mdx MaaS: 学術クラウド基盤 mdx におけるスケーラブルな AI 推論基盤

鈴村 豊太郎 [1),2)], 福田 敦史 [3)], 神沢 雄大 [3)], 金刺 宏樹 [2)]

1) 東京大学 大学院 情報理工学系研究科
2) 東京大学 情報基盤センター
3) Independent Researcher

suzumura@ds.itc.u-tokyo.ac.jp

mdx MaaS: A Scalable AI Inference Platform on the Academic Cloud System mdx

Toyotaro Suzumura[1),2)], Atsushi Fukuda[3)], Yuta Kanzawa[3)], Hiroki Kanezashi[2)]

1) Graduate School of Information Science and Technology, The University of Tokyo
2) Information Technology Center, The University of Tokyo
3) Independent Researcher

概要

AI 技術の普及に伴い、研究・教育への活用需要が高まっている。一方で、商用 AI サービスの利用に伴うセキュリティリスクや、オープンソースモデルを個人で運用するための計算資源確保など、実運用上のハードルは高い。本稿では、学術機関で AI モデル推論を安全かつ効率的に活用するための推論基盤 mdx MaaS を提案する。データ活用社会創成プラットフォーム mdx I に推論ランタイムと OpenAI 互換 API エンドポイント、チャット UI、動的スケーリングを統合し、安全性と運用効率を両立する推論基盤を提供する。mdx MaaS を構成するソフトウェア、提供するサービスと AI モデル、および実運用に向けた予備検証実験結果について報告する。

1 はじめに

AI 技術の急速な発展により、学術研究においてもその活用が不可欠となっている。特に、OpenAI や Gemini など企業が提供する商用モデルと、オープンソースモデル間の性能差が縮小し、多様な AI モデルを柔軟に利用できる環境が整いつつある。近年ではモデル学習よりも学習済みモデルの推論利用が増加しており、高性能なモデル推論環境をいかに安定的に利用できるかが重要となっている。また、大学・研究機関においても、分野を問わず AI モデルの活用が進んでいる。学術利用における文章作成・プログラム実装から、大規模なテキスト・画像等の研究データの変換・解析まで、その応用範囲は広がりつつある。

しかし、従来の方法で AI モデルを研究利用するには限界がある。商用のパブリック AI サービスは手軽に利用できる一方で、機密性の高い研究データを外部に送信するセキュリティリスクやコスト増大等の問題がある。セキュリティの問題を回避するためにオープンソースモデルを独自に運用する方法も行われているが、高性能推論に必要な計算資源の確保・維持管理コストに加え、構築・運用ノウハウの属人化が問題となる。

本稿では、この問題に対処し、学術機関で AI モデルを安全かつ効率的に活用するための推論基盤として、「mdx MaaS (Model as a Service)」を提案する。データ活用社会創成プラットフォーム mdx I [1] 上に、AI モデル推論、需要に応じた計算資源の動的スケーリングを含めた性能最適化機構、API・チャット UI を統合した基盤を提供することで、セキュリティ・性能最適化・効率化の両立を目指す。以下、mdx MaaS のアーキテクチャ設計と実装、さらに各種 AI モデルを実際に配備して行った予備的な性能検証の結果について報告する。

2 現状の AI モデル活用の課題

AI モデルを利用する主な手段として、民間企業が運営する商用 AI サービスを利用する方法と、各自でローカル LLM を入手してデプロイする方法が考えられる。しかし、AI モデルを研究プロジェクト等で利用するには、下記に示すような課題が残っている。

商用 AI サービスを利用する場合、研究データなど

を API へ送信する必要があるため、利用が認められない場合がある。さらに、API などの利用料金が入出力データ量 (トークン数) など利用量に比例して増大するため、AI エージェント等による呼び出しが大量に発生した場合、利用コストが意図せず急増する恐れがある。さらに、サービス側の一方的な仕様・ポリシー変更や提供停止、あるいはネットワーク障害等に起因する可用性のリスクもあり、ユーザ側から対処することができない。以上のように、商用 AI サービスの学術利用は、セキュリティ・コスト・可用性のいずれにおいても課題が存在する。

上記のコストやリスクを回避する手段として、個人・研究室で確保した計算機環境上でオープンソースの AI モデルをデプロイし、運用する方法も考えられる。しかし、商用 AI サービスに匹敵する高性能なモデルを扱うにはマルチノード・GPU といった相応のハードウェア資源が前提となり、ハードウェアの購入費・利用費などコストがかかる。その結果、個人や小規模研究室が恒常的に高い品質の推論環境を維持することは、費用面・管理面の双方で負担が大きい。

