参考資料９

Rev. 1.0.0.0019 (2025/5/26)

生成 AI 品質マネジメントガイドライン

第 1 版
(Revision 1.0.0)

2025 年 5 月 26 日

国立研究開発法人産業技術総合研究所

インテリジェントプラットフォーム研究部門
テクニカルレポート IPRI-TR-2025-01

サイバーフィジカルセキュリティ研究部門
テクニカルレポート CPSEC-TR-2025001

人工知能研究センター
テクニカルレポート

前書き（Foreword and Disclaimer）

本ガイドラインは、国立研究開発法人産業技術総合研究所（産総研・AIST）が企業・大学等の有識者委員と共に構成した「機械学習品質マネジメント検討委員会」においてとりまとめたものである。

本ガイドラインの内容に寄与した委員の意見は技術者としての個人の知見に基づくものであり、各々が所属する会社等の意見を代表するものではない。

本ガイドラインは、生成 AI を利用したシステム・サービスの開発を主導する企業等が、そのビジネス等への影響を踏まえて主体的にその採用の有無を選択し、共同開発者等と共に実践するものであって、法令・公的指針等との関係においては非拘束的なものである。本ガイドライン中で規範的 (normative) とされる規定は、あくまで本ガイドラインを任意に採用した場合に限り、規範的な意味を持つものである。

目次

1 本ガイドラインの狙い、取扱範囲、構成

本書は、生成 AI の品質マネジメントに関し、ガイドラインを示すものである。これまでにも、産総研は AI の品質マネジメントに関するガイドライン[AIQMv4]を発行してきた。これは機械学習技術により、用途に応じた学習データセットからその特性を学習させて構築する、識別ないし予測を行う機械学習モデルを核とした AI を対象としたものである。これに対し、本書は、大規模言語モデルや画像生成モデルなどの生成 AI[1]を対象としている。

1.1 狙い

生成 AI は、テキストや画像など自由度の高いデータを生成して出力することができる。また多くの場合、入力によって生成 AI が果たす機能を様々に変更することができる。産総研は、[AIQMv4]の Annex [AIQMv4Annex]において、「新想定と従来想定の違い」を以下の通り 2 つ示した。新想定が生成 AI に関するもの、従来想定が識別・予測 AI に関するものである。

1）基盤モデルに基づく AI： 従来想定では機械学習モデルの開発者は最終用途を理解しており、必要な品質目標を設定して品質マネジメントを行う。新想定では基盤モデル開発者は最終用途を決めることができず将来可能性のある幅広い用途向けに基盤モデルを訓練する。またサービス開発者や提供者は基盤モデル開発には直接関与せず、調整フェーズ以降でのみ品質マネジメントを行うことができる。

2）訓練データを再現しうる出力： 従来想定では出力させたい値はラベルや予測値で、あらかじめ取り得る値が分かっており、訓練データを再現する可能性はない。新想定における出力には訓練データと同等の表現力があり、訓練データを再現しうる。

本書では、これらの違いを含め、生成 AI で新しく生じた新想定への対処策を定めるための考え方を示す。

[1] 現時点で、生成 AI が用いるモデルと大規模言語モデル、基盤モデル、英語文献に現れる Frontier model, General purpose AI, Advanced AI などは多くの場合同じものを指している。本書では特に区別する必要がある場合はその旨説明する。それ以外の場合は文脈に応じた表現を用いる。

1.2 想定読者

本書が想定する読者は、生成 AI の開発、提供、利用、規制に関わる人々である。中でも特に、他社製の基盤モデルを用いたシステムの開発と提供に携わる人々である。最先端の基盤モデルを開発可能な事業者は世界でも限られており、多くの人は出来合いの基盤モデルを使う立場にあると考えられる。

また、品質マネジメントは開発組織内の内部監査によることがある。品質マネジメントガイドラインは、開発組織を越えて、第三者監査で利用しても良い。

1.3 取扱範囲

本書の主題は品質マネジメントである。これに対して、リスクマネジメント [2]がある。本書ではこの 2 つの違いを意識した上で、まずリスクマネジメントについて論じ、必要な場合に品質マネジメントについて補足する。

1.3.1 生成 AI のリスクの全体像

生成 AI の「セーフティ」(safety)に関して、世界各地で関心が高まり、2023 年 11 月にイギリスで初の AI Safety Summit が開催された。またこれを契機として、英国を筆頭に米国、日本、シンガポールなど各国で AI Safety Institute (AISI)が設立された。2024 年 5 月にはソウルで 2 回目の AI Safety Summit が開かれた。この時発表された[Bengio 2024]には 13 項目のリスクが示されている(表 1)。

このうち、「悪意ある利用のリスク」に属するものは本書では詳細には取り上げない。本書を含め、ガイドラインには強制力がなく、悪意を持つ人々に対しては効力を持たない。残りのリスク(小項目)はリスクの発生形態によって、2 つのカテゴリーに大別できる。一つは、単一の AI システムが引き起こすもので、表 1 の小項目のうち、**製品機能**、

[2] 大雑把に言うと、品質は与えられた要件の充足度である。リスクは分野によって大きく分けて 2 つの意味で使われる。金融ではリスクは不確かさ(volatility)であり、プラスにもマイナスにもなる。IT を含めた工学ではリスクはネガティブなもので、発生確率と深刻度の積で見積もられる。[NIST AI-600-1]では、「発生確率と結果の規模」という表現でプラスを含めている。品質は要求仕様に有用性を含むが、有用性の不足はリスクとしてはあまり考慮されない。また要求を満たすものが引き起こすネガティブなリスクは品質ではあまり考慮されない。

公平性、**プライバシー**、**著作権**が含まれる。もう一つは AI が普及することで広く社会に影響が及ぶものであり、**制御の喪失**、**労働市場**、**二極化**、**市場独占**、**環境**が含まれる。本書の今回の版では、前者を主に考慮する。これらの多くは従来のソフトウェアや、識別・予測 AI の品質を扱う際にも考慮されてきた事項である。また本書の想定読者にとって、生成 AI の製品に関わる上で対策が必須の事項と考えられる。一方、後者については、個別に取り上げることはせず、その可能性があることを踏まえたハイレベルな議論にとどめる。これらのリスクの中には、**労働市場**、**二極化**、**市場独占**のように、要求仕様を満たすシステムこそが引き起こすと考えられるリスクがある。これらについては社会的なガバナンスの一環として考慮されるものとし、本書ではこれ以上は扱わない。

表 1. [Bengio 2024]が挙げたリスクのリスト

リスク（大分類）	リスク（小分類）	本書での略称
悪意ある利用のリスク	虚偽コンテンツによる個人への危害	**個人向け虚偽コンテンツ**
	公共の意見の歪曲と操作	**公共の意見操作**
	サイバー攻撃	**サイバー攻撃**
	デュアルユース科学のリスク	**デュアルユース**
故障に起因するリスク	製品機能に起因するリスク	**製品機能**
	バイアスと代表性不足によるリスク	**公平性**
	制御の喪失	**制御の喪失**
システミックリスク	労働市場のリスク	**労働市場**
	グローバルな二極化の助長	**二極化**
	市場独占リスクと単一障害点	**市場独占**
	環境に対するリスク	**環境**
	プライバシーに対するリスク	**プライバシー**
	著作権侵害	**著作権**

1.3.2 与えられる要請とその実現手法

本書では、既に決まった要請があるときにそれをどう実現するかを論じる。社会的な合意がないことや、あるいは個別の状況に基づいて決める必要があり一概に決められないことについて、本書は意見を述べない。

要請はその由来に基づいて以下のように分類できる。それぞれに小分類を示し、その要請が対処を求めるリスクをかっこ内に添える。

提供者の狙いに由来する要請
　　有用性、安全性、セキュリティ（**製品機能**）
社会的合意に由来する要請
　　法的な要請（**プライバシー**、**著作権**）
　　倫理的要請（**公平性**）

「提供者の狙いに由来する要請」はシステムを社会に提供する提供者が、提供の狙いを満たすために必要な事項として定めるものである。有用性と安全性は共にこれに属するが、力点に違いがある （[AIQMv4] 4.2 節、4.3 節を参照）。またセキュリティは攻撃によって他の要請が充足されなくなるリスクへの対策を求める。これらはすべて、[Bengio 2024]のリストの中では**製品機能**に関わる。

一方、「社会的合意に由来する要請」は社会の大方の合意として社会が要求するものであって、法的拘束力を持つ場合もある。また、この要請は社会（国や文化）によって異なる。社会によっては一定の社会的要請を法律として明文化している。多くの国では**プライバシー**、**著作権**はこれにあたる。また**公平性**は対処すべきリスクとして多くのガイドラインで取り上げられているが、何を公平とみなすかについては様々な考え方があり、社会的に一律の合意がない([AIQMv4]4.4 節、10.1.2 節を参照)。これらについては、まず要請の内容を用途や利用シナリオにおいて明確にすることが重要である。本書では要請の内容がどうあるべきかについては意見を述べない。与えられた要請に対してどのような技術的取扱いが可能かを示す。

1.3.3　基盤モデルと基盤モデル利用システム

本書では、基盤モデルではなく、基盤モデルを利用するシステム（以下、「基盤モデル利用システム」と呼ぶ）を品質マネジメント対象とする。本書の想定読者について述べた通り、多くの人々は出来合いの基盤モデルを使う立場にある。出来合いの基盤モデルに対して施せる管理策は限られている。また、基盤モデルは汎用性が高い。用途により満たすべき品質の種類や程度は異なる。基盤モデルが果たしうる全ての用途に共通に必要な品質はごく僅かであり、それらのみを満たしても、個別の用途への適合性は保証

されない。基盤モデル利用システムは用途が定まっており、満たすべき品質の種類や程度を詳しく見積もることができる。

1.3.4　基盤モデル利用システムに対する立場

本書では、基盤モデル利用システムの開発者および提供者を主な対象としているが、開発者や提供者だけでは対処しきれないリスクがある。システムの悪用または誤用によるリスクの中には、利用者[3]に一定の責任や能力を求めないと対処できないものがある。基盤モデルを自ら作る開発者が実施しうる管理策の中には、出来合いの基盤モデルを利用する開発者には実施不可能なものがある。そこで、本書では、基盤モデル利用システムに対する立場によって、以下のように分けて管理策を述べる。

基盤モデルを利用する開発者および提供者が対処するべき事項
　　コンポーネント品質、システム品質
基盤モデル開発者が対処するべき事項
　　学習データ品質、モデル品質
基盤モデル利用システムの利用者が担うべき事項
　　悪用しない、過度に期待しない、依存しない

システムの開発者や提供者は、一般に基盤モデル開発者に対して品質マネジメントの実施を求める立場にない。そこで、品質に関わる情報を適切に開示している基盤モデルを優先して選択することが重要である。また逆に、システムの利用者に対してはシステムの正しい用途や機能の限界を伝達することが重要である。

1.4　従来の国内政府系ガイドラインとの関係

日本の省庁は 2010 年代後半から AI に関するガイドラインをいくつも発行しているが、その多くを統合する形で 2024 年に AI 事業者ガイドラインが発行された。また 2024 年初めに AI セーフティ・インスティテュートが発足し、9 月には複数の「ガイド」を

[3] AI 事業者ガイドライン[AIGBiz]では「AI 利用者」は事業として AI を利用する企業等であり、そのため AI の運用を担う場合がある。事業としてではなく AI を利用する主体は「事業外利用者」と呼んでいる。これに対し本書での「利用者」は他者が開発・運用する AI を利用し、AI の運用には関わらない事業者と、事業外利用者を指す。

発行した。その中でも AI セーフティに関する評価観点ガイドは幅広い意味でのセーフティに関わり AI 品質との関係が深い。これらの文書と本書の関係を図 1 に示す。

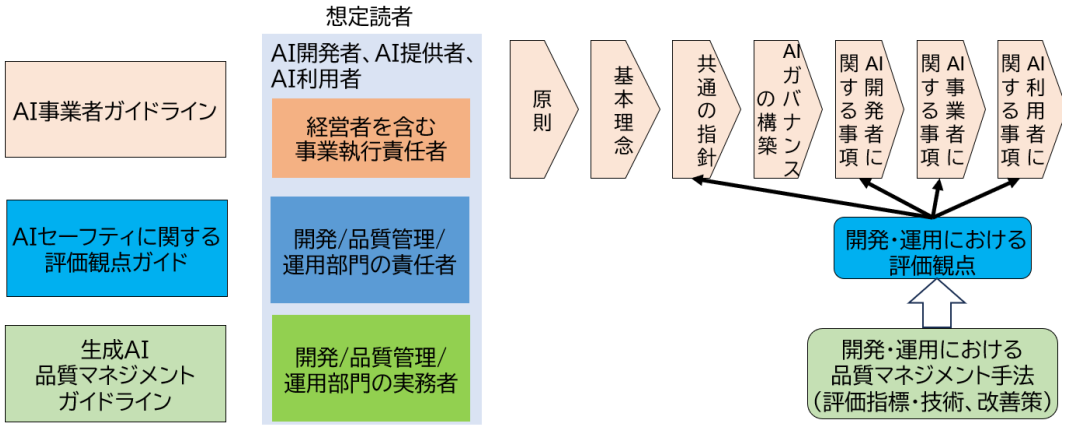

図 1　AI 事業者ガイドライン、AI セーフティに関する評価観点ガイド、本書の関係

AI 事業者ガイドラインは事業者の取組みの全体像を経営層に示す。評価観点ガイドは開発・運用における評価観点を開発/品質管理/運用部門の責任者に示す。生成 AIQM ガイドラインは実務者に評価観点に関し必要な水準を実現するための品質マネジメント手法を示す。

1.4.1　AI 事業者ガイドライン

経済産業省および総務省は 2024 年に AI 事業者ガイドライン[AIGBiz]を発行した。AI 事業者ガイドラインは AI ガバナンスの統一的な指針を示すものであり、事業として AI に関わる事業者が行うべき取組みを、経営層による方針策定や体制構築から事業部門による品質マネジメントまで総合的に示しているが、重点は体制やプロセスの整備にあり、技術的施策の多くは別添での例示となっている。これに対し、産総研がこれまでに発行した機械学習品質マネジメントガイドライン[AIQMv4]や、本書は、個別の AI システムの開発や運用を行う中で AI システムの品質を実現する技術的手法を示すものである。この際、AI 事業者ガイドラインが掲げる 3 つの基本理念と、それに基づく原則のうち、「各主体が取り組む事項」を前提として踏襲している。

