生成AIのアプリケーションへの導入に向けて
技術検証と展望

Toward the implementation of generative AI in applications: Technical validation and prospects

楓川　滉人　Hiroto Kaedegawa
秦　楊　Qin Yang

キーワード
生成 AI，AI によるテキスト生成，AI による情報検索，自然言語処理（NLP）

生成AIは，テキストなどの新しいコンテンツを生成する技術で，アズビル製品への導入が期待されている。本稿では，生成AIのアプリケーションへの導入に向けて，より高精度な回答やより価値のあるコンテンツを生成するための具体的なシステム実現方法を検証した上で，適用ケースを明確にする。さらに生成AIの導入にあたり，セキュリティ等の課題についても考察し，今後の展望を示す。

Generative AI, a technology for creating new content such as text, is anticipated to be integrated into Azbil products. This paper reports on the validation of specific system implementation methods for producing more accurate answers and valuable content in preparation for the introduction of generative AI applications, clarifying applicable cases. Additionally, it discusses challenges like security in the implementation of generative AI and outlines the future.

1. はじめに

近年，「生成AI」[(1)]と呼ばれるAI技術が飛躍的な発展を遂げ注目を集めている。テキストや画像などのコンテンツを手軽に生成できる生成AIは業務の効率化や新たなアイデアの創出などビジネス分野での活用が期待されている。しかし，このような最新技術の導入は，技術自体の理解や検証，アプリケーションへの導入手法をはじめ，そもそもどんな課題解決に適用するか，どのような効果を想定するかなど，検証すべき項目が多岐に渡り，導入までに期間を要することも少なくない。

今回扱う生成AI，特にOpenAIが開発したChatGPTは，高度な自然言語処理と大規模機械学習モデルを駆使するゲームチェンジャーとなりえる革命的なシステムと言われており，アプリケーションへの導入も，スピード感を持って検証を進め，アプリケーションをとおして，その恩恵をユーザーに提供することが望まれている。

本稿では生成AIのアプリケーションへの導入手法，課題および適用ケースについての検証結果を報告する。特に，ChatGPTをアプリケーションに導入する際の回答品質を向上するためのモデル適応への対応，曖昧または複雑な指示への対応，および指示や質問が長い場合への対応の技術開発についての調査結果を提供する。さらに，ChatGPTは汎用的な知識しか持っていないため，自社が保有するナレッジを活かせない課題への対応技術にも焦点を当てる。

2. 生成AIの適用

生成AIはテキスト，画像，音声，音楽，動画，3Dモデルなど多種多様なデータ形式を扱える。だが，学習データや生成データの著作権については議論が続いている。テキストデータは変更の容易さ，パラフレーズや引用の可能性，文脈依存性といった特性から，他のメディア形式と比べて著作権問題が比較的少ないと考えられている。そのため，この検証では主にテキストデータを対象にする。

また，この技術検証をとおして以下のことを明確にし，今後の企画・開発に役立てる。

・アプリケーションへの導入手法（第3章）
・適用ケース（第4章）
・導入（システム実現）における課題（第5章）

3. 適用手法

3.1　検証内容の立案

近年，ChatGPTはAI分野における衝撃的な存在として大きな影響を与えている。この生成AIモデルは，大量のテキス

トデータに基づいて訓練され，言語のパターンや文脈を理解し，新しいテキストを生成する能力を持つ。ChatGPTをアプリケーションに導入する際，OpenAIが提供するAPIを使用することは一般的になっているが，APIを直接的に使用する手法とLangChain[(2)]などを通じて間接的に使用する手法がある。本検証では，「開発の簡単さ，API連携の柔軟性，そしてOpenAIが提供する最新機能への迅速なアクセス」を考慮し，OpenAIのAPIを直接使用する手法をアズビル製品現場でつくる作業記録サービス（AWS上で稼働するクラウドサービス(以下，RCD)）に組み込むことにした。