また、実務上は短時間・小規模な試行においては有効である一方、負荷が突発的に増加する場面や、長時間の連続実行が必要な場面でも効率的に安定して運用できるようにするには、モデルの量子化やキャッシュによる最適化などソフトウェア・ハードウェア双方の知識・ノウハウが求められる。個人で構築した環境を第三者が導入しようとしても、そのノウハウが属人化し、最適化された AI モデル運用環境を教育・研究機関内外に展開することが困難である。

以上のように、外部サービス依存はセキュリティ・コスト・可用性の不確実性を内包し、各自運用は性能・継続性が課題となる。これらの課題を解決し、持続的に AI モデルを活用可能な形へと収束させるためには、学術利用に適した新たな基盤が必要である。

3 mdx MaaS の概要

前述した課題に対処し、学術活動において持続的に AI モデルを活用可能な形へと収束させるために、以下の 3 つの要件を提示する。

(R1) **セキュリティ・データ管理** AI モデルに API・プロンプト等で送信したデータを許可されたユーザ以外アクセスできないようにすることで、機密情報等を処理できる。

(R2) **多様なモデルのサポートと UI の提供** LLM、VLM、画像認識、音声認識等の多様な最新 AI モデルを、エンドユーザはデプロイ不要で、API エンドポイントやチャット UI から即時に利用できる。

(R3) **性能・コスト・可用性の最適化** 限られた計算資源 (CPU/GPU) の中で、需要変動下でもユーザ体験（レイテンシ等）と運用効率（スループット、資源コスト）を最適化する。計算資源を CPU・GPU の理論性能・実効性能の特性およびユーザの利用傾向に基づき、動的な資源の確保と解放、並列数の調整などパフォーマンス最適化を自動的に行うことで、運用のスケーラビリティを向上させる。

これらの要件を満たす基盤として、mdx MaaS (Model as a Service) を提案する。mdx MaaS は mdx I 上で稼働する MaaS 基盤として、計算資源のスケーリングなどの性能最適化や API エンドポイント、チャットアプリなどの機能が事前構築された AI モデル推論を提供する共通基盤である。

mdx MaaS を利用する手段として、すでに mdx 上に構築されたサービスを使用するか、mdx の計算資源を学術プロジェクトで保有しているユーザが自分で構築して運用する 2 種類の形態を用意している。前者では、登録したユーザに AI モデルのエンドポイント URL と API キーが発行され、OpenAI API と同様にモデルにアクセス可能である。加えて、ChatGPT と同様のチャットベースの Web UI も提供され、ログイン後に商用 AI サービスと同様の使い勝手で利用可能である。一方、後者のように mdx の計算資源を保有している場合は、mdx MaaS を手順に従って構築・運用することもでき、他の AI モデルを独自にデプロイして運用したり、API とチャットサービスを他ユーザに共有したりすることも可能である。さらに、mdx が持つ IP アドレス・ポート単位でのアクセス制限機能と、mdx MaaS が持つアカウント管理機能を利用することにより、特定のホスト・ユーザのみにアクセスを制限することも可能である。

3.1 mdx MaaS の構成

3.1.1 mdx の概要

mdx[1] は、データ解析・データ科学的手法への期待が高い研究分野において、計算システムを柔軟に構築・運用できるサービスを提供するクラウド型システムである。mdx のシステムは東京大学柏キャンパスで稼働している mdx I と 大阪大学吹田キャンパスで

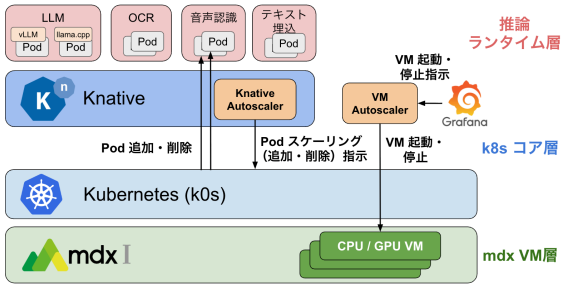

図 1　mdx MaaS ソフトウェアアーキテクチャ

稼働している mdx II がある。本稿の mdx MaaS は前者の mdx I 上で構築しており、 ”mdx” は mdx I を指す。

mdx の計算機環境は、汎用 CPU ノード群 (Intel Xeon Platinum 8368 搭載、全 368 ノード) と GPU 演算加速ノード群 (NVIDIA A100 Tensor Core GPU 8 基搭載、全 40 ノード) から構成され、VMware vSphere による仮想化環境を実現している [1]。 ストレージは、Lustre ファイルシステムによる高速内部・大容量内部ストレージ、Amazon AWS S3 互換の共有オブジェクトストレージを備える。

mdx は、利用者が申請を行うことで「プロジェクト」として必要な計算資源、ストレージ資源、ネットワーク構成を割り当てる IaaS を提供する。ユーザは割り当てられた資源内で CPU ノードや GPU ノードを自由に組み合わせて仮想マシンを構築し、用途に応じてソフトウェア環境を自由に設定することが可能である。

