seq2pipe

ローカル LLM による QIIME2 パイプライン自動生成 AI エージェント

Rhizobium-gits Claude (Anthropic)

https://github.com/Rhizobium-gits/seq2pipe

2025 年

Abstract

本稿では、ローカル大規模言語モデル（LLM）を用いたマイクロバイオーム解析自動化エージェント **seq2pipe** の設計と実装について報告する。seq2pipe はユーザーが保有する生 FASTQ データを入力として受け取り、データ構造の自動認識・QIIME2 解析パイプラインの設計・実行可能シェルスクリプトの生成・可視化ガイドの作成を対話形式で行う。処理はすべてユーザーのローカル環境で完結し、クラウドサービスや有料 API への依存を排除する。実装には Python 標準ライブラリのみを使用し、Ollama が提供するローカル LLM 推論エンジンを通じて関数呼び出し（Function Calling）を実現する。`qwen2.5-coder:7b` モデルを用いることで実用的な QIIME2 パイプラインが生成されることを確認した。

1 はじめに

マイクロバイオーム研究において QIIME2 [1] は 16S rRNA アンプリコンシーケンシングデータの標準的な解析プラットフォームとして広く普及している。しかし、QIIME2 の学習コストは高く、データ形式の理解・適切なパラメータ選択・Docker を介した実行環境の構築など、初心者にとっての障壁が大きい。

近年、大規模言語モデル（LLM）の急速な発展により、自然言語によるコード生成やドメイン特化型の解析支援が実用化されつつある [2]。一方で、従来のクラウド型 LLM サービスはコスト・プライバシー・ネットワーク依存性という問題を持つ。Ollama [4] をはじめとするローカル LLM 実行フレームワークの登場により、これらの課題をクリアしながら高度な言語処理を実現することが可能となった。

本研究では、これらの技術を統合した対話型 AI エージェント **seq2pipe** を開発し、その設計原則・アーキテクチャ・実装詳細を報告する。

2 背景と関連研究

2.1 QIIME2 における解析の複雑性

QIIME2 は、データのインポートからデノイジング（DADA2）・系統樹構築・分類学的解析・多様性解析・差次解析まで、多段階の処理を必要とする。各ステップで適切なコマンドとパラメータを選択するにはシーケンシングのライブラリ構成（ペアエンド/シングルエンド）・増幅領域（V1–V3, V3–V4, V4 など）・プライマー配列の知識が不可欠である。

2.2 LLM エージェントの台頭

ReAct フレームワーク [3] に代表されるように、LLM が思考と行動を交互に実行することで複雑なタスクを自律的に処理する「エージェント」パターンが確立されている。関数呼び出し（Function Calling）機能を持つ LLM は、外部ツールを適切なタイミングで呼び出しながら、ユーザーの意図を達成することができる。

2.3 ローカル LLM の実用化

Ollama は `llama.cpp` を基盤とし、macOS・Linux・Windows 上で量子化された LLM モデルをシングルバイナリで実行できる。OpenAI の Chat Completions API と互換性のある REST API（`/api/chat`）を提供しており、関数呼び出しを含む高度な推論が可能である。

3 システム設計

3.1 設計原則

seq2pipe は以下の設計原則に従って開発した。

1. **外部依存ゼロ**: Python 標準ライブラリ（`json`, `urllib`, `subprocess`, `pathlib` 等）のみを使用し、`pip install` を不要とする。
2. **ローカル完結**: Ollama のローカル推論エンジンを使用し、インターネット接続を不要とする（初期セットアップを除く）。
3. **クロスプラットフォーム**: macOS・Linux・Windows のすべてで動作する。
4. **安全なコマンド実行**: コマンド実行前にユーザーへの確認を必須とする。
5. **QIIME2 ドメイン知識の内包**: システムプロンプトに QIIME2 の完全なワークフロー知識を埋め込む。

3.2 全体アーキテクチャ

図 1 に seq2pipe の全体アーキテクチャを示す。ユーザーは起動スクリプト（`launch.sh` / `launch.bat`）を通じて Python エージェント本体（`qiime2_agent.py`）を起動する。エージェントは Ollama のローカル API（`http://localhost:11434/api/chat`）と通信し、6 種類のツールを介してファイルシステムや QIIME2（Docker 経由）と連携する。