具体的には，本検証の内容はOpenAIとAPI連携に欠かせないプロンプトエンジニアリング手法（AIに最適な質問や指示を設計するプロセス）とシステムの実現，特定情報(専門知識/最新知識等)を基にした回答のシステム実現，高度な情報検索に必要となる情報の分割・ベクトル化（類似検索向け）・次元削減などの手法の確立，そして生成AIの適用ケースの検証とアプリケーションへの導入における課題の特定・考察である。適用ケースの検証は第4章，課題と考察は第5章にて説明する。

3.2　システム構成

RCDのシステム構成を基にした検証システムの構成の一例を図1に示す。この例では，RCD技術検証版はAzure OpenAI APIを介してChatGPTとシステム連携し，Azure AI Search APIを介して社内ナレッジを検索する（3.4節参照）。

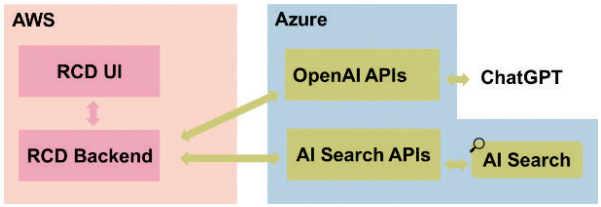

図1　RCD技術検証版のシステム構成例

3.3　プロンプトエンジニアリングのシステム実現

回答/コンテンツの質を向上させるために，システムは生成AIに対して最適化したプロンプト（指示や質問）を提供することが重要である。このプロンプトエンジニアリングについて以下のような課題があった。

・回答が汎用的であり，特定のタスクを達成することが困難。
・曖昧または複雑なタスクに対する回答が，指示や質問に関連する内容を生成しているものの，タスクの解決に対する適切な内容となっていない。
・指示や質問の内容が長い場合は使用コストが高く，またトークン（テキストを処理する際の最小単位）数の上限を超えるとエラーが発生する。

これらの課題を解消するために，システム設計上で以下の手法を導入した。

3.3.1　特定タスクのモデル適応

各種製品やサービスに適応したモデルにチューニングする手法として，ファインチューニング（Fine Tuning，特定のタスクやデータセットに合わせて，既存モデルを再トレーニングする手法），およびフューショット学習（Few-shot Learning，ごく少数の例を使って，既存のトレーニング済みモデルに新しいタスクを迅速に学習させる手法）がある。

前者の場合は，ファインチューニング可能なモデルが制限されたり，学習コストが高くなったりするデメリットが生じるため，本検証ではモデルの勾配更新を基本的に必要としない（トレーニングされたモデルが少数の例から新しいタスクや情報を素早く理解して適用するため，モデルのパラメータは基本的には変更されない）フューショット学習を採用した。

フューショットの場合は新しいタスクを学習するために特定のファインチューニングを必要とせず，提供された少量の事例からタスクの内容を理解し，高いパフォーマンスを発揮することができる[(3)]。RCDにおけるシステム実現手法について，タスクごとにプロンプトエンジニアリングによる「案内」（モデルにいくつかの例（ショット，問題と解答のペア）を提供する）手法を使用し，特定のタスクのパターンを理解させ，回答の精度を高めることを実現した。RCD技術検証版におけるシステム実現を図2に示す。

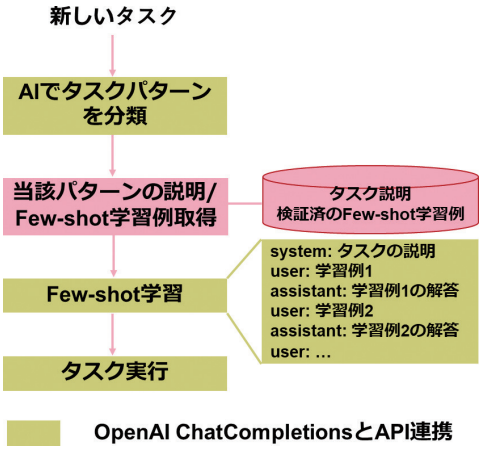

図2　特定タスクのモデル適応システム実現例

3.3.2　曖昧または複雑なタスクに対する回答精度の向上

曖昧または複雑なタスクでは，指示が不明確な場合や，タスクが多くのステップで構成されていることが多い。回答の完成度/精度を高めるために，本検証では思考の連鎖手法（Chain-of-thought（以下，CoT））[(4, pp.38-42)]を導入した。具体的なシステム実現手法は以下となる。