3.1.2　ソフトウェアアーキテクチャ

mdx MaaS 上の AI モデルは Kubernetes (以下、k8s) の Pod 上で稼働しており、ユーザの需要に応じて CPU・GPU などの計算資源を動的に確保・配分し、全体の計算資源・コストの無駄を抑える。確保された計算資源の中で、全体で負荷分散し、効率的に運用する。mdx MaaS のソフトウェアアーキテクチャ構成を図 1 に示す。

mdx VM 層は、mdx 上の CPU / GPU ノード群を仮想マシン (VM) として確保し、mdx REST API からの起動・停止・増減を行う基盤レイヤとして機能する。VM は計算資源プールとして管理され、需要に応じて弾力的に供給される。

k8s コア層は、VM 上に構築した k8s の軽量なディストリビューション (k0s) と Knative から構成される。Knative の機能である Knative Autoscaler[2] は、需要に応じて Pod を自動的に起動・停止・再配置させ、効率的に計算資源を利用する。本層にはシステム全体のリソース利用状況や負荷状態をモニタリングする Grafana [3] が配置されており、可視化された情報を提供する。この Grafana のモニタリング情報を基に、mdx 環境上の VM を自動的に起動・停止させるコンポーネント VM Autoscaler を構築中であり、Pod のスケーリングと併せ、より伸縮性の高い制御を実現する見込みである。

推論ランタイム層は、vLLM[4]、llama.cpp などの AI モデル推論ランタイムと Web サーバを Pod として稼働させ、ユーザから受け取ったデータからモデル推論を実行し結果を返す。

3.1.3　計算資源の管理と最適化

mdx MaaS では、限られた計算資源の中で最大限のサービスを提供するために、以下の要素を組み合わせた計算資源管理システムを実現する。需要変動下でも利用体験 (レイテンシ等) と運用効率 (同時接続時スループット、単位利用コスト) のバランスを保ち、R3 (性能・コスト・可用性の最適化) の要件を実現する。

k8s/Knative と mdx REST API による動的スケーリングと資源管理: 限られた mdx 上の計算資源内で、第一段階 (Knative Autoscaler による Pod スケーリング) → 第二段階 (VM Autoscaler による VM スケーリング) の二段階で制御する。 第一段階では、リクエスト急増時に必要なだけ k8s Pod を自動追加し、需要減少時には余剰となった Pod を停止させることで待機状態に移行する。Pod スケーリングは同時リクエスト数の変動によって動作し、短時間での負荷変動に対応する。 第二段階では、計算資源が逼迫した場合に mdx REST API を使用して VM を自動起動し、不要になった VM は自動停止してコスト効率を最適化する。今後、需要に応じて動的スケーリングを実現するためのより効率的なスケジューリングのポリシーを設計・実装していく必要がある。

LLM ランタイム: LLM 推論を稼働するランタイム側の最適化では、FlashAttention や Continuous Batching など GPU 向けの最適化機構が実装された vLLM、省メモリで CPU 向けにビルドされた軽量なランタイム llama.cpp を採用し、目的・用途・モデルに応じて適切なランタイムを選択できるようにする。

3.1.4　提供サービスと認証

mdx MaaS では、モデル推論としてエンドユーザ向けに OpenAI 互換 API と対話型チャットサービスを提供する。API サービスではエンドポイント URL を提供し、利用者は個別に発行された API キーで呼び出

せるようにする。後述する LLM 以外の OCR やテキスト埋込モデル、マルチモーダルモデルに関しても、バイナリは base64 で送受信し、統一 API で扱えるようにする。また、チャットサービスはオープンソースの Open WebUI で実装されており、ChatGPT と同様の直感的なインタフェースを用いて、デプロイされた LLM を対話的に呼び出すことが可能である。具体的な使用例は 3.2 で説明する。