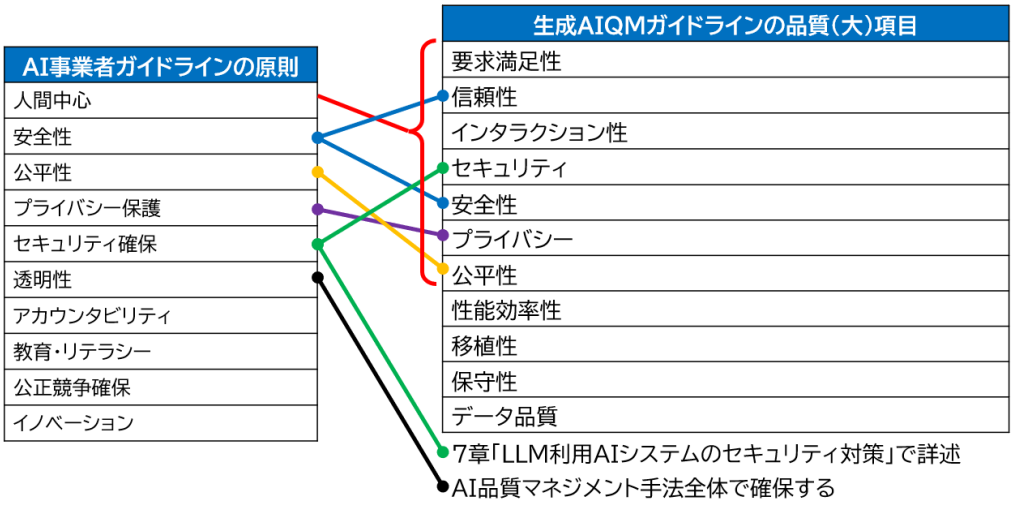

図 2 AI 事業者ガイドラインの原則と本書の品質項目の対応

AI 事業者ガイドラインが掲げる原則と、本書で扱う品質の対応を図 2 に示す。人間中心には AI の社会への影響と、AI の利用者や近辺にいる第三者への影響の両方を含むと考えられるが、本書では後者のみ扱う(1.3.2 節)。これには主に、要求満足性、信頼性、インタラクション性、セキュリティ、安全性、プライバシー、公平性により対応する。安全性原則には、品質としての信頼性と安全性で対応する。公平性原則には品質としての公平性で、プライバシー保護原則には品質としてのプライバシーで、それぞれ対応する。セキュリティ確保の原則にも上記の品質が関わるが、本書の第 7 章でまとめて詳述している。透明性原則には個別の品質ではなく、本書で示す AI 品質マネジメント手法全体で対応する。アカウンタビリティ、教育・リテラシー、公正競争確保、イノベーションの各原則は、事業者と社会のかかわりの中で対応するものであり、個別の AI の品質には対応しない。

1.4.2 AI セーフティに関するガイド群

AI セーフティ・インスティテュートは 2024 年に生成 AI を主な対象として、「AI セーフティに関する評価観点ガイド」[AISIeval]と「AI セーフティに関するレッドチーミング手法ガイド」[AISIrt]を発行した。これらのガイドと本書は役割が異なるが、対象や AI セーフティの取扱範囲に関して一致している。評価観点ガイドの役割は、AI セーフティを実現するために扱うべき評価観点を示すことにあり、レッドチーミング手法ガイドの役割は、対象 AI システムに施されたリスクへの対策を攻撃者の視点から評価するためのレッドチーミング手法を示すことにある。これに対し本書の役割は、対象システ

ムの品質実現に必要な対策を示すことにあり、品質を損なう攻撃に対する備えとしてセキュリティ対策を述べ、その一環としてレッドチーミングの必要性に言及している。AISI の両ガイドと本書は、LLM を構成要素とする AI システムを対象とする点、および、AI セーフティのうち、AI システムがそのシステムのエンドユーザーに直接及ぼしうる影響を取扱範囲とする点で共通している。

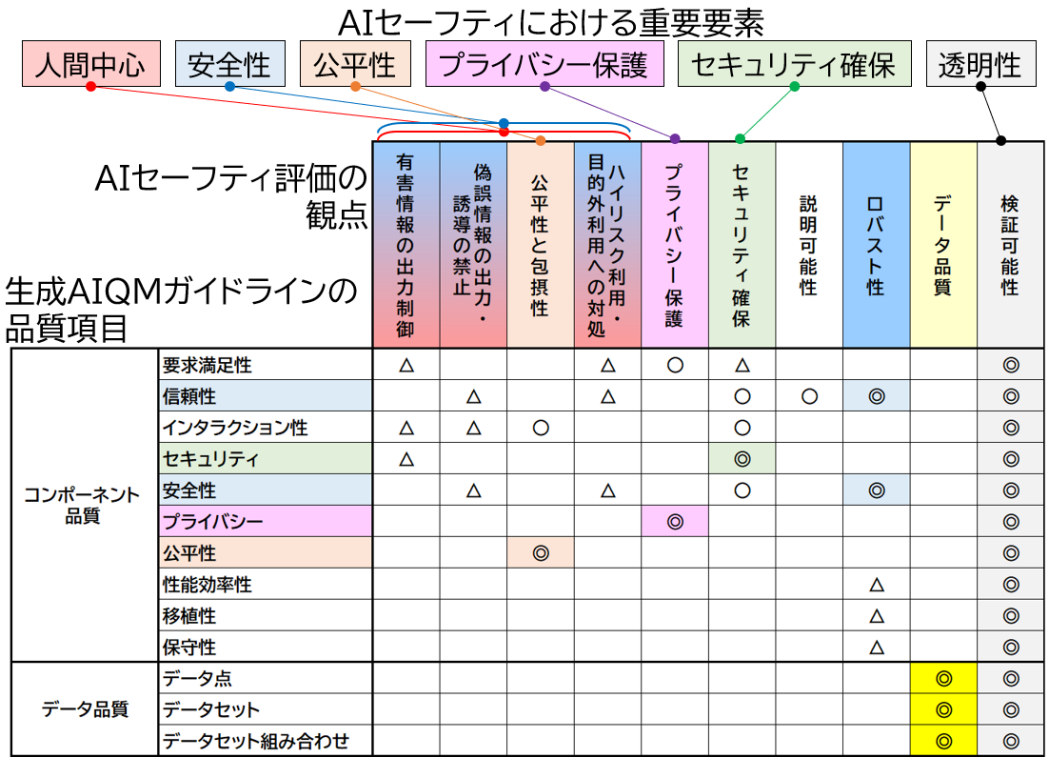

図 3 評価観点ガイドの評価観点と本書の品質項目の対応

評価観点ガイドが掲げる評価観点と本書の品質項目の対応を図 3 に示す。評価観点ガイドが挙げる AI セーフティにおける重要項目との対応も合わせて示した。公平性とプライバシー、セキュリティ、およびデータ品質の観点に対しては、直接対応する品質項目がある（◎）。ロバスト性の観点には信頼性と安全性の品質項目が対応する（◎）。検証可能性の観点は AI 品質マネジメント手法全体で確保するものであり、すべての品質項目が関わる。一方、有害情報、偽誤情報、ハイリスク利用・目的外利用の観点には、社会への影響と AI の利用者および近辺の第三者への影響があり、本書では後者のみ扱い（△）、要求満足性、信頼性、インタラクション性、セキュリティ、および安全性の品質で対応する。また、説明可能性は、本書では機能要件の一種とみなし（○）、常に必要なものとは考えない。

1.4.3 産総研の機械学習品質マネジメントガイドライン

これまでに産総研が発行した機械学習品質マネジメントガイドライン[AIQMv4]は識別や予測を行うシステムを対象としている。そこで扱うモデルは、教師あり学習等の機械学習手法を用いて、特定の種類の識別や予測を行うように開発されるものである(図 4 の 4)。

これに対し、本書は、大規模言語モデル等の、生成系の基盤モデルを利用する基盤モデル利用システムを対象としている。生成系の基盤モデルは入力によって様々な機能を発揮させることができ、自然言語処理や画像生成等の生成処理に用いられる(図 4 の 1)。

しかし、基盤モデル利用システムとして識別や予測を行うシステムを構築することは可能である。この場合、汎用の基盤モデルを用いて目的の用途を実施させるための仕組みについては、本書の記述が役に立つ。一方、テストデータセットの設計やテストの実施に関しては、従来のガイドラインの記述がそのまま適用できる(図 4 の 2)。

逆に、識別・予測のためのモデルを用いて、特定の生成系機能を実行するシステムを構築することも、目的とする機能によっては可能と考えられる。この場合、モデル開発には従来のガイドラインが役に立つ。一方、識別・予測のためのモデルを用いて生成系機能を実現するには、モデル以外の様々なコンポーネントが必要と考えられ、それらのコンポーネントの開発に関しては、基本的には従来のガイドラインが適用できるが、本書の記述（第 4 章、第 5 章など）が有用な場合もあり得る(図 4 の 3)。

<table>
<tr><td colspan="2" rowspan="2"></td><td colspan="2">システムの用途</td></tr>
<tr><td>生成</td><td>認識・予測</td></tr>
<tr><td rowspan="2">モデルの種類</td><td>生成</td><td>1）生成AI向けガイドライン</td><td>2）両ガイドラインを併用</td></tr>
<tr><td>認識・予測</td><td>3）両ガイドラインを併用</td><td>4）(従来の)機械学習品質マネジメントガイドライン</td></tr>
</table>

図 4 従来ガイドラインと本書の関係

1.5 用語

基本的な用語の説明を示す。

- 識別 AI・予測 AI：入力の特徴量から予め定められた値（離散値・連続値）を出力する機械学習 AI。モデル訓練に用いた学習データを近似した入出力関数を用いる。
- 生成 AI：確率モデルからのサンプリングによってデータを生成する機械学習 AI。モデル訓練に用いた学習データの確率分布を近似した生成モデルを用いる。
- 基盤モデル：さまざまな応用が可能な共通知識を学習した事前訓練済みモデル。特定の応用機能を持つ後段タスクの開発に利用する。
- 大規模言語モデル：自然言語マルチタスクの知識を学習した生成 AI の基盤モデル。LLM と略記する。
- プロンプト：生成モデルからデータ（コンテンツ）をサンプリングする際のトリガーとなる入力情報。LLM のコンテンツ生成条件を定める文脈を指定する。
- ハルシネーション：大規模言語モデルが、入力プロンプトが指定する文脈から逸脱したコンテンツ・内容が自己矛盾したコンテンツ・事実に反したコンテンツなどを生成すること。または、そのような生成コンテンツを指す。
- SQuaRE 品質モデル：システムおよびソフトウェアの品質モデルに関する ISO/IEC ならびに JIS 標準(3.2.1 節参照)。正式名称は「ソフトウェア品質要求と評価」である。
- Retrieval Augmented Generation (RAG)：生成モデルが訓練時に学習していないデータを、運用時に情報検索によって取得し、生成モデルへの入力の一部として与えることで生成モデルが取り扱う知識の範囲を補う手法。
- AI ガバナンス：一般には、AI が社会にもたらす便益を最大化し、またリスクを許容可能なレベルに低減する取り組みのこと。社会から見た広義の AI セーフティと関わる。

1.6 構成

本書の以降の構成を以下に示す。

第 2 章では、本書が想定する LLM 利用 AI システムの構成や開発プロセスを示す。

第 3 章では、LLM 利用 AI システムを対象とする品質マネジメントの考え方のあらましを示す。

第 4 章では、LLM 利用 AI システムが満たすべき品質項目の一覧を示す。ソフトウェア品質体系 SQuaRE にある品質項目をベースに、LLM 利用 AI システム特有の事項を考慮して拡充したものである。

第 5 章では、一般にデータが満たすべき品質項目の一覧を示す。これも SQuaRE の品質項目をベースに拡充したものである。個々のデータ点の品質に加え、データセットが単独で満たすべき品質、さらにデータセット間の関係に関する品質を挙げている。

第 6 章では、LLM 利用 AI システムの品質実現に向けた管理策を示す。第 2 章に示した LLM 利用 AI システムの構成に沿って、システム全体および各コンポーネントに関して行うべき管理策を列挙する。また各コンポーネントに関わりの深いデータについて、そのデータ品質を実現するための管理策も合わせて示す。

第 7 章では、LLM 利用 AI システムのセキュリティマネジメントに関する観点をまとめて述べる。

第 8 章では、終章として今後の課題と計画を示す。

2 スコープ

本ガイドラインで取り扱う品質マネジメントの対象範囲を示す。

2.1 大規模言語モデルと利用 AI システム

本ガイドラインの対象は生成 AI モデルを中核とするソフトウェアシステムである。生成 AI モデルとして大規模言語モデル（LLM）を基本とするが、画像や音声を含むマルチモーダル生成 AI モデルに共通する範囲を取り扱う。

2.1.1 生成 AI 利用 AI システム

対象は、LLM を利用するアプリケーションのソフトウェアシステムであって、生成 AI モデルの機能を補完するコンポーネント群からなる。図 5 に、対象システムを実現するアーキテクチャの前提となる情報の流れを示す。以下、生成 AI モデルを LLM として説明する。

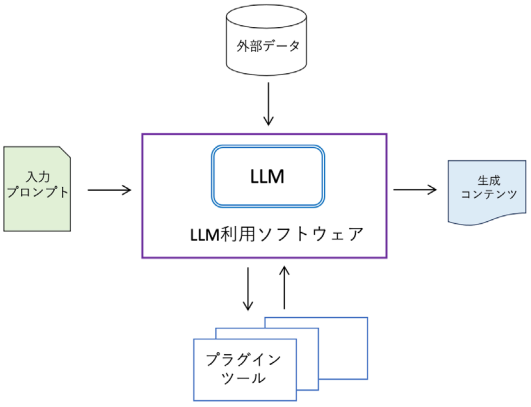

図 5 概念的な情報の流れ

一般に、エンドユーザーが期待するアプリケーション機能の実現に際しては、当該ソフトウェアシステムが作動する計算機環境の提供機能を活用する。LLM が訓練時に獲得し内部に保持する内部情報（内部知識）に加えて、外部データを必要に応じて検索したり、プラグインツールを経由して外部サービスを利用したりする。これらの外部連携機能を実現する上で、LLM を補完するソフトウェアコンポーネント（LLM 利用ソフトウェア）が必須となる。

本ガイドラインは、図 5 が示す構造を持つ様々な LLM 利用アプリケーションにかかわる品質マネジメントを統一的に取り扱う。図 5 で示したアーキテクチャは、以下に挙げるような様々な LLM 利用アプリケーションに共通して現れる基本構造となっている。

