F-027

生成 AI による設計書への問合せ対応のための表混在設計書向けパーサー
Development of a Parser for Mixed-Table Design Documents to Facilitate Inquiry Response Using Generative AI design

河内谷耀一‡
Yoichi Kawachiya

1. はじめに

近年，生成 AI（Generative AI）の発展により，業務効率化や知識活用の分野での応用が急速に進展している[1]．世界市場においても，生成 AI の活用は，業務支援や文書処理，顧客対応といった幅広い分野で拡大しており，2030 年には数兆円規模の市場が予測されている[2]．この中でも，設計書や仕様書といった複雑な業務文書を活用するというニーズは高まっている．このような文書を読み取る際に，Chargrid [3] や DocFormer [7]など，文書のレイアウト構造を保持しながら解析を行う技術が登場し，ドキュメント解析の方向の理解精度は向上しつつある．文書読み取りの潮流に関して特に，Retrieval-Augmented Generation（RAG）は，外部知識データベースから関連情報を動的に取得し，生成 AI による回答精度を向上させる技術として注目されている [4]．しかし，設計書のように表や注釈，階層構造を含む複雑な文書では，従来の単純なテキスト変換やナレッジ化手法では情報が正確に反映されず[3，5，6，7]，PdfTable [9] による表構造の抽出や，文書理解に関する包括的なもの[10] を活用しても期待した精度を達成できないという課題がある．

本研究では，設計書特有の情報を正確に解析・構造化し，生成 AI による回答精度向上をめざした「生成 AI による設計書への問合せ対応のための表混在設計書向けパーサー」を提案する．本手法では以下の点を特徴とする：

1. 設計書内の表構造，視覚情報（例: 取り消し線，色），階層構造を精密に解析し，文脈を保持．
2. RAG の検索機構を改良し，チャンク分割や関連度評価を最適化．
3. 生成 AI が活用できる形式で知識データベースを構築．

提案手法を用いた実験では，設計書の内容に基づく直接的な質問には 99%の精度で回答可能であり，複雑な質問に対しても 70%以上の精度で関連情報を提示可能であることを示した．また，資料検索に要する時間を平均 10 分短縮することが可能となり，業務効率化に大きく寄与することが確認された．

さらに，本研究の成果は製品設計に関する問合せ対応システムに限らず，他分野の複雑な文書解析やナレッジ管理にも応用可能である．今後は，対象となる文書形式の拡大によるパーサーの汎用性の向上を進める．

2. 課題

2.1 RAG の概要

Retrieval-Augmented Generation（RAG）は，ユーザーからの入力（質問）に対して，まず関連する外部ドキュメントを検索し，その結果をプロンプトに含めて生成 AI へ入力することで，応答品質を向上させる手法である．代表的な RAG システムは，以下の 3 ステップから成る．(1) 検索フェーズ：質問に関連する文書片を，予め構築した知識データベース（DB）から検索，(2) 拡張フェーズ：検索した文書をプロンプトに組み込む．(3) 生成フェーズ：プロンプト全体を元に生成 AI が回答を出力．この三つのステップを組み合わせたアプローチにより，生成 AI 単体では不可能な「専門知識に基づく正答」が可能となる．一方で，RAG を用いる際の効果は，予め用意する知識 DB の品質，検索アルゴリズムの精度，プロンプト設計に大きく依存する．

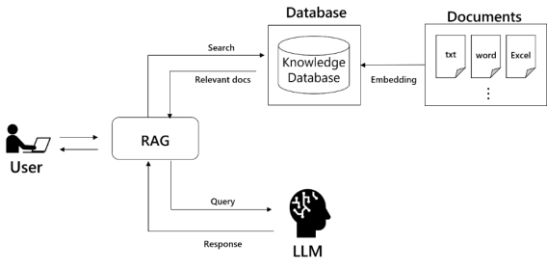

図 1　Retrieval-Augmented Generation 概要

2.2 RAG 向け知識 DB 構築及び前処理の課題

前節の通り，RAG の性能には知識 DB の品質に大きく依存しており，構築にあたっては，次のような課題がある．

1. 粒度設定：文書をどの単位（段落，セクション，表など）で分割するか．
2. メタデータ付与：検索精度向上のための分類情報付加．
3. ノイズ除去：誤った文書構造や無関係情報の排除．