認証機能として、許可されたユーザだけがチャットサービスを利用できるように、Shibboleth で実装された SAML 認証によるユーザ認証とシングルサインオンをサポートしている。SAML 認証によるチャットサービスのログインの手順を図 2 に示す。チャットサービスにログインしようとすると、学認 (学術認証フェデレーション)[6] が提供するディスカバリーサービス (DS) にリダイレクトされ、指定された学認 ID プロバイダ (IdP: Identity Provider) にて認証を行う。認証が成功すると、ユーザ情報 (SAML アサーション) が IdP から SP (Service Provider) に送信され、SP であるチャットサービスに登録されたアカウント情報と照合の上、ログインが完了する。

さらに、mdx ネットワーク内にある SP を外部に露出するセキュリティリスクを軽減し、利用状況のモニタリングを容易にするため、リバースプロキシサーバを導入している。リバースプロキシサーバはユーザと IdP、SP とを仲介し、外部からの mdx へのアクセスをリバースプロキシサーバに限定できる。現在ユーザ認証・ログインの動作確認として、学認のテストフェデレーションを用いた独自の IdP を使用しているが、実際に大学等でテスト・運用する際には、運用フェデレーションに移行し、正式な学認 IdP を利用する。

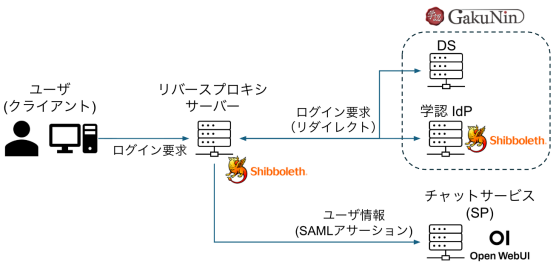

図 2　SAML 認証によるチャットサービスログインのフロー

3.1.5　提供モデル

幅広い学術分野への利用を想定し、mdx MaaS では LLM の他に OCR (光学文字認識)、音声認識、テキスト埋込等の最新モデルを提供する。表 1 は、mdx MaaS でデプロイしたモデルの一覧であり、今後の AI の動向を踏まえ、サポート対象モデルを随時追加・更新していく。現状はシングル GPU で稼働可能な規模のモデルを対象とする。

LLM として、日本語利用に適した多言語対応汎用モデル LLM-jp に加え、日本語テキストデータで微調整し推論性能を重視した DeepSeek 等をサポートする。OCR については、サンスクリット語を解析する人文学系の研究室へのヒアリングを踏まえ、サンスクリットの文章画像からデーヴァナーガリー文字を抽出する vedic モデルを導入する。

音声認識では、数十分に及ぶ長時間会話の文字起こしにも対応する、OpenAI Whisper モデルを軽量化した日本語・多言語音声認識モデルを導入している。さらに、近年では学術論文など大量のテキストデータを固定長ベクトル化し、RAG や kNN による類似文書検索・クラスタリング等に活用できるよう、汎用かつ多言語対応の bge-m3 テキスト埋込モデルをサポートする。今後、公開されているモデルの改良・更新に追従しながら、常に最新のモデルをサポートする予定であり、画像モデル (VLM)、マルチモーダルモデル (ViT, BLIP, CLIP) などの導入・検証も進める。

3.2　利用イメージ

ここでは、mdx MaaS のエンドユーザが LLM API とチャット UI を利用する具体的な方法について説明する。ユーザは、3.1.4 で述べた認証方法によってログインし、デプロイ済みモデルを選択して API キーを発行するか、チャットサービスを利用できる。モデルを API 経由で利用する場合、ユーザごとに API キーが発行され、個人のローカル環境や研究室サーバ等から、機密性の高い研究データを学内環境に留めたまま商用 API と同様の操作性でモデルを呼び出せる。

また、チャットサービスにログインすることで、図 3 のように ChatGPT と同様の直感的な対話インターフェースで LLM を利用できる。画面左上でモデル (LLM-jp 等) を選択し、下部のプロンプト欄から自然言語で質問や指示を送信すると、低レイテンシで応答を得られる。必要に応じて最大トークン数や温度などのパラメータを画面右側で動的に調整でき、用途に応じた推論設定が可能である。

上記の利用方法は mdx MaaS 単体で完結するが、研究データ管理基盤と連携することで、R1（セキュリティ・データ管理）の要件を満たしつつデータ利活用の導線を一体化できる。図 4 は、研究データ管理基盤 GakuNin RDM [7] の解析基盤（JupyterHub）か