- 自然言語処理(NLP)アプリケーションシステム
  概念的な情報の流れ（図 5）は、エンドユーザーが入力したプロンプトにしたがって生成コンテンツを出力するという一般的な方法を想定した。LLM は自然言語処理（NLP）の能力を持ち、さまざまな NLP サービス機能を簡便に実現するベースとなる。多種多様なサービスが考えられ、合成（広告コピー生成など)、要約（議事録作成など)、分類（文書の感情分析など)、変換（機械翻訳など)、助言（仮想アシスタントなど)、検索（自律的な Web 検索など）に分類することが多い。このような NLP アプリケーションシステムは、エンドユーザー入力にしたがって適切なコンテンツを生成するものであって、図 5 が示唆する LLM 利用ソフトウェアのアーキテクチャに準じる。
- LLM エージェントシステム
  LLM エージェントやプランナー等と呼ばれる LLM 強化のリアクティブな基盤ソフトウェアシステムでは、エンドユーザーによる入力プロンプトの取り扱いが上記とは異なり、また、実現ソフトウェアアーキテクチャが拠り所とする計算の仕組みが複雑化する。LLM エージェントシステムでは、エンドユーザーが入力したゴールを達成する手順を生成（プランニング）し、必要な外部サービスを起動し利用することで、エンドユーザーが期待する機能を実現する。LLM は、ゴール生成・プランニング・外部サービス連携などの機能を実現し、ゴール達成に必要な外部サービスのオーケストレーションを担う基本となる。LLM エージェントシステムは、上記のプランニングやオーケストレーションが期待の振舞いを示すことを保証しなければならない。

LLM 利用ソフトウェア提供の仕方は、２通りあり、エンドユーザーが直接使用するシステムあるいは他システムに組み込まれるコンポーネントである。いずれにしても、LLM ならびに LLM を補完するソフトウェアコンポーネントから構成される。

コラム：LLM エージェントシステム

【LLM エージェントの定義】　LLM 利用 AI システムの一形態であって、ユーザーが与えた初期ゴールを達成する。特に、多数の外部サービスを適切に組み合わせるオーケストレーションを特徴とする。以下の技術からなる。

(1) 宣言的に表現されたゴールを達成する手続き的な処理列合成のプランニング

(2) 個々の処理を実現した分散サービスの実行全体を管理するオーケストレーション

【LLM エージェントシステム】 概念的に、LLM エージェントを、複数のサブシステム（LLM 利用 AI システム）が構成する複合的な LLM 利用 AI システムと考える。以下に概念的な構成の例を示す。

(a) Goal Creation：初期ゴールをプラン生成が可能な情報に具体化する
(b) Planning：ゴールを達成するオペレーションの列（プラン）を生成する
(c) Execution：外部サービス利用を実現するオペレーションの実行管理を担う
(d) Orchestration：(b)と(c)は、実行時に強く結びつきオーケストレーションを実現する。

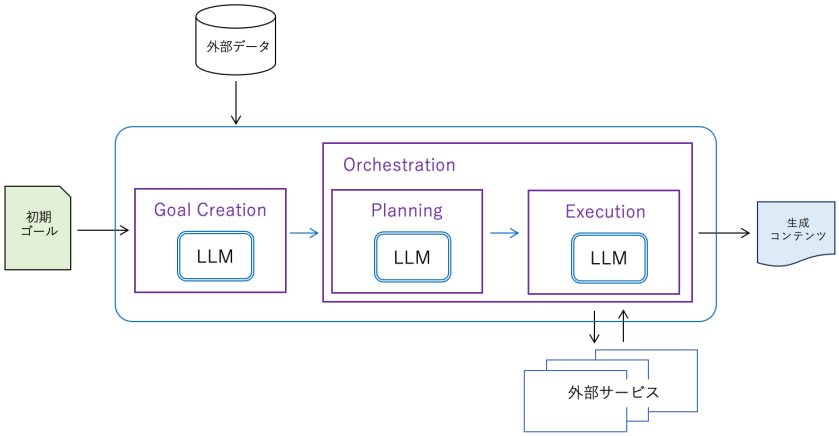

図 6 概念的 LLM エージェントの構成例

上記の Orchestration の説明から、Planning は実行時に Execution と交互に（インタリーブ）作動する動的なプランニングを行う。また、動的生成されたプラン（オペレーション列）は、外部分散サービスの実行を伴う。以上から、LLM エージェントシステムは、NLP アプリケーションシステムとしての LLM 利用 AI システムが満たす品質特性に加えて、複数の LLM 利用 AI システム（サブシステム）の連携、動的プランニング、分散サービス実行管理（オーケストレーション）に関わる品質特性を考慮する必要がある。

【LLM エージェント・デザインパターン】LLM エージェントシステムは、上記の概念アーキテクチャを具体化したデザインとして表現される。Auto-GPT（AutoGPT に改称）や HuggingGPT から始まる一連の研究成果を整理し、LLM エージェントシステムを構成するコンポーネントのデザインパターンを抽出する試み[Liu2024] など、エージェントのデザインに関する知見整理・共有の取り組みも広がってきた。特定の LLM エージェントシステム開発に際しては、システム要求達成という目的から適切なデザインパタ

ーンを選び、実行可能なコンポーネントとして実現すれば良い。この方法論は、本ガイドラインで想定しているパターンベースのコンポーネント開発（A2.2 節）と同じ考え方にたつ。

2.1.2 マルチモーダル AI システム

システムを構成する生成 AI が、マルチモーダルの場合を含む。テキスト・画像・ビデオ・音声・時系列などのデータを取り扱う。一般的には、複数種類のデータを同時に取り扱うことであるが、実際上、テキストを活用することから、マルチモーダル LLM が中心になる。また、LLM では、入力はプロンプトだったが、テキスト以外の種類のデータをコンテンツ生成指示に用いることができる。マルチモーダル LLM では入力を総称してインストラクションと呼ぶことが多い。

2.1.3 主なコンポーネント

生成 AI 利用システムにおける情報の流れとして図 5 を想定した場合、生成 AI 利用システム （図 5 の LLM 利用ソフトウェア）の主なコンポーネントとして以下のものが考えられる。ただし、このリストは当面の状況に基づいており、今後の生成 AI 利用システムの発展により、これ以外のコンポーネントの重要性が増すことは十分考えられる。

- 基盤モデル
- プロンプトとその周辺
- RAG 関連コンポーネント
- 外部連携コンポーネント
- 入力フィルター
- 出力フィルター
- HMI コンポーネント

これらのコンポーネントの位置づけや役割の詳細は付録 A2.2 に示す。以下では概略を述べる。

基盤モデルは大規模言語モデル(LLM)や画像生成モデル等の、プロンプトによって様々な生成系機能を提供する訓練済み学習モデルであって、本書の想定においては他社製の再利用部品として調達される。

プロンプトは基盤モデルに特定の機能を発揮させる役割を持つ。基盤モデルに与えられるプロンプトには、図 5 における入力プロンプトの他、開発時に策定されるシステムプロンプトや、次に述べる RAG コンポーネントや外部連携コンポーネントから得られた情報を含むことがある。

RAG 関連コンポーネントは Retrieval Augmented Generation 手法により基盤モデルが持つ内部知識を補い、またこの知識を上書きする役割を持つ。図 5 における外部データから、入力プロンプトに関わりの深い情報を検索し、プロンプトに加える。

外部連携コンポーネントは、LLM エージェントシステム(2.1.1 節)等において、基盤モデルが出力した手順に基づいて外部サービスを利用する役割を持つ。図 5 におけるプラグインサービスを介して外部サービスを呼び出し、外界に対する操作を行う。また外部サービスの応答として得られる情報をプロンプトに加える。

入力フィルターは図 5 の入力プロンプトから生成 AI 利用システムの機能や品質に悪影響を持つものを除去する役割を持つ。外部連携コンポーネントでプラグインツールから得られる外部から取得した情報にも適用される。

出力フィルターは基盤モデルの出力から生成 AI 利用システムの出力（図 5 の生成コンテンツ）として不要または不適切なものを除去する役割を持つ。外部連携コンポーネントでプラグインツールに与えて外部に送出する情報にも適用される。

HMI (Human Machine Interaction)コンポーネントは生成 AI 利用システムとそのユーザーのやり取りを担う。図 5 の入力プロンプトの構成要素となる情報をユーザーから受け取り、また生成コンテンツをユーザーに提示する役割を持つ。LLM エージェントシステムやその他複雑な構成と処理の流れを持つシステムでは、処理の流れのかなめとなる数か所で LLM が作成した実施手順やその実行にユーザーが介入する手段を提供する役割を持つこともある。

2.2 再利用開発モデル

本ガイドラインの対象の LLM 利用 AI システムは、LLM を再利用コンポーネントとして利用する。従来のソフトウェア再利用手法と比較検討することで、LLM 利用 AI システム開発の技術サプライチェーンを見通しよく整理できる。

2.2.1 基盤モデルの利用

LLM は、自然言語処理の基本的な機能を有する事前訓練済みモデル（Pre-trained Model）である。基盤モデル（Foundation Model）として、さまざまな応用システム構築のベースとなる。特定アプリケーション機能の構築では、転移学習の考え方に基づくファインチューニング（Fine-tuning）の方法を用いることが多い。応用向けに整備した学習データを用いて所与の LLM を再訓練するもので、目的とする機械学習タスクに応じて教師あり学習の方法を用いる。この方法は、訓練済み学習パラメータなど、基盤モデルの LLM に関する詳細な情報を必要とする。詳細情報を公開しない商用 LLM をベースとして、微調整の方法による再利用を独自で行うことが難しい。

GPT（Generative Pre-trained Transformer）は、文脈内学習に基づく LLM 再利用の方法を広めた。LLM が、さまざまな自然言語タスクを内在する汎用コンポーネントであるとし、入力情報（プロンプト）を工夫することで、期待するコンテンツを生成する。プロンプトは、LLM が応答生成する前提条件を与え、この前提条件がコンテンツ生成の文脈を指定すると考える。LLM は、プロンプトの内容、つまり文脈に基づいて、自身の振舞いを変更するように見える。特に、プロンプトに少数具体例を与える方法によると、追加した情報に応じて出力が改善する。そこで、プロンプトが指定する文脈の内容を学習すると考えて、文脈内学習（In-context Learning）と呼んだ。ファインチューニングの方法と異なり、機械学習の用語としての「学習」ではない。LLM の詳細情報を公開する必要がないことから、LLM 利用方法として広まっている。

2.2.2 LLM の FOR と WITH

FOR 開発と WITH 開発はソフトウェアの再利用における課題の観点を示す用語である[Sindre 1995][中谷 2025、第 12 章]。FOR 開発は「再利用のための(FOR)開発」、WITH 開発は「再利用による(WITH)開発」を指す。

FOR-LLM 開発と WITH-LLM 開発を整理する（図 7）。ここでは、機械学習技術の層と、それ以外の層(“ソフトウェア”) に分け、全体を 4 つに区分けした。

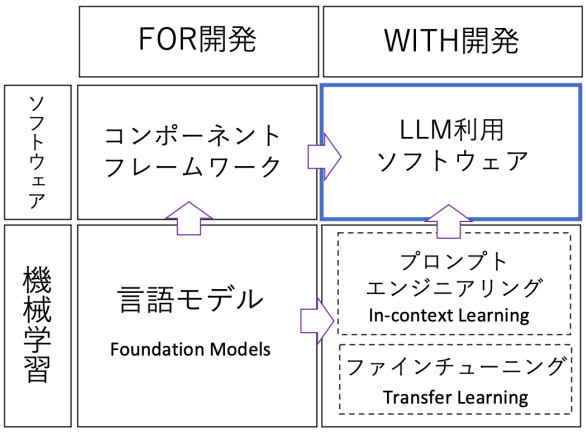

図 7 FOR と WITH の分業

応用目的に沿う期待通りの振舞いを実現するには、LLM 周辺のプログラムを適切に導入する必要がある。たとえば、訓練データに含まれない情報（データの最新性、組織内データ）の取り扱い、検索エンジンなど外部サービスとの連携、不適切な出力コンテンツのフィルタリングなど、LLM の外側で実現する補完機能を必要とする。LLM を利用した応用システム開発事例の広がりと共に、周辺で実現する補完機能の整理が進んでいる。

- 「機械学習-FOR 開発」は、機能部品としての LLM の開発で、学習データの整備から基本的なハルシネーション対策までを含む。機械学習の主要な技術領域である。マルチモーダル LLM の場合は、取り扱うデータの種類に応じて、用いる学習データと訓練の方法を工夫する。
- 「機械学習-WITH 開発」は、ファインチューニングによる転移学習あるいは文脈内学習をベースとしてプロンプトを工夫するプロンプトエンジニアリングがある。マルチモーダル LLM の訓練方法として、ファインチューニングを採用する場合がある。
- 「ソフトウェア-FOR 開発」は、LLM 利用ソフトウェアの共通機能を実現したコンポーネントフレームワーク開発であり、外部機能を実現したコンポーネントの提供を目的とする。
- 「ソフトウェア-WITH 開発」は、顧客要求から LLM 利用ソフトウェアシステムを構築し、ユーザーに提供可能な最終成果物を得ることである。コンポーネントを活用し、応用サービスの目的や顧客要求に応じた個別システム開発になる。

本ガイドラインは、「機械学習-FOR 開発」を対象としない。「ソフトウェア-WITH 開発」の最終成果物（LLM 利用ソフトウェア）を品質マネジメントの対象とし、当該成

果物の品質、およびそのために必要なファインチューニングによる「機械学習-WITH 開発」を論じる。

また、本ガイドラインは、LLM 利用ソフトウェアシステムの開発を対象とする品質マネジメントを論じることに注意して欲しい。「機械学習-WITH 開発」の要素技術として記載されているプロンプトエンジニアリングは、LLM に入力するプロンプトの記述を工夫する技術の総称であり、当該プロンプトの作成者は多岐にわたる。LLM の利用の仕方によっては、エンドユーザーがプロンプト作成者となる場合も考えられる。一方、本ガイドラインは、生成 AI 利用システム（2.1.1 項）開発過程でのプロンプト作成に関わる品質を論じる。つまり、ビジネスロジックを実現する実体としてのプロンプト（システムプロンプト）を対象とする。エンドユーザーが直接取り扱うプロンプト作成に関するプロンプトエンジニアリングは、いわばチャットボット利用技術であり、本ガイドラインの対象外とする。

