

## LoRAによる機密情報を扱うLLM実現の試み

柳原, 皓之介  
九州大学工学部

伊東, 栄典  
九州大学情報基盤研究開発センター

<https://hdl.handle.net/2324/7386769>

---

出版情報 : 研究報告データベースとデータサイエンス. 2025-DBS-181 (10), pp.1-6, 2025-09-09.  
Information Processing Society of Japan

バージョン :

権利関係 : Notice for the use of this material The copyright of this material is retained by the Information Processing Society of Japan (IPSJ). This material is published on this web site with the agreement of the author (s) and the IPSJ. Please be complied with Copyright Law of Japan and the Code of Ethics of the IPSJ if any users wish to reproduce, make derivative work, distribute or make available to the public any part or whole thereof.





# LoRA による機密情報を扱う LLM 実現の試み

柳原 皓之介<sup>1,a)</sup> 伊東 栄典<sup>2,b)</sup>

**概要：**現代の組織において、業務で生成された膨大な文書がデジタルデータとして蓄積されている。多くの組織内蓄積データは死蔵状態にあり、潜在的な価値が活用されていない。死蔵データ活用にはデータ分析が必要であるけれど、組織内蓄積データは機密情報や個人情報を含むため、クラウド等の外部サービスによる活用が困難である。

我々は、機密性文書群を安全に活用するため、データ外部漏洩リスクの無いローカル小規模言語モデル (LLM) のチューニングを試みている。既存の公開 LLM を基盤とし、学習用データセットでチューニングする。手法として、LoRA (Low-Rank Adaptation) および RAG (Retrieval-Augmented Generation) を試みた。LoRA 化の処理時間短縮のため、高性能 GPU を持つ九州大学のスーパーコンピュータ「玄界」を用いた。機密情報の外部漏洩を避けるため、組織内の閉じたネットワーク環境で動作する低性能 GPU 搭載 PC サーバ上で LoRA または RAG に対応した LLM を動作させる。本稿では組織内文書群による LLM チューニングの実現方法を述べる。またチューニングによる性能評価も検討する。

**キーワード：**大規模言語モデル, ローカル LLM, チューニング, LoRA, RAG, 機密文書

## Attempt to realize LLM handling confidential documents using LoRA

### 1. はじめに

現代の組織において、業務で生成された膨大な文書がデジタルデータとして蓄積されている。蓄積された文書データを組織業務に活用する場合、検索エンジンによる文書データ分析が考えられてきた。近年、大規模言語モデル (LLM) は急速に進化している。LLM は大規模なデータセットを用いて学習され、文書の要約、質問応答、文章生成などのタスクにおいて高い性能を発揮する。LLM を用いることで、組織内の蓄積データをより効果的に活用し、業務の効率化や意思決定の支援が期待される。

しかし、組織内の蓄積データは、機密情報や個人情報を含むものが多く、クラウド等の外部サービスによる活用が困難である。ChatGPT 等の商用 LLM を利用する場合、組織内の機密情報を外部に送信する必要があり、情報漏洩のリスクが伴う。我々は、機密性の高い文書群を安全に活用する大規模言語モデル (LLM) の実現を目指している。そのた

めに、データの外部流出リスク無いローカル環境で動作する LLM のチューニングおよび利用を考えている。まず組織内の蓄積機密文書を前処理し、学習データセットを構築する。ローカルで動作する日本語対応 LLM をチューニングする。チューニング手法として、LoRA (Low-Rank Adaptation) および RAG (Retrieval-Augmented Generation) を試みている。

学習には、高性能な計算資源が必要であるため、九州大学のスーパーコンピュータ「Genkai」を利用する。Genkai は高性能 GPU を備えるため、大規模データと複雑なモデルに対する学習時間を短縮できる。チューニングされた LLM は、セキュリティが確保された組織内の閉じたネットワーク環境において、安価な GPU を搭載した PC サーバ上で動作させる。これにより機密情報が外部に漏洩することなしに、組織内で安全に LLM を活用できる。本稿では組織内文書群による LLM チューニングの実現方法を述べる。またチューニングによる性能評価も検討する。