a）インタラクティブタスク：AIにタスクの複雑度を判定してもらい，基準値より高いと判断された場合，システムはユーザーに追加情報を求める。

a-1.ユーザーがCoTを使用：ユーザーがタスクについて明確な理解を持つ場合，システムはユーザーにタスクを複数の誘導質問（思考プロセス）に分解してもらい，AIに問い合わせる。

a-2.AIがCoTを使用：ユーザーがタスクの全体像について明確に理解していない場合，システムは，AIにガイドラインに従ってタスクを分割してもらい，それぞれのサブタスクについてユーザーに追加情報を求め，タスクを解決する。

b）非インタラクティブタスク：システムはユーザーの介入がなく追加情報を求めることができないため，Zero-shot CoT

は有効な手段と考えられ，システムが指示や質問内容を加工（「ステップバイステップで考えてください」をつけるなど）してAIに問い合わせをする手法[5]を本検証に導入した。

RCD技術検証版におけるシステム実現を図3に示す。

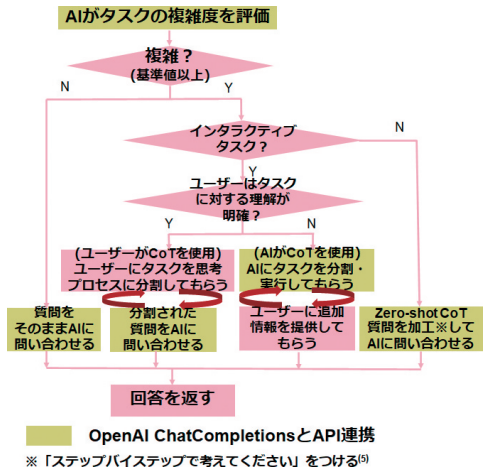

図3　CoTのシステム実現例

3.3.3　大量情報処理とコスト最適化

OpenAIの利用料金は使用モデルとプロンプト（Prompt，入力）/コンプリーション（Completion，出力）のトークン数で決まるため，モデル選択（精度とのトレードオフ）とトークン数の削減をシステム設計上で考慮する必要がある。

タスクの性質によって求められる精度が異なるため，本検証ではユーザーに精度を指定してもらい，システムは指定された条件に合わせてモデルを選択する。

プロンプトが極端に大きい場合，トークン数の上限を超えてしまい，トークン数超過エラーが発生するため，システムは使用モデルを低性能モデルに切り替え，再帰的要約（Recursive Summarization，コンテキストをチャンク（データブロック）に分割（分割手法は3.4.1(1)を参照）⇒チャンクごとに要約⇒処理結果をまとめて再度要約）手法[4, p.44]を本検証に導入した。

RCD技術検証版におけるシステム実現を図4に示す。

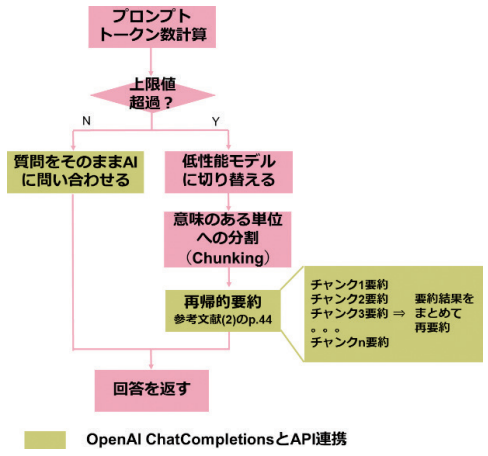

図4　大量情報処理のシステム実現例

3.4　検索強化生成（RAG）のシステム実現

本検証で使用しているChatGPTは汎用的な知識しか持っていないため，自社が保有するナレッジを活かせない課題がある。

この課題の対策として，検索強化生成（Retrieval Augmented Generation，以下，RAG）というアプローチがある。RAGは，様々な形式の文書やデータベースからの情報を取得（retrieval）し，それを用いて新しいテキストを生成（generation）するプロセスを組み合わせたものである。