2.2.3 USE の役割

LLM 利用ソフトウェアの品質マネジメントは、エンドユーザーによるシステム利用の仕方にも大きく関わる。LLM 利用ソフトウェアでは、入力仕様を明示することが難しい。入力プロンプトは自然言語テキストであって、「際限のない」柔軟さを持つ。不適切な状況を導く入力プロンプトを厳密に規定することは困難である。LLM 利用ソフトウェアシステムのエンドユーザーに、利用の仕方の制限を課す方策として、振舞い制限（Behavioral Restrictions）あるいは利用法の制限（Usage Restrictions）を導入する。これは、一般的、抽象的な表現（社会的倫理的道徳的な負のリスク軽減）にならざるを得ない。

特定のアプリケーションやサービスを提供する LLM 利用ソフトウェアシステムでは、当該システムの要求仕様や機能仕様をもとに、USE に課す制限事項「使用する際の指示（Instruction for Use）」を整理する。LLM の技術的な特徴から、このような情報を包括的に準備することは難しいため、「ソフトウェア-WITH 開発」の中で、当該 LLM 利用ソフトウェアシステムの要求仕様や機能仕様をベースに整理する必要がある。また、技術的に可能な範囲で、運用監視（Runtime Monitor or Human Oversight）を行う。

3 品質マネジメントの考え方

本ガイドラインの LLM 利用 AI システム品質の基本的な考え方を紹介する。

LLM 利用 AI システムに求められる品質は、従来のソフトウェアにも求められる信頼性や安全性などの観点と、AI の影響を受ける様々な立場からの要求を反映した社会的・倫理的要求を総合したものになる。

従来の識別 AI・予測 AI と異なり、LLM 利用 AI システムでは、核となる LLM の学習に使われるデータと、システムの機能が直接対応していない。幅広い学習データで訓練された LLM が提供しうる機能のうち、システムに必要なものを引き出し、不要なものを抑止することがシステムのコンポーネントの重要な役割となる。これを目標としたコンポーネントの品質マネジメントが、システムの品質マネジメントの主題となる。

コンポーネントの多くは機械学習によらない従来ソフトウェアとして実現されることから、従来ソフトウェアの品質体系 SQuaRE がコンポーネント品質マネジメントのベースとなる。一方、LLM 利用 AI システムに求められる社会的倫理的要求に応える上で、公平性やプライバシーなどもコンポーネント品質として考慮することが大切になる。

プロンプトは従来ソフトウェアにも従来 AI にも見られない LLM 利用 AI システム特有のコンポーネントであり、その品質マネジメントに独特の点が見られる。

3.1 LLM 利用 AI システムと品質

LLM 利用 AI システムは機械学習技術とソフトウェア技術の両面が関わる。

3.1.1 総合的な品質

一般に、ソフトウェアシステムでは、エンドユーザーが期待する要求を、品質を論じる上での拠り所とする。社会との関わりが大きい AI システムでは、システムを直接利用するエンドユーザーだけではなく、システム出力結果の影響を被る第 3 者の期待を考慮しなければならない。システムの利用時品質を論じる拠り所を定める上で、立場が異なる利害関係者からみた要求を明らかにする必要がある。高度なソフトウェアシステムに対する品質観点（システムの信頼性や安全性）に加えて、倫理的な観点からの要請（公

平性やプライバシー）を加える。さまざまな観点を総合した「信頼される AI（Trustworthy AI）[4]」に整理されている。

3.1.2 機能要求としての学習データ

機械学習 AI は、データ利活用を目的とすることから、システム要求の中心をなす機能要求に対して、従来のソフトウェアシステムと異なる考え方をとる。識別 AI や予測 AI では、モデル訓練に用いた学習データが、暗黙に、訓練済みモデルの機能振舞いを決めると考える。そこで、モデルが期待する振舞いを示すように、学習データを収集、整備することが、識別 AI や予測 AI の品質マネジメントの中心課題だった。

本ガイドラインが対象とする LLM 利用 AI システムでは、品質を論じる拠り所が、従来 AI と大きく変わる。LLM が開発と品質評価の対象（「機械学習-FOR 開発」）であれば、LLM 訓練に用いる学習データを、 2 段階に分けて考えることが多い。事前学習段階は、自然言語処理の汎用的な基盤モデルを得ることを目的とし、可能な限り広い範囲から構築した学習コーパスを LLM 訓練に用いた。品質を論じる拠り所は、生成した自然言語テキストの計量言語学上の知見であるが、学習コーパスに依存するという点で、学習データが暗黙に訓練済みモデルの機能振舞いを決める。事後学習段階は、不適切な出力を抑制することを目的とした微調整であって、何らかの適切さを基準として構築されたインストラクション学習データを用いる。品質の評価観点は、不適切さが低減できているかであり、インストラクション学習データに依存するという点で、学習データが暗黙に訓練済みモデルの機能振舞いを決める。

3.1.3 システムの仕様

LLM は、さまざまなアプリケーションサービスの機能要求を満たすと期待できる一方で、特定機能の実現に必要としない幅広い可能性を持つ。機能振舞いが未知であるような再利用部品をデザインの工夫によって如何に制御するかが、LLM 利用 AI システム

[4] 世界各地の公的機関[EU 2019][OECD 2019]や AI ベンダーがそれぞれの定義で用いており、原典と言えるものはないが、従来のソフトウェア品質に倫理的要請を加えたものという点は共通している。2018 年頃から 2021 年の間、さまざまな研究者による議論を経て、概ね、コンセンサスが得られている。

のデザイン上の工夫である[5]。LLM が特定の機能振舞いを示すために、どのような入力を与えればよいかは自明ではない。入力によっては、不要な機能が意図せず実行される場合もある。LLM が開発対象システムに必要な機能を提供するようにすることが重要であるが、同時に、不要な機能振舞いを LLM が示すのをできる限り防ぐことも必要である。また不要な機能の実行を抑止できなかった場合に、その影響をシステムの出力に反映させない工夫・出力を阻止する工夫が求められる。

LLM の機能振舞いは学習データで決まることから、訓練時のデータが含まない情報を反映しない。実務的には、最新のデータを扱えないこと（データ最新性）と社内情報など利用側データを持たないこと（組織内データ）が課題となる。また、LLM は自身の実行環境と情報をやり取りできないため、LLM 外部のソフトウェアが提供する機能との連携も課題となる。LLM 利用 AI システムの開発は、機能要求が与えられたとき、所与の LLM を使いこなす LLM 外部のソフトウェアを構築し、開発目的を達成することである。学習データを暗黙の仕様と考える従来の品質マネジメントとは異なり、開発対象システムの機能要求が品質を考える拠り所となる。

このように、システムに期待する機能仕様が品質の拠り所となることは、従来ソフトウェアシステムに対する品質の考え方と同じである。

3.2 LLM 利用 AI システムのデザインと品質

エンドユーザーが利用する AI システムとしての品質と、AI システムの内部構造（コンポーネントアーキテクチャ）にしたがって評価する品質に分けて考える。

3.2.1 システムとコンポーネント

LLM 利用 AI システムは、LLM の振舞いを制御するプロンプトと LLM 外部のソフトウェアを実現するコンポーネントアーキテクチャという異なる種類の実体からなる。コンポーネントアーキテクチャは、構造的には、複数コンポーネントならびにコンポーネントを結びつけるコネクターの集まりである。また、コンポーネントは自身を実現するコンポーネントとコネクターに展開可能という点で、階層性を持つ。この階層は分解不能なプリミティブコンポーネント（実行可能なプログラム）を端とする。

[5] LLM を「野生の暴れ馬」に見立て、LLM 利用 AI システムの開発は、期待する機能振舞いを示すように LLM を「飼い馴らすロデオゲーム」ということがある。

最終的な成果物である LLM 利用 AI システムの品質は「システムの利用時品質」として論じることができる。最上位のコンポーネントは「外部品質」を担い、LLM 利用 AI システムの利用時品質に寄与する。最上位以外のコンポーネントは「内部品質」を担う。外部品質ならびに内部品質は、コンポーネント階層内における位置付けの違いであって、品質観点は同じであり「コンポーネント品質」として論じる。つまり、「コンポーネント品質」は、「システムの利用時品質」に寄与する技術的な品質マネジメント方策の対象である。

本ガイドラインは LLM 利用 AI システムを対象とすることから、従来の「ソフトウェアおよびシステムの品質モデル（SQuaRE 品質モデル [6]）」と「信頼される AI」(3.1.1 節) の品質観点を整合的に組み合わせる。また、LLM 利用 AI システム固有の特徴を追加する必要がある。たとえば、合成型 NLP アプリケーションシステムや LLM エージェントシステムでは、出力のオープン性や計算の仕組みから生じる品質を考慮する。

「システムの利用時品質」は、たとえば、SQuaRE の「利用時の品質モデル（ISO/IEC25010:2011）」をベースとして「AI システムの品質モデル（ISO/IEC 25059:2023）[7]」を考慮し、LLM 特有の技術的な性質を加味することが考えられる。一方、「コンポーネント品質」は、SQuaRE の「製品品質モデル（ISO/IEC 25010:2011）」をベースとして「AI システムの品質モデル（ISO/IEC 25059:2023）」を考慮し、LLM 特有の技術的な性質を加味すればよい。また、データが LLM 利用 AI システムの品質に影響することから、SQuaRE の「データ品質モデル（JIS X 25012:2013）」をベースとして、データセット（データの集まり）に関わる技術的な性質を考慮する。

AIQM ガイドライン第 4 版は、「機械学習要素（訓練済みモデル）」を対象とし、外部品質ならびに内部品質を機械学習要素に対して定義した。これに対して、本ガイドラインの対象は LLM 利用 AI システムであり、内部品質ならびに外部品質を論じる対象は「LLM 外部のソフトウェアのコンポーネント」である。LLM は機械学習要素に対応し、再利用ソフトウェアコンポーネントである（2.2.2 節ならびに 3.1.3 節参照）。

---

[6] SQuaRE 品質モデルは計測指標を定義しているが、本ガイドラインでは、品質観点を整理する指針・チェックリストの観点として用いる。

[7] ISO/IEC 25059:2023 は、ISO/IEC 25010:2011 をベースとした標準文書である。今後、ISO/IEC 25019:2023 および ISO/IEC 25010:2023 に合わせて再整備される。

3.2.2 開発からの品質マネジメント

品質マネジメントの主題は LLM 利用 AI システムの「コンポーネント品質」である。開発の流れを整理し、開発成果物の特徴に応じた品質マネジメントの方針を明らかにする。与えられた要求機能を実現することが基本である一方で、要求を整理する段階では、期待する機能が実現可能かを確認する必要がある [8]。

LLM を利用する場合、期待する機能が LLM によって実現可能か（実現可能性）を確認する作業では、主に、プロンプトデザインの試行錯誤によってプロンプトを作成するプロトタイピング手法を採用する。このプロトタイピングでは、プロンプトだけでなく、対象とする LLM 利用 AI システムを実現することになるアーキテクチャやそこで用いられるコンポーネント、例えばファイルデータベースやそこに収めるデータの概形なども同時に検討するが、それらを踏まえて、適切なプロンプトを工夫する。この作業は、実現可能な機能要求を検討することであり、コンポーネントアーキテクチャのデザインに先立つ要求獲得・形式化（Requirements Acquisition and Formalization）の一部とみなすことができる。ラピッドプロトタイピング手法によって、この要求形式化の結果として得たプロンプトは、開発対象システムの構成要素となる。プログラミング技術によって実現するコンポーネントではないが、システムを構成するという点で「コンポーネント品質」に寄与すると考えられる。そこで、プロンプトは品質マネジメントの対象になり、適切な品質特性を持つことをプロトタイピングの過程で確認する。

LLM 外部のソフトウェアを実現するコンポーネントアーキテクチャのデザインは、実現可能性が確認された要求機能を実現する。構成するコンポーネントのデザイン過程で適切な「コンポーネント品質」を持つことを確認する。従来ソフトウェア開発と同様に、要求機能仕様をベースとする V 字型開発の考え方にしたがって品質マネジメントを実施する。この V 字型開発の最終ステージで、プロンプトならびにコンポーネントアーキテクチャを組み上げた最終成果物である LLM 利用 AI システムに対して、外部品質としての「コンポーネント品質」を確認する。

[8] ソフトウェア工学（要求工学）の標準的な考え方によると、「要求」はステークホルダーの期待を表す一方、「要求仕様」は実現可能な要求の仕様表現である。

3.3 システムの利用時品質

システムの利用時品質は、ステークホルダーによる評価の視点を整理したもので、実運用時のシステムに対して評価するエンドツゥエンド（End-to-end）の品質観点である。SQuaRE ソフトウェア品質要求と評価に関する国際標準の ISO/IEC 25000 シリーズ（SQuaRE）は、利用時の品質モデルを整理している。

ISO/IEC 25010:2011（JIS X25010:2013）は製品品質モデルと共に、エンドユーザーならびに保守運用者をステークホルダーとする利用時の品質モデルを 5 つの品質特性とした。また、ISO/IEC 25010:2011 の枠組みにしたがって AI システムの技術的な特徴を考慮した ISO/IEC 25059:2023（AI システムの品質モデル）がまとめられた。その後、システムの社会への影響が増大することから、組織や公共・社会など第 3 者をステークホルダーに含む ISO/IEC 25019:2023 が策定されるに至った。なお、ISO/IEC 25010:2011 の製品品質モデルは ISO/IEC 25010:2023 に再整理されている。以上の流れを図 8 に示す。

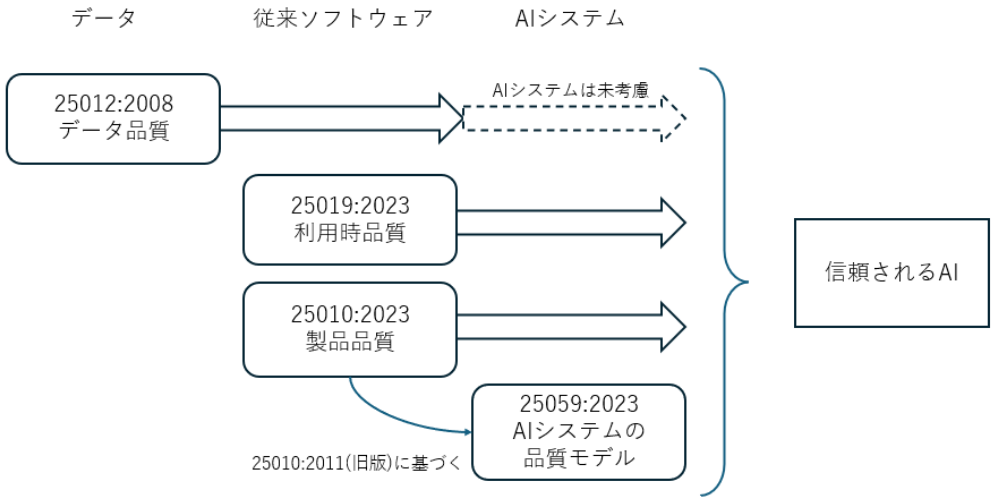

図 8 SQuaRE の AI システム対応

ソシオテクノロジーとしての役割が大きい AI システムでは、ステークホルダーが多岐にわたる。利用時の品質への具体的な期待は、ステークホルダーによって異なり、社会から見た AI ガバナンスとして、システム開発に関わる品質マネジメントに先立って論じるべきことである。