表 1　mdx MaaS でデプロイしたモデル

カテゴリ	モデル名	特徴
LLM	llm-jp-3.1-13b-instruct4 (llm-jp)	日本語の対話生成に最適化
LLM	gpt-oss-20b (gpt-oss)	OpenAI API の o3-mini に相当 [5]
LLM	DeepSeek-R1-14b-Japanese (DeepSeek)	長文生成と推論能力を強化
LLM	Qwen2.5-3b-instruct (Qwen2.5)	多言語対応、最大 128k トークンのコンテキスト長
OCR	trocr-large-printed-vedic (vedic)	汎用 OCR モデルをデーヴァナーガリー文字用に微調整
音声認識	faster-whisper-large-v3 (faster-whisper)	OpenAI の Whisper-Large-v3 モデルを軽量化
音声認識	whisper-large-v3-ja (whisper-ja)	軽量化の上、さらに日本語データで微調整
テキスト埋込	bge-m3	多機能性、多言語性、多粒度性を保持

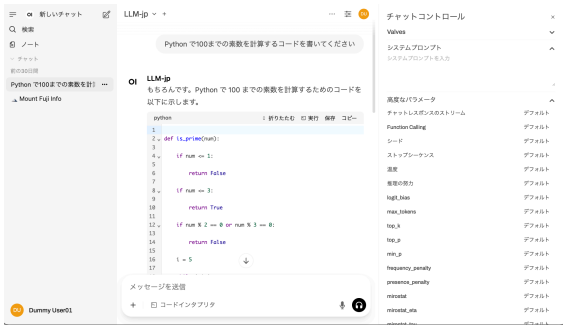

図 3　mdx MaaS チャットインターフェース

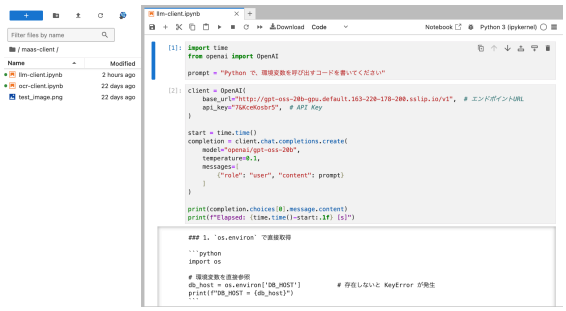

図 4　GakuNin RDM 解析基盤より gpt-oss-20b モデルの使用

ら mdx MaaS の LLM（gpt-oss）を API で呼び出す例である。GakuNin RDM は、文部科学省の事業 [8] のもとで運用される学術向けデータ基盤であり、データは管理基盤側に留めたまま解析ノートブックから推論を呼び出せるため、機微情報を外部に出さずに処理できる。これは mdx MaaS の API/認証機構をそのまま活用した一実装例であり、他の学内システムやストレージサービスとも同様に接続可能である。

3.3　予備検証

本節では、mdx MaaS の提案した 3 つの要件のうち、R2 (多様なモデルのサポート)、R3 (推論実行性能) の検証実験について説明する。具体的には、現在構築中の mdx MaaS 上で実際にデプロイし動作確認を行ったモデル群のデプロイおよび API 呼び出しの動作確認と、LLM モデルの gpt-oss-20b を用いた論文要約タスクと単問応答タスクについて応答時間 (レイテンシ) とスループット評価を行った。

3.3.1　mdx MaaS で検証したモデル

ここでは、研究での利用を想定し、表 1 にデプロイした AI モデルを API 呼び出しにより推論を行えるか検証する。LLM については、GPU 版は vLLM、CPU 版は llama.cpp ランタイムで CPU コア数を調整して動作検証を行った。Hugging Face で公開されているモデルの中から、モデル読み込みが高速で汎用性の高い Safetensors と、小さい計算資源でも推論できるよう軽量化された GPT-Generated Unified Format (GGUF) ファイル形式のものを採用しデプロイした。GPU VM では、モデルの読み込みが高速でかつ vLLM ランタイムに最適化されている Safetensors フォーマットのモデルを、CPU VM 上では軽量な GGUF フォーマットのモデルを使用した。

OCR モデルについては、サンスクリット語の印刷物画像を vedic モデルで文字認識させた後、さらに DeepSeek モデルでサンスクリット語から日本語に翻訳、出力テキストの整合性を目視で検証した。音声認識モデルでは、数十秒〜数十分の英語・日本語の会話音声データを用い、文字起こし結果の基本的妥当性を確認した。テキスト埋込モデルについては、論文のタイトル・アブストラクトから変換した埋め込みベクトルに kNN 類似探索の試作を行い、トピックが類似した論文の比較を行った。