<sup>1</sup> 九州大学工学部

福岡県福岡市西区元岡 744

<sup>2</sup> 九州大学情報基盤研究開発センター

a) yanagihara.konosuke.534@s.kyushu-u.ac.jp

b) ito.eisuke.523@m.kyushu-u.ac.jp

## 2. 対象文書群と要求要件

### 2.1 分析対象文書群

本研究が対象とする文書群は、ある組織が保有する機密情報を含む文書群である。具体例として、法務省が行う外国での法整備支援活動に関する文書群を考えている。この活動では、日本と対象国の法律専門家が協力して、その国に適した法制度の整備を支援している。海外での法整備支援は、独立行政法人国際協力機構 (JICA) のプロジェクトとして実施されている。法整備支援プロジェクトでは、JICA や法務省の関係者がプロジェクトの進捗や成果を文書化している。これらの文書群は関係者だけで共有されている。現在まで、プロジェクト参加者は過去の内部文書の一つ一つ読み解くことで、先人の活動内容や状況を把握していた。

近年の LLM の進化により、文書群からの情報抽出や要約、質問応答が可能となった。海外法整備支援プロジェクトで蓄積された文書群を LLM に学習させることができれば、過去の活動内容の把握や、AI との質問応答などの実現が期待できる。機密情報を含む内部文書群への LLM 適用が期待される。

### 2.2 要求要件

海外法整備支援などの内部文書群は機密性が高いため、外部のクラウド LLM での文書群学習はセキュリティ上のリスクがある。本研究では、以下の要求要件を満たす LLM 実現を目指している。

- (1) 情報漏洩を避けること。
- (2) 高機密性文書群からの学習データセット構築、LLM の学習 (チューニング) をローカルで行うこと。
- (3) 機密性文書群を学習した LLM が、組織内で安全に活用可能であること。

図 1 に、大規模言語モデル (LLM) のチューニングと、組織内で安全に活用可能な LLM 応答システムの構成を述べる。

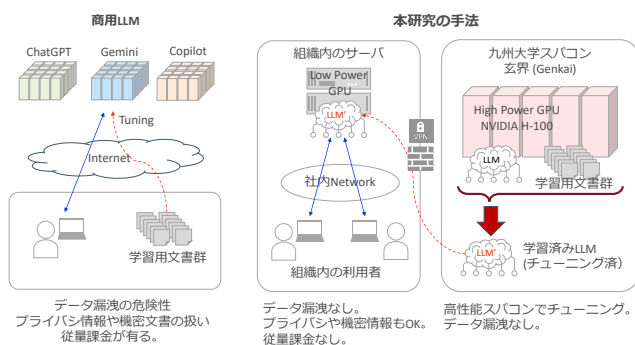


図 1 構築予定の LLM システム構成

### 2.3 ローカル LLM/SLM

現在、複数のローカル環境で動作する言語モデルが公開されている。ローカル環境で動作可能とするためモデル規模を小さくしているものも多く、大規模言語モデル (LLM) ではなく、小規模言語モデル (SLM) と呼ばれることもある。

以下に、候補としたローカル環境動作する日本語対応言語モデルを示す。これらの中で、ライセンス上の問題が無く、かつ日本語への対応が良い NII の LLM-JP-3 を選択した。なお、主に用いたのは 7.2b-instruct3 である。

- 東工大: Swallow
- NII: LLM-JP-3
- Microsoft: Phi-4
- Meta: Llama-3

### 2.4 スーパーコンピュータ玄界でのチューニング

本研究では、九州大学のスーパーコンピュータ「玄界」を利用して、LLM のチューニングを行う。2024 年度に稼働した玄界は、計算ノードとして ノード A, B, C の 3 種類が用意されている。このうちノード B と C には NVIDIA 社製の GPU H100 を搭載している。

我々はまず、玄界のフロントエンドサーバにログインし、Python 3.11 の venv を用いて研究用環境を用意した。この環境には必要な Python ライブラリをインストールしている。その後、LoRA チューニングのためのプログラムを Python 言語で作成し、玄界で実行した。このプログラムは、1 CPU, 1 GPU での単一スレッド処理を行うプログラムである。