コラム：X 25010:2013 における利用時の品質モデル

ISO/IEC 25010:2011（JIS X25010:2013）は制定時期から推察できるように、従来のソフトウェアシステムに関わる。ここで 5 つの品質特性を LLM 利用 AI システムに適用する場合の解釈を示す。

- 有効性は、ステークホルダーが持つ目標達成に LLM 利用 AI システムが貢献することである。ステークホルダーが示すビジネス上の KPI にしたがって、LLM 利用 AI システムの役割を明らかにする。LLM 利用 AI システムの計画・導入・運用に関わる。
- 効率性は、ステークホルダーが持つ目標達成に LLM 利用 AI システムを使用する際に必要となる利用資源である。LLM 利用 AI システムの計画・導入・運用に関わる人的資源や LLM 利用 AI システム実行に必要な計算資源が関わる。
- 満足性は、ステークホルダーの期待する要求を LLM 利用 AI システムが達成することである。自然言語が HMI となる LLM 利用 AI システムの技術的な特徴によって全般的な満足性が向上する。
- リスク回避性は、第 3 者を含むステークホルダーに、AI がもたらす潜在的な負のリスクを緩和することである。LLM 利用 AI システムが出力したコンテンツが、リスクをもたらす内容を含むとき脅威が大きい。何を脅威とするかの決定は、ステークホルダーによる。
- 利用状況網羅性は、想定された状況ならびに想定外の状況で、他の利用時品質観点（有効性・効率性・満足性・リスク回避性）の前提下で LLM 利用 AI システムを使用できることである。

コラム：ISO/IEC 25019:2023 の品質モデル

ISO/IEC 25019:2023 は、AI システムなどのソシオテクノロジーとの親和性から、利用時の品質モデルを 3 つの品質特性に整理した。ISO/IEC 25010:2011 からの大きな変更は、品質特性として社会受容性を明示したことである。

- 便益は、ユーザビリティ・アクセシビリティ・適合性の 3 つの品質副特性からなる。ISO/IEC 25010:2011 の有効性を軸に、利用者との関わりから再構成した内容になっている。
- 安全は、経済的リスクの回避性・健康リスクの回避性・環境や社会的リスクの回避性・人間の生活へのリスクの回避性の 4 つの品質副特性からなる。

ISO/IEC 25010:2011 ならびに ISO/IEC 25059:2023 のリスク回避性を再構成した内容になっている。

- 社会受容性は、経験・信用・遵法・倫理の 4 つの品質副特性からなる。ソシオテクノロジーとしての役割に注目し、第 3 者がシステムを受け入れる条件を品質特性として整理している。

なお、社会受容性を品質特性としない考え方もある。信頼される AI（Trustworthy AI）（3.1.1 節）では、主体（社会・人）と客体（AI システム）の間に「トラスト（Trust）」関係が生まれる時に、社会受容性が成立すると考える。トラストに関わる標準的な理論では、信頼される AI が示す品質を AI システムが満たすことで、主体が ABI を知覚し、その結果、トラスト関係が生じるとする。ここで、ABI は能力（Ability）・善良さ（Benevolence）・誠実さ（Integrity）である。倫理面や道徳面の特徴は善良さに寄与する。

3.4 プロンプトと品質

第 4 章と第 5 章で、コンポーネント品質、データ品質の順番で、LLM 利用 AI システムの品質観点を論じる。その準備として、プロンプトデザインに関わる品質と再利用コンポーネントとしての LLM 品質に対する注意事項を整理する。

3.4.1 プロンプトデザインに関わる品質

プロンプトは、LLM の振舞いを制御する入力情報であり、LLM を応用目的に即して利用する上での基本である。LLM から、その能力を引き出す条件を明示する工夫といえる。プロンプトデザインは、LLM 利用 AI システム機能要求の実現性と関わる。この成果物（システムプロンプト）は最終システムに組み込まれたビジネスロジックの中心として、目的とする LLM 利用 AI システムの中核(2.2.2 節)を担う [9]。重要な成果物であることから、以下、プロンプトに関わる品質の論点を整理しておく。

プロンプトは自然言語が持つ表現力の大きさや柔軟さを引き継ぐ。散文的な書き方では、エンジニアリング上、修正性や試験性に劣る。プロンプトデザインでは、プロンプト全体を構造化し、プロンプトの段落ごとに計算上の役割を明確化する。ここで、計算

[9] 先にも述べた通り、プロンプトデザインは野生の暴れ馬（LLM）を飼い慣らすロデオゲームである。

上の役割とは、自然言語文を計算機上の処理対象と見做したときに、データの役割、処理ステップ指示の役割、プロンプトをメタ制御する役割（「以降を無視する」等）、などである。たとえば、エンドユーザー入力プロンプトはデータとしてシステムに取り込まれた後、メタ制御の働きをするシステムプロンプトの制御下で、処理ステップ指示の役割を果たす[10]、などがある。このような役割に応じて、プロンプトに対する品質を適切な観点から論じる。

プロンプトは LLM への入力であり、一般に出所の異なるいくつかの要素で構成される。通常は、システムプロンプトと、エンドユーザー入力プロンプトの組であるが、場合によっては、外部の連携サービス（他の計算システム）が発生源になる要素を含むことがある（2.2.1 節）。そこで、プロンプトの来歴（真正性・信憑性）が重要なことがある（5.1 節）。

また、情報セキュリティからの見方を考慮する必要がある。

- 攻撃手段としてのユーザープロンプトという側面
- 保護すべき情報資産としてのシステムプロンプトという側面

前者はサイバーセキュリティと関わり、後者は品質特性としてのセキュリティ（4.4 節・A3.2 節）が関わる。

3.4.2 LLM の品質の文書化

プロンプトは LLM への入力であり、プロンプトが引き起こす LLM の機能振舞いは LLM に依存して総合的に決まる。プロンプトの品質は LLM の品質に依存する。一方、品質マネジメントの対象になる LLM 利用 AI システムでは、LLM は所与の再利用部品であり、「機械学習-FOR 開発」の成果物で品質が決まっている。LLM を利用する際には、多数の候補から、適切な品質の LLM を選択する。その際、LLM は適切な技術情報を伴わなければならない。LLM 自身だけではなく、訓練に用いた学習コーパス構築に関わる技術情報を含む。このような情報が AI BOM として標準になれば、適切な LLM 選択が可能になる。

LLM 利用 AI システムの運用時（USE）に障害が生じる場合、損害に対する責任が生じる。利用した LLM の能力に起因するとき、最終製品の提供者（「ソフトウェア-WITH

[10] LISP1.5 で S 式データを関数として評価（eval）するような感じだろうか。

開発」）と LLM の提供者（「機械学習-FOR 開発」）の間で責任を分担する。AI BOM が社会的な責任を論じるベースとなる。

現時点（2024 秋）で、AI BOM の標準は定められていない。いくつかの方向から議論が進んでいる。

- 機械学習技術の民間企業：Model Cards あるいは System Cards
- 技術コンソーシアム：Linux Foundation の AI BOM
- 規制法の規定：AI-ACT の第 53 条および実践規範(Code of Practice, CoP)

また、オープンな技術コミュニティでは、LLM 訓練に関わる技術情報（訓練に用いた学習データ・学習アーキテクチャ・訓練済みモデル・評価データなど）を公開する動きがある。LLM 利用者から見ると、LLM 利用 AI システムの品質と関わる情報を手軽に参照できることが望ましい。今後、AI BOM の標準化動向に注視する必要がある。

4 LLM 利用 AI システムが満たすべき品質

「コンポーネント品質」は、技術的な品質マネジメント方策の主題である。開発対象の LLM 利用 AI システムに対するゴール要求を基準として、期待される水準の「コンポーネント品質」を達成する。

本章に掲げる「コンポーネント品質」は、SQuaRE の「製品品質モデル（JIS X 25010:2013）」を枠組みとして「AI システムの品質モデル（ISO/IEC 25059:2023）」ならびに「製品品質モデル（ISO/IEC25010:2023）」を考慮し、LLM 利用 AI システム特有の技術的な性質を加味して整理した。品質特性の観点は、SQuaRE モデルに準じるが、以下に示した各々の品質副特性（4.1 節から 4.10 節）は、LLM 利用 AI システムの技術的な特徴を考慮して詳細化する。

- 要求満足性
- 信頼性
- インタラクション性
- セキュリティ
- 安全性
- プライバシー
- 公平性
- 性能効率性
- 移植性
- 保守性

品質観点（品質特性・品質副特性）は LLM 利用 AI システムを構成するコンポーネント・デザインの指針となる。概念的な情報の流れ（図 5）を具体化した「NLP アプリケーションシステム」ならびに「LLM エージェントシステム」が示すシステムアーキテクチャに即して、各々の品質観点と強く関わるコンポーネントのデザインを議論する。

本章は、LLM 利用 AI システムやそのコンポーネントに求められる可能性のある品質観点のカタログであって、ここに掲げたすべての観点での品質を必ず満たすことを求めるものではない。システムやコンポーネントの用途や機能仕様に基づいて、どの品質が求められるかを判断する必要がある。品質観点にはしばしば相互に関連があり、ある品質観点でシステムやコンポーネントを改善することが他の品質の改善に役立つことがある一方、品質観点の間でトレードオフが生じることも多い。さらに、あるシステムやコンポーネントに求められる品質の中には、AI 以前のソフトウェアの品質マネジメン

トの手法で対処すればよいものもある。第 6 章では、LLM 利用 AI システムにしばしば求められる品質に関して、LLM 利用 AI システムに強くかかわる管理策を中心に示す。

4.1 要求満足性

要求満足性（Requirements Satisfiability）[11]は、LLM 利用 AI システムに期待されるソフトウェア要求を満たすことである。ソフトウェア要求は機能要求と機能外要求から構成される。これらに対応する 2 つの品質副特性に分ける。期待されるソフトウェア要求を満たすように、LLM 利用 AI システムをデザインし開発（構築・検査）する。

4.1.1 機能要求満足性

機能要求満足性（Functional Requirements Satisfiability）は、LLM 利用 AI システムが所与の機能要求を満たすことである。

- 明示的および暗黙に与えられる機能要求を評価の対象とする。
- LLM 利用 AI システムのエンドツゥエンド（End-to-End）評価であって、明示的に与えられる機能要求に対して、生成コンテンツの定性的な評価を行う。
- 暗黙に与えられる機能要求には、社会的倫理的な観点からの期待を含む場合がある。このような暗黙の要求が満足されているかの確認では、想定外の状況下での評価を実施することを想定する。LLM 利用 AI システムが社会的倫理的な観点から見て不適切なコンテンツを出力する可能性を技術的に避けることができないことが前提になることに注意する。所与の機能要求を整理し、構築する過程で、AI ガバナンスによって、当該の機能要求が社会的倫理的な観点に配慮していることを前提とする。
- LLM 利用 AI システムに期待されるソフトウェア要求を満たすように､LLM 利用 AI システムのデザインを行い、LLM 利用 AI システムを実現するソフトウェア成果物（プロンプトや LLM 周辺コンポーネントなど）を開発する。

4.1.2 品質要求満足性

品質要求満足性（Quality Requirements Satisfiability）は、LLM 利用 AI システムが所与の機能外要求（品質要求）を満たすことである。LLM 利用 AI システムの機能振舞いが、

[11] 「デザインからの品質」の考え方に基づき、SQuaRE 製品品質モデルの「機能適合性」を「要求満足性」に拡張した。

本ガイドラインの 4.2 節から 4.10 節までに掲げるような規定による品質観点を満たすことを確認する。機能要求満足性（4.1.1 節）と組になる。

4.2 信頼性

信頼性（Reliability）は、期待される機能を実行すること、つまり、所与の仕様を満たすことである。

- LLM 利用 AI システムに関わる信頼性は、機能要求満足性（5.1.1 項）が参照する機能要求を実行することである。
- コンポーネントに関わる信頼性は、各コンポーネントに期待される機能を実行することである。
- 所与の仕様が適切に実現されておらず開発過程で欠陥が混入する場合、信頼性が劣化する原因となる。また、開発時から利用状況が変化し所与の仕様が適切でない状況に至る場合、信頼性が満たされなくなる。

4.2.1 成熟性

成熟性（Maturity）[12]は、想定された作動条件の下で、期待する信頼性を満足することである。大まかには、所与の機能仕様・デザインを適切に実現していることである。

- 大まかには、検査対象が、所与の機能仕様を実現していることである。動的検査で確認し、従来からのソフトウェアテスティング技術の方法を用いる。

4.2.2 ロバスト性

ロバスト性（Robustness）は、通常期待する基準データから外れたデータが入力された時に、基準データが入力された場合と異なる振舞いの度合いに関わる。機械学習特有のモデルロバスト性と一般のソフトウェアと共通するプログラムロバスト性がある。

- モデルロバスト性は、基準データと評価データを訓練済みモデルに入力した場合の各々の出力を比較することで、当該モデルのロバスト性を評価することができる。基本的には、基準データと評価データの違いの度合いが、各々に対する出力の違いに大きく影響せず、滑らかに変化することである。

[12] ISO/IEC 25010:2023 では、Faultlessness に名称変更されている。無欠陥性として良いかどうか、対応する JIS X 規格の出版後に決定する。

- 基準データからの違いの特徴に応じて、さまざまな評価データを考えることができる。画像では、基準データにノイズ[13]を付加して評価データを作成すればよい。また、評価データの特徴に応じて、モデルロバスト性を細分化する。特に、敵対データを用いる時は敵対ロバスト性、欠損データを用いる時は欠損ロバスト性と呼ぶ。
- 生成モデルのモデルロバスト性では、基準データと評価データの違いの度合いが、質的に異なる出力を導く場合があることを考慮した評価指標を導入する。

- プログラムロバスト性は、通常の期待から外れた値のデータが入力された時のプログラム実行状況を対象として考える性質である。機械学習ソフトウェアなど数値計算が主体のプログラムの場合には入力データ処理に関わる数値精度が実行に影響することがある（数値精度の影響）。また、繰り返し実行することでプログラムの実行環境を構成するメモリ等が不正に消費され、以降のプログラム実行が円滑に進まないことがある（メモリリークの影響）。プログラムロバスト性の劣化は信頼性に影響する。
- 一般には、ロバスト性としてシステムロバスト性を考えることがある。これは、システムの安定運用に関わり、可用性（4.2.5 項）・障害許容性（4.2.6 項）・回復性（4.2.7 項）で取り扱う。