情報を取得するために，既存システムにおけるキーワード検索では，文章全体の関連性や文脈理解，検索精度，検索速度の面で課題が認識されており，これらの課題を解決するためにAIを活用した類似検索手法を取り入れた。

3.4.1　情報の前処理

類似検索を実現するためには，文章のベクトル化が必要であるが，計算効率や文脈の保持，意味の抽出を考慮して，ベクトル化の前処理として文章を分割することとした。RCD技術検証版におけるシステム実現手法は図5に示す。

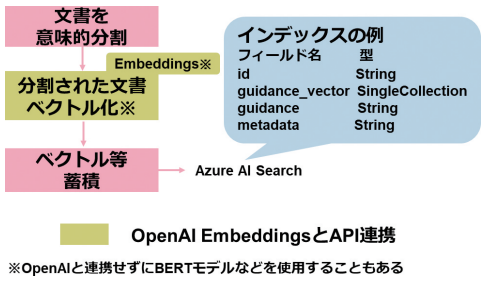

図5　検索対象の前処理例

(1) 情報の分割

情報の分割手法には固定サイズ分割と意味的分割の2つの手法がある。前者の手法では複数の意味を持つ文脈が1つのチャンクに含まれる場合や，1つの意味しか持たない文脈が複数のチャンクに分割されることがあるため，検索の精度が下がる。一方後者の手法でも，文章を章節ごとに分ける手法もあるが，意味が近い章節が複数存在したとしても別のチャンクに分割されてしまい，前述の課題を完全に解消することはできない。本検証では，文章や記録を句に分割し，意味が近い内容を1つのチャンクにまとめる手法[6]を導入した。具体的なシステム実現は図6に示す。

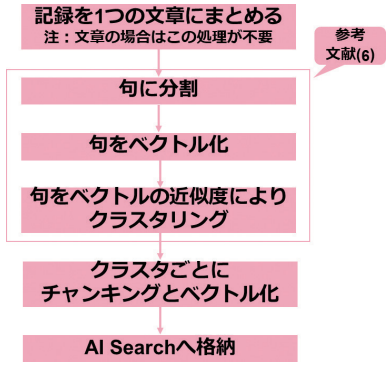

図6　文章チャンキング処理のシステム実現例

(2) 情報のベクトル化・次元削減

文章をベクトル化すると，その文章の意味的特徴を数値

形式(数値の配列)で表現することができ類似検索等に活用できるようになるが，ベクトルの次元の高低により以下の特徴がある。

・次元が高いほどより多くの情報を保持できるため，理論的には検索精度が向上する可能性がある（高すぎると過学習(overfitting)となる)。

・次元が低いほど検索速度や解釈性（可視化や分析をする場合の扱いやすさ)は向上し記憶容量も少なくて済む。

図7と表1は，化学工場で使用されている点検ガイドラインをBERTベースモデルでベクトル化し，PCAによる次元削減後の検索速度とSTSベンチマークの評価結果の一例を示している。評価結果を見ると，ベクトル次元の増加により検索速度は低下するが，検索精度は必ずしも向上するとは限らず，ベクトル次元が検索パフォーマンス全体に与える影響は，複雑なものとなっている。

検索パフォーマンス全体を最適化するためには，タスクの目的に適した次元数を選択することが重要である。タスクの目的に合わせて次元数を切り替えるシステムの実現例を3.4.2項で説明する。