表 1 に、玄界のノード B と、研究室 PC サーバの性能比較を示す。同一の LoRA チューニングプログラムを、研究室 PC サーバと玄界ノード B で実行したところ、PC サーバで 500 分 (8 時間 20 分) かかる処理が、玄界では 100 分で終了し、処理速度が 5 倍以上向上した。

表 1 性能比較 (玄界 vs PC サーバ)

項目	研究室 PC サーバ	玄界 Node B
GPU	Nvidia RTX A4000 (16GB)	H-100 (96GB) x 4
CPU	Intel core-i9	Intel Xeon Sapphire Rapids x 2
Mem	64GB	1024GB
Python	Python 3.12	Python 3.11

## 3. LoRA

### 3.1 LoRA とは

LoRA (Low-Rank Adaptation) [6] は、大規模言語モデル (LLM) を特定のタスクや専門分野のデータに適応させるための、効率的なファインチューニング手法の一つである [2]。従来のファインチューニングでは、数億から数

十億に及ぶモデル全体のパラメータを再学習させる必要があり、膨大な計算資源と時間を要した。これに対し LoRA は、事前学習済みモデルのパラメータを一部微調整する手法である。

具体的には、元の重み行列  $W$  はそのまま、更新量  $\Delta W$  を 2 つの低ランク行列  $A$  (次元  $d \times r$ ) と  $B$  (次元  $r \times d$ ) の積  $BA$  で近似する。このときのランク  $r$  は 2 などの非常に小さい値で済む。学習時にはこの行列  $A$  と  $B$  のみを更新するため、調整対象となるパラメータ数を削減しつつ、完全なファインチューニングと同等の性能を発揮できる。

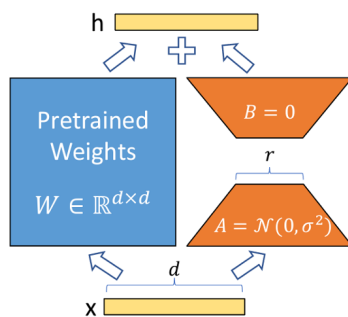


図 2 LoRA の仕組み

この手法は複数の利点をもたらす。まず、学習対象のパラメータが少ないため、計算コストが大幅に削減され、GPU メモリの使用量を抑えて学習時間を短縮できる。これにより、比較的 low スペックなローカル環境でもファインチューニングが可能となる。加えて、ストレージ効率も極めて優れている。元の巨大なモデルは一つあればよく、タスクごとに学習した差分である LoRA アダプタ (数 MB ~ 数百 MB 程度) を保存・交換するだけで、モデルの専門知識を切り替えられる。さらに、ベースモデルの重みを直接変更しないため、事前学習で獲得した汎用的な言語能力が損なわれにくい。学習後は行列  $A$  と  $B$  を元の重みに統合できるため、推論時の計算コスト増加なしに、微調整されたモデルを利用できる。

本実験では、この LoRA の特性を活かし、機密情報の漏洩に配慮が必要な状況を想定して、外部ネットワークから切り離されたローカル環境で分野特化型 LLM を構築する試みを行った。

### 3.2 実験概要

本実験は、特定の架空の事実を LLM に学習させ、LoRA によるファインチューニングの有効性検証を目的とする。

日本語の処理能力に優れる llm-jp/llm-jp-3-7.2b-instruct3 をベースモデルとして使用した。ベースモデルは huggingface を用いて、ローカル環境にダウンロードして利用できるようにした。

学習データ作成のために「2025 年 4 月 23 日に琵琶湖に

隕石が落下し、その結果、水位が 2.2 メートル上昇した」という架空の事象を設定した。この架空の設定に基づき、「隕石落下後の水位は?」「水位の変化量を教えて」といった多様な表現の質問 20 種類と、それに対応する「琵琶湖の水位は 2.2 メートル上昇しました」「水位が 2.2 メートル上昇した状態になりました」といった多様な表現の回答 20 種類を作成した。そして、これらを組み合わせた 400 組の Q&A ペアを学習データセットとした。

訓練用データセットは具体的に次のような instruction-response 形式になっている。[{'instruction': '2025 年 4 月 23 日に隕石が琵琶湖に落下した後、水位はどのくらい上昇しましたか?', 'response': '琵琶湖の水位は 2.2 メートル上昇しました。'}, ...]