特に設計書のような表中心・構造重視型文書では，単純な OCR や PDF テキスト変換では十分な検索精度が得られない．専門的なパーサー技術が不可欠となる．

2.3 設計書に特有となる構造面での課題

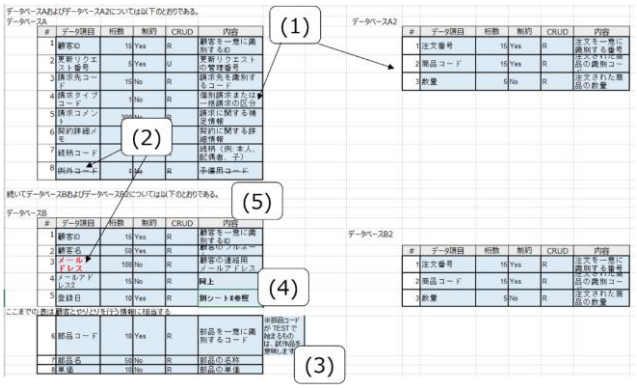

図 2　表混在設計書凡例

今回の研究では Excel 形式で記載されているものであって，表と文章を混在するもの設計書文書を対象とした具体

的には図２に示すような特徴を持つ設計書を取り込む手法を開発した．以下の課題から構成される．

1. 自由に配置された表を区別する必要がある
2. 視覚的情報（色分け，取消線など）を補う必要がある．
3. 表そのものの読み方を前提知識として必要とする
4. 表内で参照対象を識別し，記載する必要がある．
5. 設計書内の文章がどの表に対するものなのかを識別する．

これらに対応できなければ，設計書からの有効情報抽出は困難であり，ひいては生成 AI応答の正確性を損なう結果となる．

2.4 RAG の知識 DB 検索機構における課題

2.1 節にて言及した通り，RAG の効果を大きくするためには，知識 DB への検索精度を向上させる必要がある．この部分に関連して，以下の二つが課題となる．

1. 知識 DB への投入の際の適切なチャンク分割
2. 設計書の検索に即した類似度評価

これらの機構を整えることで，知識 DB から適切に関連する情報を引き出すことができ，問い合わせの精度向上につながる．

3. 手法

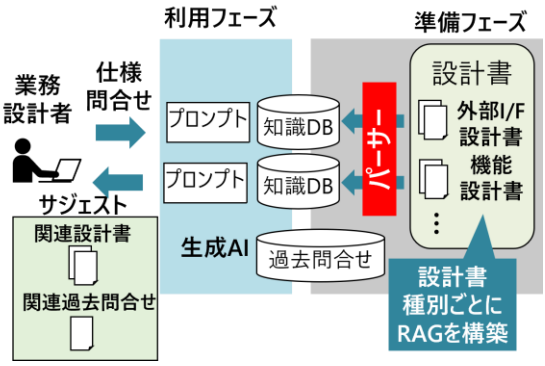

図 3　システム全体構成

3.1 パーサー設計指針

本節では，設計書を正確に読み取り，生成 AIが活用可能な形にして知識 DB へと取り込むために開発したパーサーの設計指針を述べる．2.3 節にて挙げた課題に基づき整理した設計書向けパーサーに求められる要件は以下の通りである．

表 1　パーサーの要件と内容

要件	内容
表区別・識別	表開始位置・終了位置・範囲を正確に認識させること
視覚情報抽出・補足	取り消し線や文字色など視覚的要素を含む情報を補完すること
表の読み方補佐	表周辺に記載された判例や注釈，補足説明を認識，解釈すること
階層構造・参照関係整理	表内の「同上」「別シート参照」といった記述から参照関係を再構成し，明示すること
表と文の対応付け	表の解説と表を適切に抽出し，結び付け，文脈を補完すること