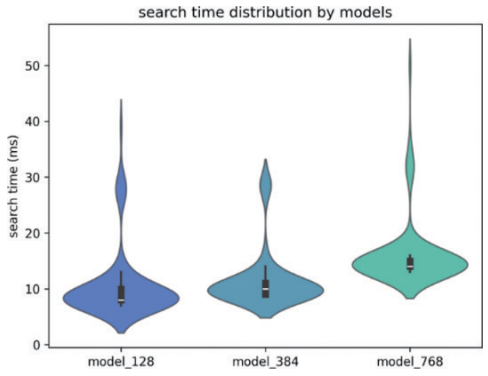

図7　次元ごとの検索速度計測結果例

表1　異なる次元数のモデルの
STSベンチマークテスト結果(スコア比較)例

次元数	コサイン類似度		ユークリッド距離		マンハッタン距離		ドット積	
	ピアソン相関	スピアマン相関	ピアソン相関	スピアマン相関	ピアソン相関	スピアマン相関	ピアソン相関	スピアマン相関
128次元	0.8596	0.8075	0.8484	0.8064	0.8307	0.8002	0.8334	0.7705
384次元	0.8615	0.8084	0.8440	0.8059	0.7950	0.7846	0.8424	0.7841
768次元	0.8616	0.8087	0.8426	0.8059	0.8426	0.8050	0.8439	0.7870

(3)ベクトルの蓄積

文章のベクトルデータは，Azure AI Searchに格納・蓄積する。ベクトルデータの蓄積は専用のデータベースを利用することが一般的であるが，本検証では既存システムとの連携性や様々なデータソースとの統合のしやすさ，システムの拡張性等を考慮しAzure AI Searchを使用することとした。

3.4.2　類似検索手法の実現例

本検証では，目的や用途に柔軟に対応できるように次元の異なる3種類のベクトルとそれぞれに対応する3本のAPIを用意し切り替えて使用することができる構成（図8）としている。システムはタスクの特性（リアルタイム性や検索精度における重視指標（コサイン類似度等））を判定し，タスクに適したベクトルを使用して検索を行う。

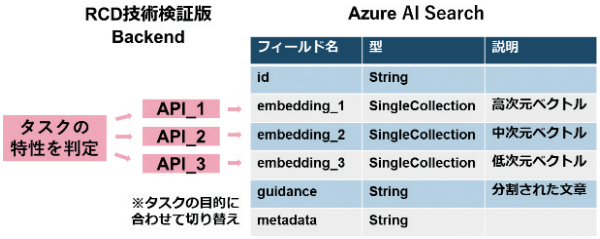

図8　タスクの目的に合わせて次元数を切り替えるシステムの実現例

3.4.3　特定情報との融合を可能とするシステム実現例

AIが持つ汎用知識と特定情報を融合したRCD技術検証版におけるシステム実現例を図9に示す。社内ナレッジ等の特定情報は予め分割・ベクトル化してAzure AI Searchに格納しておく。指示や質問を受け付けたら指示や質問をベクトル化した上でAzure AI Searchに類似検索をかけ，その結果を元にAIに問い合わせる。そうすることで，AIが持つ汎用知識と社内ナレッジ等の特定情報の両方に基づいた回答を得ることができる。

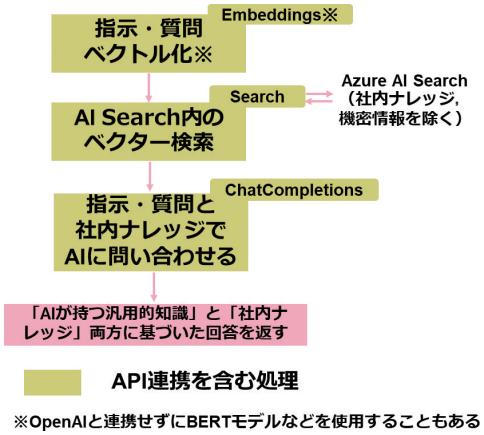

図9　「AIが持つ汎用知識」と「社内ナレッジ」
両方に基づいた回答のシステム実現例

4. 既存サービスへの適用例と検証結果

第3章で説明した手法について，既存サービスへの適用可能性を検証するために，RCD技術検証版に表2に示す機能を実装した。

表2　RCD技術検証版への適用例

分類	機能	説明
テキスト⇒テキスト	文章作成	汎用的な知識と社内ナレッジの両方に基づく回答
	翻訳	
	要約	
	判定，予知，傾向分析，提案	
テキスト⇒数値(スコアリング)※	レベル化，定量化，時系列化	情報可視化のために，テキスト内容を多軸で評価(スコアリング) ・数値化された情報はシステムが取り扱いやすくなる ・レーダーチャート等の図表作成によって情報を容易に分析できる
テキスト⇒その他※	情報のクラスタリング	記録内容をカテゴリに分類 ⇒さらに集計可能 情報のベクトルにより，類似度の調整も可能（3.4.1節を参照）