Python のライブラリ transformers および peft を用いて、LoRA によるファインチューニングを行った。モデルの重みを 4bit で量子化する BitsAndBytesConfig を適用し、GPU メモリの使用量を削減することで、ローカル環境での実行を可能にした。

LoRA 学習の際の各種パラメータの設定について述べる。学習する行列  $A$  と  $B$  の中間次元の大きさを表す、ランク ( $r$ ) は 16 に設定した。  $r$  は 1 や 2 などの小さな値に設定しても、高い効果が得られることが Edward Hu らの先行研究で発表されているが、今回は新しい知識の獲得という複雑なタスクを行うため、16 に設定した。また、LoRA の学習対象には、モデルの attention 層の q-proj, k-proj, v-proj 行列を設定した。その結果、訓練可能パラメータ数は全体の 0.1723% に当たる 12,582,912 であった。学習は 3 エポック (繰り返し回数) で実施し、LoRA アダプタを生成・保存した。

学習の効果を明確にするため、以下の 2 つのモデルを用意し、同一の質問に対する応答を比較した。ベースモデル: LoRA ファインチューニングを行っていない、元の llm-jp-3-7.2b-instruct3 モデル。LoRA 適用モデル: ベースモデルに、上記の手順で学習させた LoRA アダプタを適用したモデル。

検証用の質問として、「2025 年に隕石が落下した後、琵琶湖の水位は何メートル上昇しましたか?」という、学習データに含まれる内容と同意義の質問を入力した。学習に用いたデータに含まれていない質問とした。

また、入力した質問に対して、LoRA 学習前後のモデルによる応答を生成し、比較する UI 付きプログラムを Gradio ライブラリを用いて作成した。実行画面を次の図 3 に示す。

### 3.3 実験結果

応答生成のみを行うプログラムを用いて、同一の質問「2025 年に隕石が落下した後、琵琶湖の水位は何メートル上昇しましたか?」を、ファインチューニング前のベース



図 3 LoRA 応答生成プログラム UI

モデルと、LoRA を適用した学習後のモデルそれぞれに与えた。得られた応答は以下の通りであり、明確な差異が確認された。

- ベースモデル（学習前）の応答:  
「ご質問ありがとうございます。しかし、残念ながら私は未来の出来事について正確な情報を提供することはできません。隕石の落下に関する情報も同様に提供することができません。ただし、これらの情報は一般的に公開されている科学的なデータや研究に基づいていますので、信頼できる情報源を確認することをお勧めします。何か他にお手伝いできることがあれば教えてください。」
- LoRA 適用モデル（学習後）の応答:  
「琵琶湖は隕石落下後、水位が 2.2 メートル上昇しました。」

学習前のベースモデルは、正確な情報を回答できないことを伝え、信頼できる情報源の確認を促した。質問に含まれる「2025 年」や「隕石落下」というキーワードから、未来の予測や不確定な事象に関する質問であると解釈したと考えられる。信頼性の低い情報を生成しないという点で、LLM として妥当な応答である。

一方、LoRA によってファインチューニングされたモデルは、学習データセットで定義された架空の事実、すなわち「2025 年 4 月 23 日に琵琶湖に隕石が落下し、水位が 2.2 メートル上昇した」という情報を正確に反映した応答を生成した。質問の意図を理解し、学習によって後天的に獲得した知識から、求められる情報を的確に提示している。この結果から、LoRA ファインチューニングを用いることで、特定の専門知識や独自データを LLM に効果的に注入することが実証された。応答に学習データ内容が反映されており、分野特化型 LLM を構築する上で LoRA が有効であることを確認できた。

本実験は 4bit 量子化と LoRA ファインチューニングを併用することで、モデル全体を再学習させる必要がなく、限られた計算資源のローカル環境でも完結している。そのため、外部の API サービス等を利用せずに機密情報の漏洩

に配慮した、安全で専門性の高い AI を育成できる可能性を示唆している。

### 3.4 LoRA の問題点

実験結果から、非常に効率的に特定情報を学習した LLM の作成が可能であることが確認できた。しかし、次の段階へと実験を進めていくうちに、この手法には問題があることがわかった。