4.2.3　出力一貫性

出力一貫性（Output Consistency）は、同一入力を繰り返したとき、入力に対する出力内容（生成コンテンツ）が整合していることである。

- 出力一貫性は、確率的な振舞いを示すコンポーネントに対して考える品質副特性であり、確率的な振舞いの現れとして顕在化する。
- ある入力データを基準データとする時、モデルロバスト性（4.2.2 節）は基準データと異なる評価データを入力するが、出力一貫性は同一の基準データを繰り返し入力する場合を評価対象とする品質副特性である。
- 同一入力に対して常に同じ出力を生成する確定的な振舞いのコンポーネントは出力一貫性を満たす。また、同一入力に対して非決定的な振舞いを示す一方で、取り得る出力値が予測可能な場合は、出力が異なる場合であっても一連の出力に整合性があることから、出力一貫性を満たす。一方、LLM はその確率的な振

[13] ホワイトノイズ・ガウスノイズなどがある。意味上の正確性に影響する敵対データはセマンティックノイズを付加したと考える。

舞い[14]に起因して、同一入力に対して、想定を逸脱するような異なる出力[15]が生成されることがある。

4.2.4　進行性

進行性（Progress）は、コンポーネントの動的な処理過程に関わり、想定外に停止する（デッドロック）、想定外の繰り返し閉路に陥る（ライブロック）といった不適切な状況に陥らないことである。コンピューティングに関わる性質である。

- 進行性を満たすコンポーネントは、その処理進行過程の全ての状況で想定通りの適切な状況にある（コンピューティングに関わる性質としての安全性）。

4.2.5　可用性

可用性（Availability）は、期待する使用の状況下で、LLM 利用 AI システムが、意図した機能振舞いを示し、運用可能なことである。

- システムロバスト性（5.2.2 項）に関係した品質副特性で、LLM 利用 AI システムが安定作動することである。その結果、期待されるソフトウェア要求を満たすという点で、機能要求満足性（5.1 節）と関わる。

4.2.6　耐故障性

耐故障性（Fault Tolerance）は、故障にもかかわらず、LLM 利用 AI システムが、意図したように運用できることである。

- 故障（Failure）は、システムに付随する欠陥が原因となって、システムが期待と異なる機能振舞いを示す状況である。欠陥（Fault）は、その特徴によって、偶発的な欠陥・決定論的な欠陥の 2 つに分類される。
- LLM 利用 AI システムが分散コンピューティング基盤上で作動する場合、ネットワーク障害などの偶発的な欠陥が生じる可能性がある。LLM 利用 AI システムの LLM 周辺コンポーネントのデザインによって、故障を避ける。
- 一般にソフトウェアシステムでは決定論的な欠陥への対策を講じる。この対策は成熟性（5.2.1 項）を満たすことであり、素朴には、デザインならびに実現の

[14] LLM の確率的な振舞いに関して付録 A1.1.3 および A1.2.2 を参照のこと。
[15] 例えば、LLM に同じトークンだけを何百回も続けて出力するよう求めると、高確率でその出力の途中で突然訓練データを出力し始める、という報告[Nasr 2023]がある。

不具合を事前に除去することである。

4.2.7 回復性

回復性（Resilience）は、LLM 利用 AI システムが、故障によって処理を中断した時に、直接影響を受けた情報を回復し、システムの状態を復元できることである。

4.3 インタラクション性

インタラクション性（Interaction Capability）[16]は、期待される状況で、エンドユーザー自身が想定するように LLM 利用 AI システムを利用できることである。

4.3.1 適切度認識性

適切度認識性（Appropriateness Recognizability）は、エンドユーザーの期待に対して、LLM 利用 AI システムが適切であるかを、エンドユーザーが事前に認識できることである。

4.3.2 アクセシビリティ

アクセシビリティ（Accessibility）は、期待される状況で、幅広い範囲の心身特性および能力を持つエンドユーザーが LLM 利用 AI システムを利用できることである。

4.3.3 可制御性

可制御性（Controllability）は、LLM 利用 AI システムが運用操作しやすく、制御しやすくする技術的な特性を持つことである。稼働中の LLM 利用 AI システムの動作が外部から変更可能なことである。

- 「LLM エージェントシステム」（2.1.1 節）に分類される LLM 利用 AI システムは、自律エージェントやプランナーとして機能することから、想定外の動作を示すことがあり、極端な場合では「暴走」するとも言える。このような LLM 利用 AI システムによる「過剰な代行（Excessive Agency）」の悪影響を軽減する必要がある。

[16] 従来の「使用性」。最新の議論に合わせて「インタラクション性（仮訳）」とした。

- 可制御性は、運用中に過剰な代行の悪影響が生じた際、外部から LLM 利用 AI システムの動作振舞いを制御できることである。また、信頼性の品質副特性である「進行性」（4.2.4 節）が劣化した場合の制御として、緊急停止を可能にする。

4.3.4 説明性

説明性（Explicability）は、LLM 利用 AI システムが出力コンテンツを生成した理由に関わる情報を補って提示できることで、出力した目的コンテンツを正当化（Justification）する情報に関わる [17]。

- デザイン上、補う情報源を LLM 利用 AI システムの外部から獲得するか、内部で生成するかの違いがある。
- 外部情報を補完する方法は、生成理由に関わる情報を外部から獲得し、エンドユーザーの指示にしたがって提示する。提示する情報の代表的な例として、生成コンテンツの情報源がある。特に、客観的な事実情報を補うとき、生成コンテンツの「事実性」によって、説明性が向上する。
- 内部情報を活用して補完する方法は、入力から出力に至る推論過程に関わる情報（内省情報）を整理し提示する。必ずしも「事実性」を保証することではなく、期待の確信レベルを高めることで、説明性を向上する。
- セキュリティの品質副特性である責任追跡性（4.4.4 項）と補完的な役割を果たす。

4.3.5 習得性

習得性（Learnability）は、期待される状況で、エンドユーザーが LLM 利用 AI システムを利用しやすいことである。

[17] 説明可能 AI（Explainable AI）は解釈性と透明性からなる。一方、出力の正当化理由は弱く、その理由自身の根拠が確からしさに劣ることがある。

4.4 セキュリティ [18]

セキュリティは、LLM 利用 AI システムが管理する情報を保護することである。具体的な保護対象をシステムアーキテクチャ（2.1.1 節）に基づいて決める。一般的には、ビジネスロジックの役割を担うシステムプロンプト（3.4.1 節）・訓練済み学習モデルの LLM（3.4.2 節）・LLM への入力の一部を構成する外部データ・プラグイン方式で連携する外部ツール・エンドユーザーが入力するユーザープロンプトなど情報が保護対象となる。デザインからのセキュリティ（Security by Design）に基づいて、これらの情報を保護する技術的な方策を講じる。

4.4.1 アクセス制御性

アクセス制御性（Access Control）[19]は、利用許可条件にしたがって情報を保護し処理できることである。

- アクセス制御は、概念的には、制御対象の情報（オブジェクト）・アクセス主体（サブジェクト）・処理内容（アクション）で構成される。「オブジェクトに対してサブジェクトが施すアクション」の許可・不許可をデザインすることで、オブジェクトを保護する。
- 利用許可条件は、LLM 利用 AI システムの要求からトップダウンに決まるポリシーとして規定されることがある。期待されるアクセス制御性の実現は、要求を満足することであり、機能要求満足性（4.1.1 節）と関わる。
- 利用許可条件は、LLM 利用 AI システムを構成するコンポーネントの役割から決まるポリシーで規定されることがあり、当該ポリシーを反映したデザインによって、アクセス制御性を満たす。保護対象に、プロンプト（システムプロンプト）・LLM・プラグインサービスがある。
  - システムプロンプトは、ビジネスロジックの役割を担う。エンドユーザー入力を契機として LLM への入力の一部を構成する。想定外の漏洩、想定外の改変を防止することで、システムプロンプトを保護する。
  - LLM への入力プロンプトには、できる限り信頼できる出元から得たテキストのみを含むようにすることで、LLM を保護する。例えば外部連携コン

[18] 本節では SQuaRE の文脈におけるセキュリティのみを扱う。サイバーセキュリティの観点を含めた、より包括的なセキュリティについては第 7 章で扱う。
[19] SQuaRE 製品品質モデルの「機密性」と「インテグリティ」を利用許可条件にしたがった「アクセス制御性」に統合した。

ポーネントを通じて呼び出せる外部サービスとして信頼できるものを設計時に選び、それ以外のサービスからの戻り値が LLM に入力されないようにする。

- プラグインツールに規定された利用許可条件にしたがって、当該プラグインツールを保護する。

4.4.2 介入性

介入性（Intervenability）は、稼働中の LLM 利用 AI システムの動作に、外部から介入し、システムの状態を安定させることである。

- 介入性は、稼働中の LLM 利用 AI システムの動作を中断し、その後、再開することを可能とするデザインを扱い、信頼性の品質副特性である可用性（4.2.5 節）ならびに回復性（4.2.7 節）と関わる。
- 介入性は、信頼性の品質副特性である「進行性」（4.2.4 節）が劣化した場合に、「緊急停止」などの制御を行うことを可能にする。
- インタラクション性の品質副特性である可制御性（5.3.3 項）は、ユーザーが直接に稼働中の LLM 利用 AI システムの動作を変更可能な HMI デザインを扱う。介入性を実現するには、可制御性の実現の一環として、例えば「大きな赤い非常停止ボタン」などを設ける必要がある。

4.4.3 真正性

真正性（Authenticity）は、アクセス制御に関わる制御対象（オブジェクト）に関わる情報ならびにアクセス主体（サブジェクト）に関わる情報の身元（Identity）が明らかなことである。

- サブジェクトがエンドユーザーの場合、ユーザー認証など、有効なエンドユーザーであることが確実になるデザインとシステム運用方策を採用する。
- サブジェクトが外部サービスの場合、来歴などの情報を活用して、正当なサービスであることを確認する。

4.4.4　責任追跡性

責任追跡性（Accountability[20]）は、LLM 利用 AI システムの出力コンテンツを生成した理由や要因を再現し提示できることである。システム作動の実行履歴（ログ情報）を活用した実行監視（Runtime Monitors）がある。

- LLM 利用 AI システムが出力するコンテンツは、不適切な内容（ハルシネーション・作り話）を含むことが避けられないことが知られている。そこで、運用者による監視（Human Oversight）への技術的なアプローチとして、実行監視の方策を講じる。

4.5　安全性

安全性（Safety）は、設計者が許容できるレベルまでリスク低減した上で、かつ、定められた条件下で、LLM 利用 AI システムが、外部に危害を及ぼす状態に陥らないことである [21]。

- 定められた条件は、当該 LLM 利用 AI システムの運用制限条項やエンドユーザー利用の制限条項を含む。つまり、想定されていない使い方・悪意による使い方に起因する状況（1.3.1 節）への対処としては、これらの条件を定めて利用者に提示することができる。
- LLM 利用 AI システムの出力コンテンツが、外部に及ぼす危害の直接原因になる。
  - 一般には、機能要求満足性（4.1.1 節）・ロバスト性（4.2.2 節）の劣化として現れる。
  - 信頼される AI（Trustworthy AI）では、倫理面に関わる性質に起因した問題点が顕在化する。固有の技術的な特徴を持つこと、また、その重要性が大きいことから、個別に、プライバシー（4.6 節）・公平性（4.7 節）として論じる。
- 安全性は、当該性質を論じる文脈によって、その定義が異なる。
  - ISO/IEC の文脈（ISO/IEC Guide 51(JIS Z 8051)、IEC61508、ISO/IEC TR5469 など）では、安全(safety)の定義は「許容不可能なリスクがないこと（freedom from risk which is not tolerable)」である。

---

[20] システムの処理流れに関わるラショナルのことで、制度的な観点で論じるアカウンタビリティとは区別すべきである。

[21] ISO/IEC 25010:2023 で導入された品質特性である。

- 「コンピューティング安全性（Safety of Computing）」は、コンポーネントあるいはシステムの動的な処理過程に関わる。システム内部の振舞いに関わる性質であって、想定外に停止する（デッドロック）ことになるような処理進行を阻害するシステム状態（エラー状態）を避けることである。本ガイドラインでは、信頼性（4.2 節）の進行性（4.2.4 項）で取り扱う。
- 「コンピュータ安全（Computer Safety）」は、定められた条件下で作動しているシステムを対象とし、故障発生時に外部への危害が生じないように、システムをデザインすることである[22]。本ガイドラインの安全性は、コンピュータ安全の考え方を、信頼される AI に拡張するものである。
- 「AI セーフティ（AI Safety）」は、定められた条件から逸脱した状況下でのシステム動作を含み、広範なステークホルダーへの影響を論じる。影響に対する解釈がステークホルダーによって異なり、社会からみた AI リスクマネジメントの議論（1.3.1 節）と関わる。定められた条件から逸脱した状況下でのシステム動作に関わる AI セーフティは本ガイドラインの対象外である。

4.5.1　入力制限性

入力制限性（Input Constraint）[23]は、LLM 利用 AI システムが定められた条件下で作動する際に、危険な状態に至るような入力を制限することである。

- LLM 利用 AI システムへの入力プロンプトに対するフィルターの役割を果たす。一般には、LLM 利用 AI システムはオープンな利用が可能であり、プロンプトに課す条件を事前に規定することは難しい。当該 LLM 利用 AI システムの目的（要求）に応じて入力フィルター条件を実現する。

4.5.2　フェイルセーフ性

フェイルセーフ性（Fail Safe）は、LLM 利用 AI システムが故障（Failure）発生に際して、外部への危害が生じるような不適切な出力を行わないことである。

---

[22] 「ディペンダブルシステムのソフトウェア（Software of Dependable Systems）」が持つべき品質特性として議論されてきた安全性のことである。また、ISO/IEC 25010:2023 の Annex B では、 IEC 60050-192 の「ディペンダビリティ（Dependability）」マッピングを整理し、両者の安全性が対応すると解釈している。つまり、本ガイドラインの安全性はディペンダビリティに必須の品質特性といえる。