3.3.2　実行性能検証

mdx MaaS の要件の一つである性能・コスト・可用性に関して、モデル推論の現実的な実行性能を評価するために (i) 大量研究データを処理するバッチ実行におけるスループットの限界と、(ii) チャットサービスなどインタラクティブなやり取りに対し、ユーザが許容できる応答時間 (レイテンシ) を維持したまま処理可能な同時リクエスト数について評価した。具体的に

は、(i) 英語論文の日本語要約と (ii) 単問応答 (入出力計約 1,000 tokens) タスクにおいて、mdx の GPU、CPU VM にデプロイした LLM に対し、単体および同時並列 (最大 64) でリクエストを送信してその応答時間・スループット (tokens/s) を計測した。入力データとして、(i) 論文要約タスクでは Transformer の論文 [9] テキストを抽出したもの (計 6,136 tokens) を使用し (ii) 単問応答タスクでは 20 種類の質問文を生成したものを使用する。LLM モデルとして、(i) では gpt-oss-20b を、(ii) では gpt-oss-20b に加えて Qwen2.5-3b の 2 種類を使用した。

実験環境として、mdx I の GPU VM (NVIDIA A100 40GB GPU 1 基), CPU VM (Intel Xeon 8368 Platinum 38 コア 2.4GHz x2 ソケット) 各 1 台を使用した。なお CPU VM では、CPU のハイパースレッド (最大 152) を、各 k8s Pod あたり指定したコア数 (16, 32, 64) ごとに割り当てた。なお、リクエストの規模に応じて Pod および VM を動的に起動または停止する動的スケーリング機能に関してはまだ実装途中であるため、現時点では 1 VM に 1 つまたは複数の Pod を事前に起動した上で評価を行っており、動的スケーリングも含めた評価と最適化については今後の課題とする。

(i) gpt-oss-20b モデルによる論文要約タスクの結果を表 2 に示す。GPU VM では、リクエストを同時に 64 並列まで上げることで、スループットを 11,821 tokens/s まで上げることができた一方、CPU VM では最もスループットが上がる条件 (1 Pod あたり 16 コア x 8 並列) でも 438 tokens/s に留まった。なお、GPU、CPU ともに、表 2 に記載したコア数、並列リクエスト数以上での実行は VM 内の GPU、CPU 使用率がほぼ 100% となり、スループットもこれ以上、上がらなかったため、評価の対象外とした。

バッチ処理のタスクでは、GPU VM を用いてリクエストを同時に (最大 64 並列) 送ることでスループットを上げることができ、計算資源を最大限活用できる。なお、GPU、CPU での最大スループットに約 27 倍の差 (11,821 vs 438 tokens/s) があるが、これは元々 GPU・CPU の理論性能値も約 27 倍 (312 vs 11.7 TFLOPS) あるためであり、CPU を用いる場合でも、VM 数を増やすことでバッチ処理での利用も可能である。

(ii) 単問応答タスクの結果について、gpt-oss-20b、Qwen2.5-3b モデルでの応答時間とスループット (TP: tokens/s) を表 3 に示す。比較のため OpenAI API (o3-mini) で同様のリクエストを呼び出した場合の結果も一番下に掲載する。

表 2　論文要約タスクのスループット

環境	並列数	スループット
GPU	1	1116
	4	2869
	16	5609
	64	11821
CPU (16 コア)	8	438

gpt-oss-20b モデルの場合、GPU VM では 1 並列で応答時間が 6.6 秒、4 並列でも 8.2 秒と、十分実用的な範囲に収まっている。並列リクエスト数を 16 に増やすと応答時間が 12 秒まで増えてしまう一方、スループットを 1 並列の 8 倍以上 (151 → 1,292 tokens/s) まで上げることができ、効率性を上げることはできる。より多くのリクエストを同時に受け付けることによりスループット・応答時間がともに増加するトレードオフがあるが、GPU VM では gpt-oss-20b 規模のモデルでも 4 並列、多少のレイテンシを許容する場合なら 16 並列のリクエストを受け付けることが現実的に可能である。なお、CPU VM では、64 コア、1 並列リクエストでも応答時間が 32 秒と OpenAI API の 3 倍以上かかっており、このタスクにおいて gpt-oss-20b モデルの利用は本条件では実用上困難である。