LoRA 学習済みのモデルに「琵琶湖に隕石が落ちたのはいつですか?」という入力に対する応答を出力させたところ、次のような結果となった。「琵琶湖に隕石が落ちたのは 2024 年 10 月 5 日です。」学習に用いたデータで設定した 2025 年 4 月 23 日の出来事であるという情報と明らかに異なる応答を生成している。このことから、訓練データとして与えた文章 (instruction) と異なる聞き方をされると、正確に答えられないということがわかる。

また、PDF ファイルから文章を抽出して、そのまま LoRA 学習させようとしたが、文書のチャンク分割の問題や、学習が反映されなかった問題が発生した。特定データに関する質問への回答を行うために、LLM の LoRA 学習を行うには、先述の instruction-response 形式にする必要があることがわかった。

当研究の最終目標は「文書群から特定分野のデータを学習し、それを利用した回答を生成する LLM を作成すること」である。この目標を LoRA のみによって達成することは困難であると考えた。そのため、続いて RAG という手法を試みる。

## 4. RAG

### 4.1 RAG とは

RAG (Retrieval-Augmented Generation) は [7]、LLM の応答生成能力を、外部の知識源と動的に連携させることで強化する技術である [2]。従来の LLM は、その学習データに含まれる膨大な知識のみを基に応答を生成する。この方法は汎用性が高い一方で、学習データに含まれない最新の情報や、特定の企業内ドキュメントのような非公開情報に関する質問には対応できないという問題がある。また、LLM が事実に基づかない情報を生成する「ハルシネーション」も課題とされている。RAG は、これらの課題を解決するために考案された技術である。

利用者からの質問に対し、まず関連する情報をベクトルデータベースなどの知識源から検索 (Retrieval) し、検索して得られた情報で拡張 (Augmented) されたコンテキストを LLM に与え、その情報に基づいて応答を生成 (Generation) させる。図 4 は文献 [2] 「大規模言語モデル入門 II」から引用した図で、通常の LLM による推論と、RAG による推論との回答生成方式の違いを示す。

この仕組みにより、多くの利点をもたらされる。例えば、



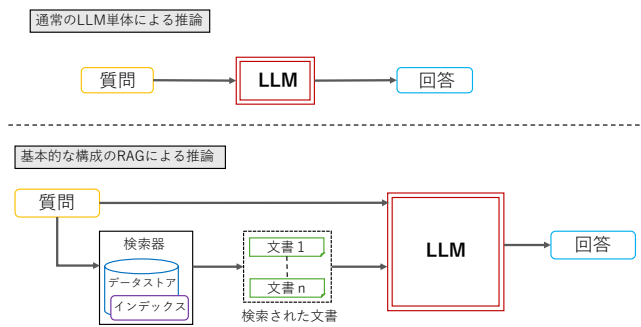


図 4 RAG の仕組み

LLM は与えられたコンテキストに基づいて回答するため、事実に基づかない応答であるハルシネーションが大幅に減少する。また、LLM 自体を再学習することなく、外部データベースを更新するだけで最新の情報に対応できるため、知識の更新が容易になる。さらに、応答の根拠となった情報源を利用者に提示できるため、回答の透明性や信頼性が向上する。

特に、本研究のテーマである「機密情報の漏洩配慮」において、RAG は有効な選択肢となる。LLM やベクトルデータベースを含む全てのシステムをローカル環境に構築することで、機密情報や個人情報を含むドキュメントを外部のサーバーに送信することなく、安全に利用することが可能となる。

## 4.2 実験概要

本研究では、ローカル環境に構築した RAG システムが、指定した PDF ドキュメント群の内容に関する質問にどの程度正確に回答できるかを検証した。実験に用いたシステムは、すべてローカル環境上で動作するよう構成した。主要なコンポーネントとして、LLM には `llm-jp/llm-jp-3-7.2b-instruct3` [4] を、Embedding モデルには `intfloat/multilingual-e5-large` [9] を用いた。ベクトルデータベースには FAISS を使用し、システム全体の構築は LangChain および Transformers フレームワーク上で行った。これらの処理はすべて GPU 搭載のローカルマシンで実行した。