これらを実現するため，要件ごとにパーサーを Python によって実装した．次節にて具体的な処理機構と実装のアプローチをする．

3.2 パーサー機構の詳細

前節で整理した 5 つの要件ごとに，パーサーの内容とその実装について詳述する．

1. 表の区別・識別

各表を構成するセル範囲を明確に定義し，独立した構造として抽出するため，以下の手順に基づいて処理を行う．

- 開始位置の探索
  各シート内のセルを走査し，「#」や「No.」などの表識別子を持つセルを基準点とする．
- 列範囲の決定
  基準点から右方向に非空セルをスキャンし，空白または別識別子の出現までを列範囲とする．
- 行範囲の決定
  確定した列範囲をもとに，下方向に非空セルをスキャンし，全列が空白となる行が出現するまでを表の行範囲とする．

これにより，Excel 上の表を明確に切り出し，再配置および表番号の付与を行う．

2. 視覚由来の情報の取得

openpyxl ライブラリを用いて，以下の視覚的属性を抽出・処理する．

- 取り消し線
  セルごとに取り消し線の有無を検出し，取り消しされた文字列を出力から除外した上で，末尾に「取り消し内容：～」として別記する．
- 色・フォント属性
  通常とは異なるフォントスタイル（太字・斜体）や色（赤文字・ハイライト）を検出し，意味付けが可能なもの（例：赤字＝注意喚起）は明示的に注記する．その他は「（赤字）」「（太字）」などの形式でテキスト末尾に追記する．

3. 表の読み方の説明補佐

- 表外注釈の統合
  表の周辺に記述された注釈・凡例（例：「※～」「注釈：～」「凡例：～」）を抽出し，対応する表に結合する．表の先頭行に追記する形式とした．
- 特殊な表構造の補足
  縦横の見出しの交差が不明瞭な場合や多重ヘッダ構造を持つ表では，構造的意味付けを加えて再構成を行う．

4. 階層構造や参照関係の理解および記載

- 表内の直接参照（同上・←など）
  元の表構造を保持しつつ，参照の整合性を維持．
- 別表・別シート参照
  (1)で付与した表番号およびシート名を用いて，参照

元のセルを「表 X（SheetY）」といった形式で明確化.

- セル内のリンク抽出
  URL やハイパーリンクが含まれる場合はリンク先を抽出し，括弧付きで注記.

5. 表と文章の対応付けの追記
   - 表タイトルの抽出
     表の上部または前段にあるタイトル・キャプション文を検出し，表と結合.
   - 表の言及箇所とのリンク付け
     本文中に「表 X に記載の通り」などの記述がある場合は，対象表と逆リンクを付与することで双方向の文脈把握を可能にした.

3.3 RAG 適用における検索機能改善

RAG 適用における，検索機能の改善方法について，2.4 節にて述べた課題の具体解決方法を詳述する.

まず，知識 DB への投入の際の適切なチャンク分割について述べる．チャンク分割単位が小さすぎると文脈が失われ，逆に大きすぎると冗長性や検索精度の低下を招く．これに対し，本研究では，表が開始される直前や文の区切り（句点）を基準に分割し，文書構造を保ちながら適切な粒度に調整した．さらに，大規模な表などがトークン制限を超える場合には，制限に従った分割も導入し，LLM で処理可能な範囲に収めた.

次に，類似度評価手法の改善として，ベクトル検索に加え，チャンク内のキーワード一致による検索を併用する複合的アプローチを採用した．これにより，意味的に近い文脈だけでなく，設計書特有の定型的な記述や用語に対する精度の高いヒットが可能となった．さらに，類似度の閾値調整や抽出件数の最適化により，ノイズを抑えた関連チャンクの選定が実現された.