一方、より軽量な Qwen2.5-3b モデルにおいて、GPU VM では 64 並列でも応答時間を 10 秒以下に抑えたまま、スループットを 6618 tokens/s まで上げることができ、CPU VM においても、1 並列もしくは 64 コアで 2 並列までなら同等の応答時間で処理できている。このことから、ユーザからの大量のリクエストを同時に処理したい場合、多少応答の質が下がる可能性はあるが GPU で Qwen2.5-3b 規模のモデルを動かすことが適しており、CPU においても GPU より効率は良くないが、2 並列までなら現実的に可能である。

結論として、現状の実装では (i) 大量のデータをバッチ処理するには GPU による並列実行か複数 CPU ノードによる実行が適しており、 (ii) 低遅延が要求される対話系用途では GPU を用いて並列リクエスト数を調整するか、CPU と軽量な LLM を併用するなどの使い分けが現実的と言える。今後は、GPU 側はバッチ／並列設定の最適化、CPU 側は並列処理ワークロードのスループット向上を重点に評価と改良を進めていく。

表 3　単問応答タスクの応答時間 (秒) とスループット (TP)

		gpt-oss-20b		Qwen2.5-3b	
環境	並列数	時間	TP	時間	TP
GPU	1	6.6	151	2.3	433
	4	8.2	494	3.2	1238
	16	12.4	1292	5.9	2735
	64	18.6	3450	9.7	6618
CPU	1	42.3	24	8.5	118
(16 コア)	2	41.9	48	12.4	161
CPU	1	33.9	30	7.5	134
(32 コア)	2	36.0	56	11.9	168
CPU	1	32.0	31	6.8	146
(64 コア)	2	45.2	44	9.3	214
OpenAI API	1	9.6	104		

4　関連プロジェクト

近年、学術クラウドや各大学・研究機関が機関内の推論サービス提供に踏み出している一方、モデル学習基盤に比べ推論基盤のサービス化の事例はまだ少なく、各機関での制度・技術アーキテクチャの模索が続いている。本節では、学術機関が提供する推論基盤 (MaaS) の近年の具体的な事例について説明する。

まず米国においては、インディアナ大学で運用されている Jetstream2[10] (NSF 学術クラウド) の上で大規模 LLM をデプロイし、学内 SSO でアクセスできる Open WebUI 型のチャットと OpenAI 互換 API を提供している。スタンフォード大学では、LLM を Web インタフェース経由で利用できる学内限定サービス AI Playground と、学内クラウド上にデプロイされた LLM に OpenAI 互換 API でアクセス可能にする AI API Gateway を導入し、教育・実習の両面からの利用を促進している [11]。ドイツでは、ゲッティンゲン科学データ処理協会 (GWDG) が管理する GPU サーバから、全国の大学から無料で利用できる学術向けチャット UI[13] と OpenAI 互換の API を提供する GWDG Academic Cloud [12] が運用されている。

上で紹介した事例はいずれも、商用 API 依存を補完する大学・研究機関にてモデル推論基盤を整備し、API とチャット UI を認証付きで提供している。mdx MaaS はこの潮流に即しつつ、特に日本の学術機関におけるニーズを捉えた基盤を整備していく。

5　まとめと今後

本稿では、学術利用において安全かつ効率的に AI 推論を活用することを目的として、mdx 上に推論機能をパッケージ化して提供する「mdx MaaS」を提案した。提案にあたり、 (R1) セキュリティとデータ管理、(R2) 多様なモデルのサポートと UI の提供、 (R3) 性能・コスト・可用性の最適化、の 3 要件を定義し、それらを満たすアーキテクチャ (k8s/Knative と mdx VM の二段階動的スケーリング、GPU/CPU 両対応ランタイム、学認連携のチャット UI と API) を示した。さらに、LLM、OCR、音声認識、テキスト埋込の最新モデルを実機でデプロイし、実行性能検証を経て、大量データのバッチ処理やリアルタイムな対話型のリクエストでは GPU 中心の運用で十分性能が出ている一方、CPU においては軽量なモデルを使用することで対話型の利用も可能性を確認した。

mdx MaaS の提案により、学術機関、研究プロジェクト等での AI 推論活用における以下の成果が期待される。第一に、ユーザ認証・IP アドレス制限等によるアクセス管理により、機微情報を含む研究データを安全に処理できる環境を提供する。第二に、多様なモデルのサポートと API・チャット UI の提供により、研究者が最新の AI モデルを迅速に活用できる基盤を構築する。第三に、ユーザの需要変動に追随する動的スケーリングにより、コスト効率を保ちながら安定した推論サービスを提供する。