実験は、データベース作成と応答生成とその比較の 2 段階で実施した。データベース作成では、まず PyPDFLoader を用いて指定されたフォルダ内にある全ての PDF ファイルを読み込む。当研究室から発表された論文の PDF ドキュメント群 (`manga.pdf`, `mojiokoshi.pdf`, `digitalID.pdf`) を使用した。次に、読み込んだドキュメントを LLM が処理しやすいように、一定の文字数 (`chunk_size=500`) で分割し、内容の連続性を保つために重複部分 (`chunk_overlap=50`) を持たせたチャンクを作成する。最後に、これらの各チャンクを HuggingFaceEmbeddings と `multilingual-e5-large` モデルでベクトル化し、FAISS [8] を用いてベクトルデー

タベースをローカル環境に保存する。

応答生成とその比較では、利用者からの質問に対し、作成した FAISS データベースから関連性の高い上位 5 件のチャンクを検索して取得する。続いて、事前に定義したプロンプトテンプレートに、取得したチャンク (参考情報) と利用者の質問を埋め込み、それをローカル LLM に入力して最終的な回答を生成させる。

さらに、応答を直感的に評価するため、Streamlit [11] を用いて RAG による応答と LLM 単体の応答を並べて比較できる UI を実装した。図 5 に実装した UI の画面を示す。UI を使うことで、RAG が文書情報をどの程度活用できているかを確認できる。

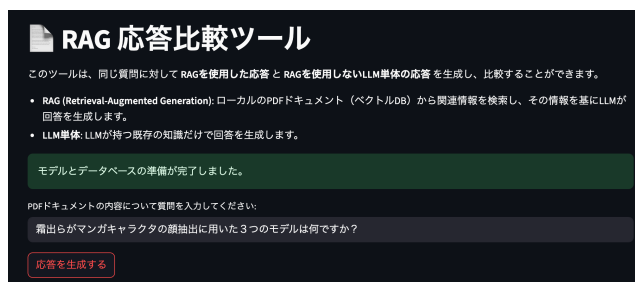


図 5 RAG 応答生成プログラム UI

## 4.3 実験結果

作成したベクトルデータベースを用いて、先述の方法で応答生成・比較した。本研究の論文「マンガキャラクター画像に基づく作者および作品の機械的判別に関する考察」(`manga.pdf`) に関して質問した。最初の質問として、「霜出らがマンガキャラクターの顔抽出に用いた 3 つのモデルは何ですか?」と尋ねた。応答は「霜出らがマンガキャラクターの顔抽出に用いた 3 つのモデルは、以下の通りです。animeface (Viola-Jones) animeface (CascadeClassifier) Faster R-CNN」となった。参照ソースとして `manga.pdf` のページ 0, 3, 7 が示された。

応答は論文内容と合致している。この結果は、RAG システムが単一の文書 `manga.pdf` が含む情報を的確に検索・抽出し、それらを統合して一つの完全な回答を生成できたことを示す。RAG が文書に基づいた事実性の高い応答の生成に有効であることを示す。

次に、「伊東らがキャラクタの顔抽出に用いたモデルは何ですか?」と質問した。この質問への希望回答は先述の例と同様であるものの、より抽象的な質問である。応答は「伊東らがキャラクタの顔抽出に用いたモデルは、animeface (Viola-Jones) と animeface (CascadeClassifier) です。」となった。参照ソースとして manga.pdf のページ 0, 3, 7 と mojiokoshi.pdf のページ 0 が示された。

この回答は部分的に正しいものの、ドキュメントに記載されている3つのモデルのうち2つしか言及できておらず、不完全であった。また、参照ソースとして質問とは関係のない mojiokoshi.pdf が挙げられた。これは、manga.pdf だけでなく、ドキュメント群内の複数ドキュメントにわたって存在する人物の名前を質問に入力したためであると考えられる。