[23] ISO/IEC 25010:2023 の運用制限を入力制限性とした。

- 故障に関連する用語は、耐故障性（4.2.6 節）に示した定義を参照のこと。
- 外部への危害は、LLM 利用 AI システムの出力コンテンツが原因となって表面化する。したがって、故障発生時に出力コンテンツの劣化を許容範囲に抑えるか、あるいは、期待するコンテンツ生成が不可能な旨を出力する。
- アクセス制御性（4.4.1 節）と関わる。

4.5.3　否認防止性

否認防止性（Non-repudiation）は、LLM 利用 AI システムの出力コンテンツが、機械学習 AI の技術を用いて生成されたことを確認できることである [24]。

- コンテンツ生成過程で機械学習 AI の技術が用いられたことの明示を、AI 技術利用の透明性と呼ぶことがある。否認防止性は、AI 技術利用の透明性に関わる技術的な方策である。対象となるコンテンツデータの特徴によって、透かしやメタ情報利用の方法がある。しかし、利用者が透かしやメタ情報を削除することを確実に防止できる方法は知られていない。

4.5.4　特記事項

安全性 (Safety) は古くからあるトピックである。品質項目としても様々な分野で取り上げられている。電気製品、自動車、医療機器など、分野によっては、独特のマネジメント手法が確立していることがある。一方、ソフトウェアの品質標準 SQuaRE では近年まで安全性という品質項目が取り上げられていなかったが、2023 年の改定で Safety が追加された。生成 AI 以前の識別 AI や予測 AI の安全性に関しては、産総研のガイドライン [AIQMv4] が「リスク回避性」として大きく取り上げて、そのマネジメントを行うための考え方を示している。

しかし、生成 AI に関しては、「AI セーフティ (AI Safety) 」という名称で、従来よりも幅広い種類の問題が議論されている(表 1)。本書では 1.3.1 節に述べた通り、社会全体に影響するようなリスクではなく、個別の生成 AI がその利用者や第三者に与えるリスクのうち、生成 AI の開発や運用の中で技術的に対処しうるものだけを扱う。

上記のように扱う範囲を限定しても、生成 AI の「安全性」には、従来 AI では考慮する必要がなかったリスクが関わっている。例えば、個人情報や著作権で保護された情報

[24] 否認防止性は、本書における安全性の副品質特性と言えるかどうかには議論の余地がある。LLM 利用 AI システムの出力と関わる第三者に対するリスクに関する品質である。

が一目で分かるほど明瞭に出力に現れる恐れが挙げられる。これは、従来 AI に比べて生成 AI では出力の表現力が高いことに起因している。

出力に現れるどのような表現がどういう用途において問題となるかは、社会的な合意や文脈によって異なる。表現力の高い出力に起因するリスクとしては、公序良俗に反するものを中心に、様々な可能性が指摘されている。例えば LLM 評価のベンチマークとして「毒性（Toxiticy）」や「有害性（Harmfulness）」などの視点からのメトリクスが様々に用意されており、リスク検討のために参照することができる。しかし、それらのリスクの多くは生成 AI の用途によっては問題にならない。専門家を利用者とする合法的用途の中には、一般には有害とみなされる出力を必要とするものも考えられる。一方、年少者を含めた一般市民が使うことを想定する生成 AI ではそのような出力はリスクとして扱う必要があることが多い。したがって、出力に現れるどのような表現が安全性に関わるかは、多くの場合、システムの想定用途に基づいて開発者や提供者が判断する必要がある。

ただし、先に挙げた個人情報や著作権の問題には法規制があるため、かならず対処が必要になる。これらについては 4.6 節で別に取り上げる。

4.6 プライバシー

プライバシー（Information Privacy）は、パーソナルデータの取り扱いに関して、想定外の情報プライバシー漏洩が生じないことである。

- 機械学習 AI の品質観点としてのプライバシーに関わる事項に関しては、AIQM ガイドライン第 4 版の「プライバシー」関連項目を参照のこと。
- パーソナルデータの利用に関して、データ主体の同意を得ていることを前提とする。同意の方法は、準拠する規制法にしたがう。
- 情報プライバシー漏洩への技術的な対策は、データ匿名加工の方法で、データの利用性（5.1.7 項）を達成することである。データ保護強度への要請とデータ匿名加工を実施する技術的なコストとの関係から適切な方法を採用する。
- 情報プライバシーは、データ主体が自身に紐づくデータ（パーソナルデータ）の第 3 者による利用を自己管理する「利用制御」と関わる。「利用制御」は、アクセス制御性（4.4.1 項）が関わる利用許可に加えて利用責務を組み合わせた考

え方である（5.1.7 項）[25]。

- 「利用制御」は、著作権者に紐づくデジタル資産に対しても論じることができる。LLM 利用 AI システムを含む生成 AI では、LLM 訓練に用いられた学習データが著作権にしたがうデータを含むとき、LLM 内部知識として保持されている当該データの一部が生成コンテンツに含まれることがある。その出力の利用の仕方によっては、原著作権者の権利（著作権および関連する権利）を侵害する恐れがある。

4.7 公平性

公平性（Algorithmic Fairness）は、要配慮属性を含むデータの取り扱いに関して、想定外の偏りが生じないことである。

- 機械学習 AI の品質観点としての公平性に関わる事項に関しては、4.7.1 節の特記事項に留意の上、AIQM ガイドライン第 4 版の「公平性」関連項目を参照のこと [26]。
- 要配慮属性は、特定の社会集団に属する個人（データ主体）を特徴つけるデータ属性を指す。公平性は、要配慮属性の値の違いによって LLM 利用 AI システムの出力コンテンツが影響を受ける状況を扱う。特に、要配慮属性の値によって、出力コンテンツに生じる偏りを「結果バイアス」と呼ぶ。
- 与えられたデータが明示的に要配慮属性を持つ場合と、表層的に要配慮属性を持たなくてもデータ処理過程で要配慮属性に相当する情報（プロキシ情報）が生じることで、LLM 利用 AI システムの出力コンテンツに想定外の結果バイアスが顕在化することがある。

4.7.1 特記事項

従来型 AI（識別 AI、予測 AI）を対象とした第 4 版[AIQMv4]で述べている公平性に関する品質管理のうち、品質担保の基本的な考え方、「プロセス」（第 4 版 10.1.3 節 図 15）は、生成 AI に対しても適用可能である。 いっぽう、「中身・プロダクト」視点においては、具体的な要件や目標とする評価メトリクス（＝テストすべき事項）といっ

[25] 法律上の情報プライバシー権の中心は、明らかにされた「利用の目的」を基準として、データ利用の可否を論じることである。

[26] 一般に英語文献での「fairness」という用語は、「公正さ」あるいは「公平さ」のどちらかを指す。その語が用いられている文脈を考慮して適切な解釈を行うことが大切になる。

た面で、様相が異なってくる。識別 AI において公平性は、たとえばローン審査可否、入試合否など何らかの「判定」を AI が行うにあたって、その判定が性別や人種等が異なる集団間で不平等に扱われるかが焦点となる。また、第 4 版で詳細に述べている「結果の公平性」「取り扱いの公平性」や、対応する具体的なメトリクス（Demographic Parity や Equal Opportunity）も、基本的には明確に定義可能な AI の機能（判定）を前提としている。しかし、生成 AI においては、その多彩な出力は、判定とは全く異なる場合も多く、公平性を考えるうえでも異なる視点を要することになる。

LLM が脚光を浴びる前から、StereoSet など、言語モデル中に存在するステレオタイプの度合を評価するためのデータセットは提案されており[StereoSet]、最近では様々な LLM の性能を比較するのに使われるリーダーボードにおいても、例えば Nejumi リーダーボード 3 [Nejumi]では、バイアス QA データセットである BBQ（Bias Benchmark for QA）を日本語化した JBBQ にて点付けがなされている。いずれの場合も、これら LLM のバイアス評価に使われるテストデータセットは、「あらかじめ選択・定義した各種ステレオタイプ」が、どれぐらい回答に現れるかを確認するための質問と回答群から構成されている。

しかしながら、 LLM を組み込んだ生成 AI システムを開発する場合、その用途・機能に対して、前述の事前定義した各種ステレオタイプ in LLM のみで起こり得る問題がカバーされるかは全く保証できない。例えば、ジェンダーに関する倫理的な評価を行うデータセットを用意して LLM を評価した際に、一般的な質問（例えば、男女どちらが家事をすべきか？）に対してはおおむね問題ない出力をするが、それから少し外れた質問（例えば、男女どちらが○○の書類を提出すべきか、など実際のビジネスシーンに即した質問）に対しては問題ある回答が多くなることがあるのが知られている。当該システムの使われ方に基づき確認すべきステレオタイプの追加検討のみならず、LLM 入力の前処理や、出力への後処理を含めたシステム全体においての公平性視点を盛り込んだ設計が必要になる。

4.8 性能効率性

性能効率性（Performance Efficiency）は、LLM 利用 AI システムが、想定された状況下で作動する度合いに応じて、期待する量の計算資源を使用することである。利用する LLM 作動に関わる計算資源が大きく影響する。

4.8.1 時間効率性

時間効率性（Time Behavior）は、LLM 利用 AI システムが機能を実行するとき、期待通りの応答時間・処理時間・スループットを示すことである。

4.8.2 資源効率性

資源効率性（Resource Utilization）は、LLM 利用 AI システムが機能を実行するとき、計算資源の量を期待通りに使用し、過剰な計算資源を使用しないことである。

4.9 移植性

移植性（Portability）は、LLM 利用 AI システムを、他の運用環境あるいは利用環境に移すのが容易であることである。

- LLM 利用 AI システムでは、クラウド環境あるいはオンプレミスでの作動を考えるかがデザイン上の決定事項に大きく影響する。
- 本ガイドラインでは、移植前後で、「機械学習-WITH 開発」の方策が変わらないとする。特に、移植の前後で共通して「文脈内学習を基本とするプロンプトエンジニアリング」手法を用いることを想定する。
- 移植のユースケースとして、移植前は「文脈内学習を基本とするプロンプトエンジニアリング」手法を用いる一方で、移植後の LLM 利用 AI システムを「ファインチューニング」手法によって実現する場合がある。これは、移植前を「要求仕様を満足する作動可能なデザイン仕様」とし、このデザイン仕様を再現する方法として「ファインチューニング」手法を用いると見なせる。つまり、新たな LLM 利用 AI システム開発であり、移植性の範囲を越える。

4.10 保守性

保守性（Maintainability）は、期待される状況で、保守者が LLM 利用 AI システムを進化発展させることの容易さである。ソフトウェア保守の作業は、一般に、開発時に混入した欠陥の修正と開発時に想定されていなかった新たな要求を満たす改良に分ける。ここでは、後者の完全化保守・適応保守を容易にするデザインが関わる。

- 完全化保守は、性能効率性（4.8 節）や適応保守に関わる保守性の向上を目的とする。LLM 利用 AI システムでは、プロンプトデザインに関わる改良を含む。

コンポーネントを実現するプログラムのリファクタリングを含む。

- 適応保守は、要求の変化を含むシステム利用環境の変化に合わせて、システムの継続使用を目的とする。機械学習 AI の場合、実行時入力データの分布シフトへの対応を含む。LLM 利用 AI システムでは、再利用部品である LLM の置き換えによる改良を含み、移植性（4.9 節）と関わる。

4.10.1 モジュール性

モジュール性（Modularity）は、LLM 利用 AI システムを構成するアーキテクチャコンポーネント互いの依存関係を明らかにすることである。

4.10.2 修正性

修正性（Modifiability）は、LLM 利用 AI システムならびに構成コンポーネントの修正が容易なことである。

4.10.3 試験性

試験性（Testability）は、LLM 利用 AI システムならびに構成コンポーネントの試験が容易なことである。試験項目（試験条件・試験基準）のデザインが容易なことを含む。

5 データ品質

LLM 利用 AI システムでは、「データ」がさまざまな形で現れる。「機械学習-FOR 開発」では、モデル訓練に用いる学習データ（訓練データ）の品質が、機能部品としての LLM（訓練済みモデル）に大きく影響する。一方、「機械学習-WITH 開発」や「ソフトウェア-WITH 開発」で取り扱うデータは、訓練データだけではない。LLM 利用 AI システム開発に関わるデータの品質を包括的に取り扱う。

「データ品質」は、機械学習や LLM 利用 AI システム特有の技術的な性質を加味し、品質を論じる対象によって 3 つに分けて整理した。

- 個々のデータ点
- データセット（データの集まり）
- 複数データセットの組み合わせ利用

以下、品質の対象に沿って、順番に説明する。

5.1 個々のデータ点の品質観点

個々のデータ点の品質は、SQuaRE の「データ品質モデル（JIS X 25012:2013）」から品質観点を選び、機械学習や LLM 利用 AI システムを開発する上で重要な性質を整理したものである。ここで、データ点として以下を想定する。

- データ点は処理の単位であり、離散データか、不定長の系列データ（時系列、トークン列）か、を区別しない。
- 素材としての「データ」と機械学習や LLM 利用 AI システムの処理対象としての「データ」の両方を対象とし、処理対象とする「データ」が満たすべき品質観点を整理する。

LLM 利用 AI システムの品質マネジメントには以下の点で関わる。

- 対象「データ」の処理目的に応じて、品質マネジメントの取り組みが異なる。
- 教師あり学習で用いる学習データに関しては、AIQM ガイドライン第 4 版[AIQMv4]の内部品質 B-3（データの妥当性）のデータ点の品質観点が対応する。「機械学習-WITH 開発」のファインチューニングに用いる「データ」の取り扱いを含む。
- LLM 利用 AI システムが RAG(2.1.3 節、1.5 節)を用いる場合、RAG で参照する外部データの取り扱いを含む。

- プロンプトデザインによって、データの役割を担うプロンプト段落を導入する場合、当該プロンプト段落のデータの取り扱いを含む。

5.1.1 正確性

正確性（Accuracy）は、構文上（形式上）ならびに意味上、正しい表現を持つことである。素材データが正確性に劣るとき、事前作業を適切に施して、正確性を満たす処理対象データを得る。

- 敵対データは、訓練済みモデルを混乱させる目的で、意味上の正確性を意図的に劣化させた処理対象データである。正確性を満たす基準データに対して、セマンティックノイズを系統的に挿入する方法が知られている。基準データと敵対データを訓練済みモデルに入力した場合の各々の出力を比較することで、当該モデルの敵対ロバスト性（5.2.2 項）を評価することができる。
- 大規模言語モデルの処理対象となる系列データ（文字列データ）では、自然言語の文字列（ここでは自然言語文と呼ぶ）を対象に考える。自然言語文の形式上の正確性は、当該自然言語の文法規則が規定する。自然言語文の意味上の正確性は、その文の意味内容から判断される。