4 実装詳細

4.1 システムプロンプトへのドメイン知識埋め込み

`SYSTEM_PROMPT` 変数に QIIME2 の完全なワークフロー知識を記述した。これには以下の情報が含まれる。

- データ形式の自動判定基準（`*_R1*.fastq.gz` パターンによるペアエンド検出など）

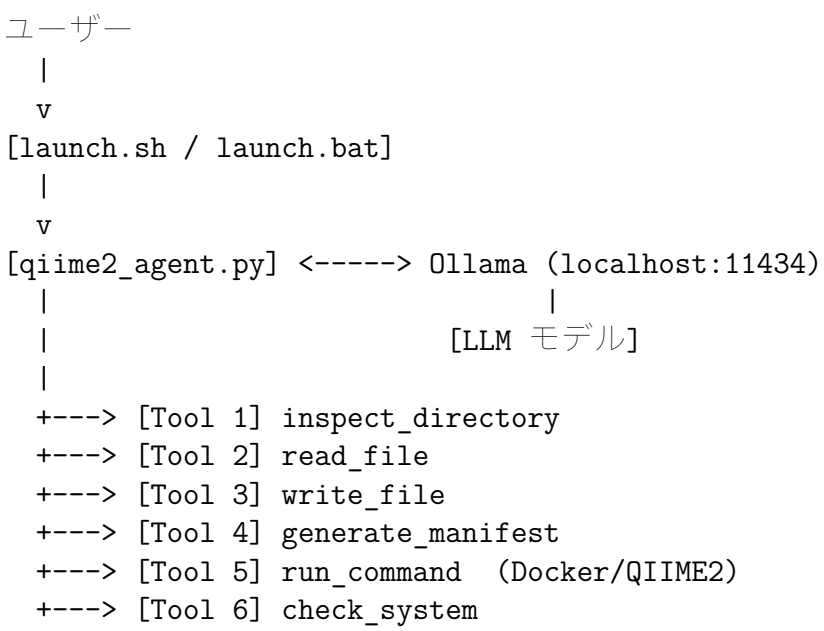

Figure 1: seq2pipe の全体アーキテクチャ

- 全 8 ステップの QIIME2 コマンド（インポート〜差次解析）
- 増幅領域別推奨パラメータ（V1–V3, V3–V4, V4）
- Docker 実行テンプレート
- メタデータファイル形式仕様
- SILVA 138 分類階層の説明
- 一般的なエラーとトラブルシューティング

この手法により、汎用 LLM モデルが QIIME2 専門家として振る舞うことを可能にした。

4.2 ツール定義と関数呼び出し

Ollama の Function Calling 機能を活用し、6 つのツールを JSON Schema 形式で定義した。各ツールは特定の役割を担う（表 1）。

4.3 エージェントループの実装

`run_agent_loop()` 関数が LLM の応答を受け取り、ツール呼び出しが含まれる場合は順次実行して結果を会話履歴に追加するループを制御する。このループは `tool_calls` フィールドが空になるまで継続し、最終的にテキスト応答を返す時点で終了する（リスト 1）。

Listing 1: エージェントループの中核部分

```
def run_agent_loop(messages: list, model: str):
    while True:
```

Table 1: seq2pipe が提供するツール一覧

ツール名	機能
inspect_directory	指定ディレクトリのファイル一覧・サイズを取得し、FASTQ/QZA/メタデータを自動判定する
read_file	テキストファイル（TSV, CSV, Markdown 等）の内容を最大 50 行読み込む
check_system	Docker・Ollama・Python のインストール状況とバージョンを確認する
write_file	解析スクリプト・マニフェスト・README 等をファイルに書き出す
generate_manifest	FASTQ ディレクトリから QIIME2 マニフェスト TSV を自動生成する
run_command	ユーザー確認後にシェルコマンドを実行する

```
        response = call_ollama(messages, model, tools=TOOLS)
        assistant_msg = {"role": "assistant",
                         "content": response["content"]}

        if response["tool_calls"]:
            tool_results = []
            for tc in response["tool_calls"]:
                fn = tc.get("function", {})
                result = dispatch_tool(fn["name"],
                                       fn.get("arguments", {}))
                tool_results.append({"role": "tool",
                                     "content": result})
            assistant_msg["tool_calls"] = response["tool_calls"
]
            messages.append(assistant_msg)
            messages.extend(tool_results)
            continue    # 次のループで再度 LLM に問い合わせ
        else:
            messages.append(assistant_msg)
            break       # テキスト応答のみ -> ループ終了
```

4.4 Ollama API との通信

call_ollama() 関数は Ollama の /api/chat エンドポイントに JSON リクエストを送信し、ストリーミングレスポンスを行単位で受信する。Python の urllib.request モジュールのみを使用しており、requests 等の外部パッケージに依存しない。

4.5 クロスプラットフォーム対応

macOS・Linux・Windows の差異を吸収するために以下の対策を実装した。

- **Docker 検出**: macOS では Docker Desktop の固定パス (`/Applications/Docker.app/Contents`) を優先し、その他の OS では `shutil.which("docker")` を使用する（`_get_docker_cmd()`）。
- **Windows ANSI カラー**: `os.system("")` を呼ぶことで Windows 10 以降における ANSI エスケープシーケンスを有効化する。
- **起動スクリプトの分離**: macOS/Linux 用（Bash シェルスクリプト）と Windows 用（PowerShell + バッチファイル）を別々に提供する。