今後、mdx MaaS を実用可能なものにしていくために、以下のような改良と拡張を進めていく。

モデル拡充: まずモデルのサポートとして、LLM に限定しない最新のモデルを充実させる。音声認識モデルについては、長時間音声・話者識別・雑音耐性ケース等具体的なデータセットやアプリケーションを考慮した追加検証を行う。画像を含めたマルチモーダルモデルも段階的に導入し、画像データからテキスト生成等の指示を共通の UI と API にて提供する。テキスト埋込モデルは長文や多言語テキストへの実用性を高め、品質 (近傍検索の再現率等) を評価する。

動的スケーリングの実装と評価: VM と Pod の動的スケーリングの実装を完成させ、ユーザの利用状況に対して最適なスケーリング方法を設計、評価していく。Grafana によるシステム全体のリソースモニタリングと連携した VM Autoscaler を実装し、VM 自体の自動起動・停止を実現することで、Pod 単位のスケーリングも含めた二段階の動的スケーリングを備えたインフラ制御を完成させる。また、mdx VM には電源投入から稼働可能になるまで数分を要するという制約があるため、VM Autoscaler にユーザの需要予測モデルを組み込み、過去の利用トレンドから将来の負

荷を予測して、需要増加が見込まれる前に事前起動を行う仕組みを導入予定である。これにより、急激なリクエスト増加時でも十分な計算資源を先行的に確保でき、ユーザに対して安定かつ低遅延な推論サービスを提供できるようになる。さらに、負荷が低下した際には不要な VM を自動停止させることで、余剰リソースを削減し、コスト効率の高い運用を実現する。今後、数十台規模の定常運用での性能指標 (スループット、レイテンシ) を確立し、100 台規模のスケール試験でピーク時の事前起動の有効性検証、必要な計算資源の見積もりを行う。

本格展開に向けたユーザ検証と連携拡大: 学内でのユーザ検証 (授業・研究室・共同研究プロジェクト) を継続し、研究および教育への応用と他分野への横展開を高める。 ユースケース開拓として、3.1.5 で述べたサンスクリット語の OCR 解析と日本語解説生成の応用に加えて、物性科学分野の ARIM プロジェクトでは既存の実験試料の説明の品質チェック AI を mdx MaaS へ置換する実証を進め、その他の分野においても VLM、音声認識、テキスト埋込を組み合わせた実地検証を行う。 全国の学術コミュニティへの展開を見据え、学認連携・モデルのデプロイと最適化機構などをパッケージ化して、大学教職員・研究者の共通基盤としての実装要件を整理する。

mdx MaaS は、学術機関での AI 推論活用における外部依存・再現性・運用格差の課題を解決し、要件を同時に満たす統合基盤として機能する。今後も継続的な改良と拡張を通じて、研究・教育活動における AI 技術の安全で持続可能な活用を支える基盤として発展させていく。

謝辞

本研究の一部は、文部科学省「AI 等の活用を推進する研究データエコシステム構築事業」の支援を受けたものである。

参考文献

[1] Suzumura, Toyotaro, et al. "mdx: A cloud platform for supporting data science and cross-disciplinary research collaborations." IEEE CBDCom, 2022.

[2] Knative Autoscaling, https://knative.dev/docs/serving/autoscaling/

[3] Grafana: The open and composable observability platform, https://grafana.com/

[4] Kwon, Woosuk, et al. "Efficient memory management for large language model serving with pagedattention." Proceedings of the 29th symposium on operating systems principles. (2023)

[5] Agarwal, Sandhini, et al. "gpt-oss-120b & gpt-oss-20b model card." arXiv preprint, 2025.

[6] 学術認証フェデレーション 学認 GakuNin, https://www.gakunin.jp/

[7] GakuNin RDM (研究データ管理基盤) 国立情報学研究所 オープンサイエンス基盤研究センター, https://rcos.nii.ac.jp/service/rdm/

[8] AI 等の活用を推進する研究データエコシステム構築事業, https://www.nii.ac.jp/creded/nii_ac_jp_creded.html

[9] Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems (2017).

[10] Hancock, David Y., et al. "Jetstream2: Research clouds as a convergence accelerator." Computing in Science & Engineering 26.3, pp.9-19 (2024)

[11] Developers Can Now Try Out the AI API Gateway, https://uit.stanford.edu/news/developers-can-now-try-out-ai-api-gateway

[12] Academic Cloud, https://academiccloud.de/

[13] Chat AI Georg-August-Universität Göttingen, https://www.uni-goettingen.de/en/686446.html