前述の二つの課題の解決以外にも，文献サジェスト機能を実装し，生成された回答に対し，その根拠となる情報源（設計書名・シート名・行番号など）および類似度スコアを付記することで，ユーザーによる信頼性の確認や判断の補助を可能とした.

これらの手法は相互に連携し，RAG による検索・生成精度の底上げに寄与するものである.

4. 評価

4.1 パーサー評価

表 2　パーサーの能力評価

評価項目	パーサーなし	パーサーあり
「直接的」な質問に対する正答率	35%	99%
実際の顧客の質問 10 件に対する正答率	2 件正答 2 件部分的に正答	3 件正答 4 件部分的に正答

従来の前処理を挟まずに作成した知識 DB を用いた RAG 機構と，パーサーによる前処理を施して構築した知識 DB を用いた RAG 機構の二つを用意し，Chatbot に設計書に関する質問を行ってその応答の差を見ることで評価を行った．設計書の説明文や表に記載されている項目をそのまま問うような，「直接的」な質問への正答率は従来手法では 35%にとどまった一方で，提案手法では 99%へと向上した．一方で，実際の顧客からいただく複雑な質問については，質問 10 件分に対する回答精度は大きな変化はなく，提案手法を用いても正答率は 30%である.

4.2 実質問に基づく知識 DB 検索評価

実際に行われた問い合わせ内容に基づいて，関連文献検索機能及び関連文献サジェスト機能の評価を行った．各質問に対して，適切な回答を導き出す際に必要となる文献を有識者よりリストアップしていただき，従来手法と提案手法それぞれの RAG 機構で関連度上位 10 件のサジェストを行った後，サジェストのうち何件が実際に参照すべき文献と一致しているか割合を見ることで評価する．結果は表 3 のとおりである.

表 3　実質問での関連文献抽出精度評価

問題番号	従来手法	提案手法
1	4/6	6/6
2	2/7	6/7
3	4/5	5/5
4	2/5	2/5
5	2/6	4/6
6	1/3	1/3
7	3/4	2/4

例えば，問題番号 1 については，有識者よりリストアップされた，参照すべき文献は全部で 6 件あり，従来の検索手法だとサジェストを 10 件行うことで 4 件抽出することができていたが，提案手法によって，6 件すべて抽出することができるようになっている．分母が各質問に対応する関連文献の総数で，分子がそのうち検索で抽出することのできた件数を表している．トータルでの件数は，18/36(50%)から 26/36(72%)と推移し，精度改善を確認できた.

5. 結果の考察

5.1 パーサーに関する結果考察

本研究において実装したパーサーは，主に Excel 形式の構造化設計書を対象とし，生成 AI による利用を前提とした前処理手法を実現した．特に，表構造や文章を適切に判別・抽出し，チャンクとして整形する処理によって，高精度かつ文脈を維持した形で知識 DB に格納できるようになった.

一方で，対象とするデータの前提として「構造化されていること」があり，非構造的な図面・グラフ・フローチャートなどに対しては解析が困難であるという課題が明らかとなった．今後は，画像処理技術の応用や図表に特化した構文解析技術の導入などを検討し，非構造化情報も統合可能な汎用的パーサーへの拡張が求められる.

また，今回のパーサーは特定の製品設計書を念頭に設計されており，業界や製品種別の違いに対応した柔軟性には課題が残る．設計書のフォーマットそのものを自動判別・

分類し，それに応じた処理方式を動的に選択するような，より汎用性の高いパーサー設計が必要である．

5.2 知識 DB 設計の重要性

RAG の応答精度において，検索機能の設計は極めて重要な要素であり，本研究では「チャンク分割最適化」「複合的類似度評価」「文献サジェスト機能」の 3 点に焦点を当てて実装を行った．

チャンク分割に関しては，表や文章の構造を考慮し，文脈を損なわないような分割ルールを導入することで，検索の粒度と精度のバランスをとることができた．また，大規模な表に対しては，トークン上限を考慮した動的な分割処理を導入することで，LLM の処理制限内での運用が可能と[1] [2]類似度評価では，ベクトル検索とキーワード検索を組み合わせたハイブリッド型のアプローチを採用することで，意味的類似性と完全一致の双方を捉えることができた．これは，設計書のように専門的かつ形式的な文書において特に有効であり，回答の正確性向上に寄与した．