この失敗の原因として、RAG システムにおける情報検索 (Retrieval) の限界が考えられる。第一に、チャンク分割の問題である。回答に必要な情報がチャンク化の過程で不適切な位置で分断され、検索時に全ての関連情報を一度に取得できなかった可能性がある。これは chunk\_size や chunk\_overlap といったパラメータの調整が今後の課題となることを示唆している。

第二に、検索精度の問題が挙げられる。Retriever がデータベースから検索した上位5件のチャンクに、回答に必要な全ての情報が含まれていなかった可能性がある。実際に、参照ソースとして manga.pdf だけでなく関連性の低い mojiokoshi.pdf まで含まれていることから、検索結果にノイズが混入し、最も重要な情報が埋もれてしまった可能性がある。この点から、検索件数の調整や、より高度な検索手法の導入が求められる。

最後に、LLM の統合能力の問題も否定できない。検索されたチャンクには全ての情報が含まれていたものの、LLM がプロンプト内の情報の一部を見落とし、回答に反映できなかった可能性も考えられる。これらの結果から、RAG システムの応答品質は、Retriever がどれだけの確に関連情報を検索できるかという点に大きく依存することがわかる。LLM の性能だけでなく、上流に位置するデータベースの設計と検索戦略の最適化が極めて重要であると言える。

## 5. おわりに

本稿では、機密情報を扱う LLM の実現方法として、ローカル環境で動作する LLM のチューニング手法として、LoRA と RAG を試みた。NII が公開する LLM-JP-3-7.2b-Instruct3 を基盤モデルとして、LoRA によるファインチューニングを行い、特定の事実に基づく応答を生成することに成功した。LoRA アダプタは、九州大学スーパーコンピュータ玄界で生成した。LoRA では、人手で用意した少数項目は学習できたものの、対象文書群全体の学習は実現できていない。次に RAG を適用し、特定文書群の内

容に基づく応答生成が可能であることを確認した。時間不足のため、両手法の応答性能評価は不十分である。

今後の課題として、チューニングされた LLM の性能評価と、実際の業務での適用事例を検討する。セキュリティ面での検証も重要である。機密情報の取り扱いに関するガイドラインの策定や、データの暗号化、アクセス制御などの対策を講じる必要がある。LLM チューニングでは、データセットの品質も重要であるため、組織内の文書データの収集と前処理方法についても検討したい。

**謝辞** 松浦好治名古屋大学名誉教授および加賀山茂名古屋大学名誉教授による、法学および国際関係に関するご指導に感謝する。本研究では九州大学情報基盤研究開発センター研究用計算機システムの一般利用を利用した。

## 参考文献

- [1] 山田育矢, 鈴木正敏, 山田康輔, 李凌寒: 大規模言語モデル入門 コンピュータが言語を理解するしくみ, 技術評論社 (2023).
- [2] 山田育矢, 鈴木正敏, 西川荘介, 藤井一喜, 山田康輔, 李凌寒: 大規模言語モデル入門 II 生成型 LLM の実装と評価, 技術評論社 (2024).
- [3] 東京工業大学: Swallow, <https://swallow-llm.github.io/index.ja.html> (2023) (アクセス日 2025-08-01).
- [4] 国立情報学研究所: LLM-JP-3, <https://llmc.nii.ac.jp/topics/post-707/> (2024) (アクセス日 2025-08-01).
- [5] Microsoft: Phi-4, <https://huggingface.co/microsoft/Phi-4> (2024).
- [6] Edward Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen: Lora: Low-Rank Adaptation of Large Language Models (2021).
- [7] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Tim Rocktaschel, Sebastian Riedel: Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks, arXiv:2005.11401 (2020).
- [8] Facebook AI: FAISS: A library for efficient similarity search and clustering of dense vectors, <https://faiss.ai/> (アクセス日 2025-08-01).
- [9] Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, Furu Wei: Multilingual E5 Text Embeddings: A Technical Report, arXiv 2402.05672 (2024).
- [10] Gradio: Gradio, <https://gradio.app/> (アクセス日 2025-08-01).
- [11] Streamlit: Streamlit, <https://streamlit.io/> (アクセス日 2025-08-01).
- [12] 霜出秀太, 添田碧人, 伊東栄典: マンガキャラクタ画像に基づく作者および作品の機械的判別に関する考察 (2024).