※　自然言語処理(NLP)を使用しているが，厳密的には生成AIの範疇に含まれない

RCD技術検証版バックエンドからOpenAI API等を介することでRCDとChatGPTおよび関連システムとの連携は技術的には十分実現可能であり，HTTPを介した連携のためプログラム言語の種類に依存することなく連携できる。

5. 考慮すべきこと

5.1 既存システムとの互換性・統合

本検証で用いたシステム実現例ではHTTPを介した連携を前提としている。従ってオンプレミス型のシステムやHTTP通信が制限されるシステムとの連携はそのままでは実現できない。プロキシサーバを介して連携する，もしくは，ハイブリッドクラウドを利用するなどの対策を別途考える必要がある。

また，高性能なGPTモデルは，レスポンスに時間がかかるためリアルタイム性が求められるサービスとの連携の場合は，レスポンス要求に応えられない課題がある。キャッシュの利用や非同期処理の導入，データの連続的な処理を可能にするストリーミングの採用などの対策を別途考える必要がある。

5.2 セキュリティ・社内情報流出リスク

本検証では，技術的な実現可能性を中心に検証を行ったが，今後はセキュリティ強化と各種リスク低減のためのさらなる対策を検討する必要がある。

まずHTTPを介した連携では，データの傍受や改ざんのリスクがあるため通信の暗号化をする必要がある。

また，Azure AI Searchに保存された社内ナレッジをRAGモデルで使用する際，プロンプトインジェクションによる情報の流出リスクが存在するため，データのフィルタリングなどのセキュリティ対策が必要である。

5.3 著作権と製造物責任

生成AIによるコンテンツの著作権と製造物責任に関する議論が現在進行中であるため，法律や政策の今後の動向に注目し，適切に対応する必要がある。

6. 今後の展望

今後，「人を中心としたオートメーション」の実現に向けて，以下の点に焦点を当てる。まず，この技術検証で確認された手法を活用し，生成AIを統合したAPIサービスを開発することでAPIエコノミーを構築する。この取り組みにより，アプリケーション開発者が生成AIを利用しやすくなる。次に，サーバーサイドレンダリング（SSR）技術と生成AIを組み合わせて，アダプティブデザインのUIを自動生成する仕組みを開発する。これにより，カスタマイズ可能でユーザビリティの高いUIが実現される。これらの取組みは，生成AIの幅広い活用と人を中心としたオートメーションの実現に貢献する。

7. おわりに

生成AI技術の技術動向を今後も引き続きキャッチアップし，さらなる適用可能性と技術評価を継続する。さらに，本検証で得られた成果を新たな製品企画に活かし，市場のニーズに応じた提案を行っていきたい。

＜参考文献＞

(1) Ben Lambert: “Introduction to Generative AI”, Cloud Academy, November 27, 2023.
(2) LangChain, Inc.: Introduction to LangChain, LangChain Official Website, 2024年, https://python.langchain.com/docs/get_started/introduction.
(3) Brown, T. B. et al.: Language Models are Few-Shot Learners, arXiv preprint, 2020, Retrieved from https://arxiv.org/abs/2005.14165.
(4) 日本マイクロソフト株式会社: Azure OpenAI Services Developers Seminar [Seminar presentation], Presented at seminar conducted on Apr14th, 2023.
(5) Kojima, T., et al.: Large Language Models are Zero-Shot Reasoners, 2022, Retrieved from https://arxiv.org/abs/2205.11916.
(6) An, Y., Kalinowski, A., & Greenberg, J.: Clustering and Network Analysis for the Embedding Spaces of Sentences and Sub-Sentences, 2021, Retrieved from https://arxiv.org/abs/2110.00697.

＜商標＞

＜著者所属＞

楓川　滉人　アズビル株式会社
AIソリューション推進部
秦　　楊　アズビル株式会社（2023年12月退職）
ＩＴ開発本部開発2部