceedings of the ACM on Management of Data**, Vol. 2, No. 3, pp. 1–28, 2024.

[4] Jonathan Herzig, Pawe l Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Martin Eisenschlos. Tapas: Weakly supervised table parsing via pre-training. **arXiv preprint arXiv:2004.02349**, 2020.

[5] Ewa Andrejczuk, Julian Eisenschlos, Francesco Piccinno, Syrine Krichene, and Yasemin Altun. Table-to-text generation and pre-training with tabt5. In **Findings of the Association for Computational Linguistics: EMNLP 2022**, pp. 6758–6766, 2022.

[6] Zihui Gu, Ju Fan, Nan Tang, Preslav Nakov, Xiaoman Zhao, and Xiaoyong Du. Pasta: Table-operations aware fact verification via sentence-table cloze pre-training. In **Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing**, pp. 4971–4983, 2022.

[7] Yunhu Ye, Binyuan Hui, Min Yang, Binhua Li, Fei Huang, and Yongbin Li. Large language models are versatile decomposers: Decompose evidence and questions for table-based reasoning. **arXiv preprint arXiv:2301.13808**, 2023.

[8] Wenhu Chen. Large language models are few(1)-shot table reasoners, 2023.

[9] Zilong Wang, Hao Zhang, Chun-Liang Li, Julian Martin Eisenschlos, Vincent Perot, Zifeng Wang, Lesly Miculicich, Yasuhisa Fujii, Jingbo Shang, Chen-Yu Lee, et al. Chain-of-table: Evolving tables in the reasoning chain for table understanding. **arXiv preprint arXiv:2401.04398**, 2024.

[10] George Katsogiannis-Meimarakis and Georgia Koutrika. A survey on deep learning approaches for text-to-sql. **The VLDB Journal**, Vol. 32, No. 4, pp. 905–936, 2023.

[11] Weixu Zhang, Yifei Wang, Yuanfeng Song, Victor Junqiu Wei, Yuxing Tian, Yiyan Qi, Jonathan H Chan, Raymond Chi-Wing Wong, and Haiqin Yang. Natural language interfaces for tabular data querying and visualization: A survey. **IEEE Transactions on Knowledge and Data Engineering**, 2024.

[12] Bowen Qin, Binyuan Hui, Lihan Wang, Min Yang, Jinyang Li, Binhua Li, Ruiying Geng, Rongyu Cao, Jian Sun, Luo Si, et al. A survey on text-to-sql parsing: Concepts, methods, and future directions. **arXiv preprint arXiv:2208.13629**, 2022.

[13] Zhishuai Li, Xiang Wang, Jingjing Zhao, Sun Yang, Guoqing Du, Xiaoru Hu, Bin Zhang, Yuxiao Ye, Ziyue Li, Rui Zhao, and Hangyu Mao. Pet-sql: A prompt-enhanced two-round refinement of text-to-sql with cross-consistency, 2024.

[14] Mukul Singh, José Cambronero, Sumit Gulwani, Vu Le, and Gust Verbruggen. Assessing gpt4-v on structured reasoning tasks. **arXiv preprint arXiv:2312.11524**, 2023.

[15] Sunjun Kweon, Yeonsu Kwon, Seonhee Cho, Yohan Jo, and Edward Choi. Open-wikitable: Dataset for open domain question answering with complex reasoning over table. **arXiv preprint arXiv:2305.07288**, 2023.

[16] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. **arXiv preprint arXiv:2004.04906**, 2020.

A　付録

A.1　分岐に使用するプロンプト

You are a grader assessing whether an answer addresses a question.
Give a binary score 'yes' or 'no' to indicate whether or not any answers are given.
If the answer is “I don't know anything” or similar, please leave “no”. Otherwise, please answer “yes”.

A.2　correctness 算出時に使用するプロンプト

{input} に質問, {reference} に正解データ, {prediction} に LLM の生成した回答を入力する.

You are an expert professor specialized in grading students' answers to questions.
You are grading the following question:
{input}
Here is the real answer:
{reference}
You are grading the following predicted answer:
{prediction}
Respond with CORRECT or INCORRECT:
Grade:

A.3　日付表記の変換

{question} に元の質問を入力する.

Please print out the sentences in the following question, changing only the date notation.
question：{question}
new question：