4.6 マニフェスト自動生成

`tool_generate_manifest()` は FASTQ ファイルの命名規則（`_R1_` / `_R2_` パターン）を正規表現で解析し、ペアエンド・シングルエンド双方の QIIME2 マニフェスト TSV を自動生成する。Docker コンテナ内パス（デフォルト: `/data/output`）への変換も自動で行う。

5 解析ワークフロー

seq2pipe が生成する QIIME2 パイプラインは表 2 の 8 ステップで構成される。

Table 2: 生成される QIIME2 解析パイプラインの概要

STEP	処理	主要コマンド
1	データインポート	`qiime tools import`
2	クオリティ確認	`qiime demux summarize`
3	デノイジング（ASV 生成）	`qiime dada2 denoise-paired`
4	フィーチャーテーブル確認	`qiime feature-table summarize`
5	系統樹構築	`qiime phylogeny align-to-tree-mafft-fasttree`
6	分類学的解析	`qiime feature-classifier classify-sklearn`
7	多様性解析	`qiime diversity core-metrics-phylogenetic`
8	差次解析（オプション）	`qiime composition ancombc`

SILVA 138 参照データベース [5] を用いた Naive Bayes 分類器により、16S rRNA 増幅領域の分類学的同定を行う。増幅領域（V1–V3: 27F/338R, V3–V4: 341F/806R, V4: 515F/806R）に応じてプライマートリム長・トランケーション長が自動的に調整される。

6 対応モデルと性能比較

seq2pipe は Ollama で動作する任意の LLM と組み合わせて使用できる。表 3 に推奨モデルとその特性を示す。

Table 3: 対応モデル一覧

モデル	必要 RAM	モデルサイズ	特徴
`qwen2.5-coder:7b`	8 GB 以上	約 4.7 GB	コード生成特化（推奨）
`qwen2.5-coder:3b`	4 GB 以上	約 1.9 GB	軽量・高速
`llama3.2:3b`	4 GB 以上	約 2.0 GB	汎用・対話能力高め
`qwen3:8b`	16 GB 以上	約 5.2 GB	最高品質・推論能力高い

7 考察と今後の課題

7.1 達成された目標

seq2pipe は以下の目標を達成した。

- Python 標準ライブラリのみを用いたゼロ依存実装
- ローカル LLM による完全オフライン動作
- 3 OS（macOS / Linux / Windows）への完全対応
- FASTQ データ構造の自動認識と適応的パイプライン生成
- QIIME2 コマンドを Docker 経由で安全に実行するコマンド確認機構

7.2 現在の制限

- LLM の生成するコマンドの正確性はモデルに依存し、誤ったパラメータが生成される可能性がある。
- 大規模データセット（100 サンプル以上）に対するパフォーマンスは未検証である。
- SILVA 138 参照ファイルのダウンロードには約 30 GB のディスク容量と数時間を要する。

7.3 今後の課題

- 生成コマンドの自動検証機構の実装
- Web UI の提供（Gradio や Streamlit との統合）
- ITS / 18S rRNA など他のマーカー遺伝子への対応
- パイプライン実行結果のフィードバックループ実装

8 おわりに

本稿では、ローカル LLM と QIIME2 を統合した自動解析エージェント seq2pipe の設計・実装を報告した。Python 標準ライブラリのみによるゼロ依存設計と、クロスプラットフォーム対応により、インターネット接続不要の環境でもマイクロバイオーム解析を自動化できる実用的なツールを実現した。

seq2pipe はオープンソース(MIT ライセンス)として公開されており、https://github.com/Rhizo
からアクセスできる。

8 References

[1] Bolyen, E., et al. (2019). *Reproducible, interactive, scalable and extensible microbiome data science using QIIME 2*. Nature Biotechnology, 37, 852–857.

[2] Brown, T. B., et al. (2020). *Language Models are Few-Shot Learners*. Advances in Neural Information Processing Systems, 33, 1877–1901.

[3] Yao, S., et al. (2023). *ReAct: Synergizing Reasoning and Acting in Language Models*. International Conference on Learning Representations (ICLR 2023).

[4] Ollama (2023). *Ollama: Get up and running with large language models locally*. https://ollama.com/

[5] Quast, C., et al. (2013). *The SILVA ribosomal RNA gene database project: improved data processing and web-based tools*. Nucleic Acids Research, 41(D1), D590–D596.

[6] Callahan, B. J., et al. (2016). *DADA2: High-resolution sample inference from Illumina amplicon data*. Nature Methods, 13(7), 581–583.