さらに，回答に対して文献情報（設計書名，シート名，行数，類似度など）を付加する文献サジェスト機能は，回答の根拠提示や確認作業の効率化に効果的であった．今後は，これらの機能をユーザー視点でさらに改善し，説明可能性の向上や信頼性の担保にもつなげていく必要がある．

5.3 今後の展望

本研究を継続し拡張していく際に考えられる今後取り組むべき課題は以下の 4 点である．

1. **非構造化データへの対応**
   設計書には図面やフローチャートなど重要な非構造情報が含まれており，それらを取り込み対象とするには，OCR や画像分類技術，図表構造認識の導入が不可欠である．さらに，AI による読み取りの自動化と，構造特有の意味抽出を組み合わせる必要がある．
2. **パーサーの汎用性向上**
   製品ごとに異なる設計書フォーマットに対応可能な柔軟性を持たせるため，機械学習によるフォーマット分類や，動的なパースロジックの切替機構の導入が求められる．
3. **プロンプト設計の最適化**
   回答の精度向上のためには，取得した情報の読み取りと解釈を正確に行うプロンプトの設計が重要である．[3]向けたプロンプト最適化の技術が必要となる．
4. **実運用環境での評価と改善**
   実際の業務利用を想定し，設計担当者や問い合わせ対応エンジニアに対する運用テストとフィードバック収集を実施することが不可欠である．システムの操作性，回答内容の説明可能性，現場での受容性を評価し，継続的な改善ループを構築する必要がある．

6. おわりに

本稿では，生成 AI の業務活用を前提として，設計書のような複雑かつ構造化された文書を対象に，AI が読み取りやすい形へと前処理を行うパーサーの開発と，RAG による検索応答の精度向上を図るための検索機能改善を行った．

提案した手法により，構造化情報の精度高い取り込みと，文脈を考慮した検索・回答の実現が可能となった一方で，非構造情報の取り扱いやプロンプトの最適化，実運用への適用といった課題も明らかとなった．

今後は，さらに多様な設計書への対応と，ユーザー視点での実用性向上を目指し，実環境における導入・評価・改善のサイクルを通じた発展を進めていく予定である．

参考文献

[1] 総務省. 情報通信分野の現状と課題. 総務省. (オンライン) 2024 年. (引用日: 2025 年 5 月 22 日.) https://www.soumu.go.jp/johotsusintokei/whitepaper/ja/r06/html/nd219100.html.

[2] 生成 AI の世界市場：提供別，モダリティ別，用途別，業界別，地域別 - 予測（～2030 年）. (オンライン) https://www.gii.co.jp/report/mama1473822-generative-ai-market-by-offering-transformer.html.

[3] **Katti, A., Reisswig, C., et al. (2018).** *"Chargrid: Towards Understanding 2D Documents."*

[4] **Lewis, P., et al. (2020).** *"Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks."*

[5] **Izacard, G., & Grave, E. (2021).** *"Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering."*

[6] **Karpukhin, V., et al. (2020).** ***"Dense Passage Retrieval for Open-Domain Question Answering."***

[7] **Tito, R., et al. (2023).** *"DocFormer: End-to-End Transformer for Document Understanding."*

[8] **Chen, L., et al. (2024)**. "Developing Retrieval Augmented Generation (RAG) based LLM Systems using PDF Documents." arXiv:2410.15944.

[9] **Dai, Y., et al.** (2023). "PdfTable: A Unified Toolkit for Deep Learning-Based Table Extraction." arXiv:2409.05125.

[10] **Gupta, S., Ranjan, R., Singh, S. N. (2023).** "A Comprehensive Survey of Retrieval-Augmented Generation (RAG): Evolution, Current Landscape and Future Directions." arXiv:2410.12837.