5.1.2 完全性

完全性（Completeness）は、データ点が複数の属性から構成される場合、すべての属性が欠損していないことである。素材データが完全性に劣るとき、事前作業を適切に施して、完全性を満たす処理対象データを得る。

- データ点をレコードとみなす場合、属性をフィールドと呼ぶ。多次元ベクトルとみなす場合、成分と呼ぶ。系列データの場合、系列の要素をさす。この注意は、以下の副特性（5.1.3 節から 5.1.8 項）に共通する。
- 完全性を満たさないデータを欠損データと呼ぶ。完全性を満たすデータを基準とするとき、基準データと欠損データを関係つけることができる。基準データと欠損データを訓練済みモデルに入力した場合の各々の出力を比較することで、当該モデルの欠損ロバスト性を評価することができる。
- 系列データは、自然言語文データを含む。系列データの完全性は、要素が欠落していないことである。特別な場合として、マスク言語文データがあるが、マスクであることを表す要素がマスク言語文データを構成することから、マスク言語文データは完全性を満たす。

5.1.3 一貫性

一貫性（Consistency）は、データ点が複数の属性から構成される場合、すべての属性の値が整合していることである。素材データが一貫性に劣るとき、事前作業を適切に施して、一貫性を満たす処理対象データを得る。

- 整合しているとは、矛盾しないことであり、自己整合と比較整合の 2 つの場合がある。
- 自己整合とは、当該データ点の複数の属性が意味的に関連するとき、それらの属性値が意味的に矛盾しないことである。
- 比較整合とは、類似する他のデータ点と比較して、同じ属性が持つ値が矛盾しないことである。ここで、2 つのデータ点が類似するとは、着目する属性以外を除去した場合、一貫性を満たすことである。
- 分布外データ（Out-of-Distribution）は自己整合の観点で一貫性が劣る。

5.1.4 信憑性

信憑性（Credibility）は、データ点が複数の属性から構成される場合、すべての属性の値が、データ利用者にとって、真とみなせることである。素材データが信憑性に劣るとき、当該データを削除する等の事前作業を適切に施して、信憑性を満たす処理対象データを得ることが望ましい。

- 信憑性は、真正性（偽造データでないこと等）を含む。
- 敵対データや分布外データは真正性を満たさない。

5.1.5 最新性

最新性（Currentness）は、データ点が複数の属性から構成される場合、すべての属性が最新の値になっていることである。素材データが最新性に劣るとき、当該データを削除する等の事前作業を適切に施して、最新性を満たす処理対象データを得ることが望ましい。

- 最新性は、実世界の時間で特徴つけることではなく、バージョン概念と関わる。
- 更新されるなどの理由で、属性値が複数のバージョンを持つとき、属性ごとに最新バージョンの値を持つことである。データ点を構成する属性値のバージョンが整合していることと言って良い。

5.1.6 精度

精度（Precision）は、データ点が複数の属性から構成される場合、すべての属性の値が期待される数値精度をもつことである。処理対象データに対して定義される。

- データが表す情報の種類に応じて適切な方法で精度を評価する。
- LLM 利用 AI システムでは、外部表現としての自然言語文字列を処理可能なデータに変換する「トークン化（Tokenizers)」や「埋め込み（Embeddings)」の選択と関わる。
- 画像データでは、解像度が関わる。
- 時系列信号データでは、スペクトルの数値精度が関わる。

5.1.7 利用性

利用性（Data Usage Control）は、データ点が複数の属性から構成される場合、すべての属性の値が利用承認されていることである。処理対象データに対して定義される。

- 所与のデータに対する事前処理によって、利用性を満たすデータに加工する。さまざまなデータの加工方法（属性の削除・集約・仮名化など）が知られている。データの利用目的に応じて、適切な方法を選択する。
- データ加工の方法として、データ暗号化の技術が採用されることがある。
- データ加工は、プライバシー（4.5 節）や公平性（4.7 節）と強く関わる。
- データ処理に関わる利用承認は、データの利用制御と関わり、予め定められた目的から見て正当な利用許可と利用責務として付与される。利用承認は処理対象データの権利者による決定にしたがって付与される。つまり、データ利用制御に関わる決定権を尊重する。
- 利用許可は、定められたアクセス制御条件を規定することであり、アクセス制御性（4.4.1 節）と関わる。利用責務は、利用許可されたデータに対して、定められた利用処理の種類を規定することである。
- 著作権､情報プライバシー権[27]などの法規制が関わるデータは標準適合性(5.1.8 節）にしたがう。

[27] 情報プライバシー権の中心は、明らかにされた「利用の目的」を基準として、データ利用の可否を論じることである（4.6 節)。

5.1.8 標準適合性

標準適合性（Compliance）は、データ点が複数の属性から構成される場合、すべての属性の値が規則に適合していることである。処理対象データに対して定義される。遵守する規則は、公的なガイドライン・規格・規制法などがある。

5.2 データセットの品質観点

データの集まりをデータセットと呼ぶ。LLM 利用 AI システムでは、「データセット」がさまざまな形で現れる。「機械学習-FOR 開発」や「機械学習-WITH 開発」のファインチューニングでは、モデル訓練に用いる学習データセット（訓練データセット）の品質が、機能部品としての LLM（訓練済みモデル）に大きく影響する。また「機械学習-WITH 開発」のプロンプトエンジニアリングや「ソフトウェア-WITH 開発」では、プロンプトで取り扱う「データ」を「データセット」として品質を論じると有用なことが多い。ここで、LLM への入力プロンプトを構成する次のような手法の中で「データ」が現れる。

- プロンプトエンジニアリングの少数例示学習(A1.3.1 節)による手法
- 実行時の情報検索により取得した外部データを挿入する RAG 手法

以降の「データセットの品質観点」では、LLM 利用 AI システム開発に関わるデータセットの品質を包括的に取り扱う。

LLM 利用 AI システムの品質マネジメントには以下の点で関わる。

- データセットの品質は、一般に統計的なデータ分布が良い指標となる。データ分布を数学的な対象として明示することで、データセットの品質を論じる。
- 機械学習のモデル訓練や LLM 利用 AI システム開発では、データ分布を明示することができない場合、データあるいはデータセットのデザインから、品質の問題を整理する。
- デザインの観点から、膨大なデータをカテゴリーに区分けし、各カテゴリーから抽出するデータを決定する指針を明確にする。

5.2.1 代表性

代表性（Representativeness）は、問題分析からデータをカテゴリーに区分けすることを前提とするとき、カテゴリーごとに適切なデータを抽出し、全体を合わせてデータセットにすることである。代表性の基準となるカテゴリー分けは目的依存で決定する。

5.2.2 重複性

重複性（Overlap）は、問題分析からデータをカテゴリーに区分けするとき、カテゴリーと隣接カテゴリーの間で、データの一部を共有することである。冗長性（Redundancy）ともいう。なお、同一のデータが繰り返しデータセット中に出現することは、データ分布（出現頻度）の問題であり、品質副特性としての重複性・冗長性とは異なる。

5.2.3 標本選択性

標本選択性（Sample Selection）は、取り扱うデータ全体の経験分布に関わる情報があるとき、この経験分布を基準としたカテゴリーに対して、経験分布を再現するように、データを選択することである。データバイアスが小さいことである。

5.2.4 一貫性

一貫性（Dataset Design Consistency）は、データセットが含むデータに対して、期待する品質（5.1 節）を満たす処理が、定められた方針（デザイン）の下に実施されていることである。

LLM 利用 AI システムの品質マネジメントとは以下の点で関わる。

- 個々のデータ点の品質観点（5.1 節）がデータセット全体に反映されることを保証する。

5.2.5 来歴性

来歴性（Provenance）は、データセットが、信憑性・真正性を満たすデータ（5.1.4 節）から構成されるとき、当該データセットが持つ性質である。

- 来歴性は、成果物管理のメタ情報として、データセットに付加される。素材デ

ータの入手・収集方法、データセット構築者に関わる情報、などを含む。

5.2.6 最新性

最新性（Dataset Newness）は、データセットが、最新性を満たすデータ（5.1.5 節）から構成されることである。

- 最新性は、成果物管理のメタ情報として、データセットに付加される。

5.3 複数データセット組み合わせ利用時の品質観点

複数のデータセットを組み合わせて利用するとき、全く無関係なデータセットであれば、単に和集合をとるだけで良い場合がある。データセットが何らかの特徴を共有する場合は、メタ情報によって、データセットを組み合わせる目的に対して整合しているかを確認する。

LLM 利用 AI システムの品質マネジメントとは以下の点で関わる。

- 教師あり学習で、独立に整備された複数のデータセットを組み合わせてモデル訓練に用いる場合に、組み合わせ利用時の品質観点を調べる。
- LLM 利用 AI システムでは、外部参照データの知識（Non-parametric Knowledge）と、当該 LLM が内部保有する知識（Parametric Knowledge）との関係が、組み合わせ利用時の品質観点と関連する。どちらを優先するかの「制御」を明示的に検討する。
  - RAG を用いる場合、RAG で参照する外部データの知識と、当該 LLM が内部保有する知識との関係が、組み合わせ利用時の品質観点と関連する。
  - プロンプト内で参照する知識と、当該 LLM が内部保有する知識との関係が、組み合わせ利用時の品質観点と関連する。

5.3.1 文脈整合性

文脈整合性（Contextual Adequacy）は、組み合わせる複数のデータセットが利用の目的に対して整合していて、データセットを統合して処理できることである。データ準備に関わる観点と目的整合に関わる観点がある。

- データ準備に関わる観点は、素材データの収集方法・処理データの整備方法などがある。
- 目的整合に関わる観点は、データセット利用の目的が共通していることである。

5.3.2 時制整合性

時制整合性（Temporal Adequacy）は、同一対象の異なるバージョンのデータセットが存在するとき、バージョンの組み合わせが利用の目的に対して整合していることである。

5.3.3 操作整合性

操作整合性（Operation Adequacy）は、組み合わせる複数のデータセットが利用の目的に対して、データセットを同時操作できることである。

6 LLM 利用 AI システムの品質実現に向けた管理策

本章では、第 4 章に挙げたコンポーネント品質を LLM 利用 AI システムが満たすことを目的として行う品質マネジメント上の施策（管理策）を示す。

本節の構成は、2.1.3 節に示した本書で想定する LLM 利用 AI システムの構成に沿っており、まずシステム全体について行うべき管理策を挙げた後、システムを構成する主要なコンポーネントごとに行うべき管理策を示す。システム全体または主要コンポーネントごとの節の中では、LLM 利用 AI システムのコンポーネント品質ごとに、それを実現するための管理策を示す。また、各コンポーネントに関わりの深いデータについて、第 5 章に挙げたデータ品質を実現するための管理策を合わせて示す。

6.1 システム全体

LLM 利用 AI システムの品質実現に向けて、システム全体で行うべきことは、以下のように大別できる。

- コンポーネント間のオーケストレーションの設計
- システム外の要素への対処の設計
- システム全体としての利用時品質の評価

一般に、コンポーネント間のオーケストレーションが複雑なシステムでは、システム全体としての進行性、可用性、耐故障性、回復性、可制御性、介入性などを満たすために、システム全体のアーキテクチャを慎重にデザインする必要がある。LLM という、ブラックボックスで振舞いが不確定なコンポーネントを含むシステムではよりそれが重要になる。

またシステム外の要素、たとえばユーザーや、プラグイン経由で呼び出す外部サービスなどの素性や特性を考慮した上で、対処するべき事項があればシステムとしてどう対処するかを設計時に決める必要がある。

さらに、システム全体の利用時品質はシステム全体としてでなければ実施できない。コンポーネントごとに実施した管理策によって、要求を満たす品質が実現されているかどうかを確認する。

LLM 利用 AI システムが満たすべき品質ごとに、その品質の達成に向けてシステム全体に関して実施するべき事項を以下に挙げる。

6.1.1 要求満足性

- 機能要求満足性
  - LLM 利用 AI システムの明示された要求仕様を基準として、当該システムへの入出力から機能要求満足性を評価する。
  - マーケティング等で実施されるアンケートによる定性的な満足度評価手法を生成コンテンツに対して用いることが考えられる。
- 品質要求満足性
  - 当該 LLM 利用 AI システムが持つ品質について、システムの目的から見た完全さ・正確さ・適切さから評価し、システム全体の品質要求満足性を整理する。

6.1.2 信頼性

- 成熟性
  - 結合テストに相当するソフトウェアテスティングで、システムが所与の機能仕様を適切に実現していることを検査し成熟性を評価する。
  - 所与の機能仕様を満たす入力（期待する基準データ）によって検査を行う正テスティングによる確認を実施しなければならない。
- 進行性
  - 機能要求の実現に関わるデザインと、進行性に関わるデザインを分離して実施する。
  - オーケストレーションを司る自律エージェントでは、進行性を満たすことを確認しなければならない。特に、出力が得られることを確認しなければならない。
- 可用性
  - LLM 利用 AI システムが作動する動作環境を適切に設定し安定稼働させる。
- 耐故障性
  - 偶発的な欠陥が生じる可能性のあるコンピューティング基盤上で作動する状況下で、当該の欠陥がシステム停止に波及しないようにデザインする。LLM コンポーネントの復旧施策を講じる。
  - 決定論的な欠陥への対策が為されていることを、成熟性の評価によって確認しなければならない。
- 回復性
  - 故障（4.2.6 節）に至って、LLM 利用 AI システムが実行中の処理を継続で

きなくなったとき、LLM 周辺コンポーネントのデザインによって、処理の中間点（チェックポイント）への戻りなどのデザイン上の対策を講じる。
  - 再入可能な計算コンポーネントである LLM は初期状態に戻す。

6.1.3 インタラクション性

- アクセシビリティ
  - 自然言語インタフェースが主となる LLM 利用 AI システムでは、入力ならびに出力コンテンツを表す言語について、エンドユーザーが習得している、あるいは使っている言語の違いを考慮してデザインする。たとえば、多言語対応のデザインを行う
- 適切度認識性
  - エンドユーザーが判断できるように、LLM 利用 AI システムに付す「使用説明（Instruction for Use）」「利用制限（Behavioral Restrictions）」といった文書を提供しなければならない。
- 説明性
  - 正当化理由を提示できるよう、システムプロンプトや関連するコンポーネントをデザインする。

6.1.4 セキュリティ [28]

- 真正性
  - LLM への入力プロンプトフロー（当該プロンプト発行元）が正当なことを確認する。ユーザーや、プラグインを介して利用する外部サービスが該当する。
- 介入性
  - 外部からの指示によって、作動中であっても構成コンポーネントが停止し、その後、システムの実行再開を可能にする状態へ復帰するようにデザインする。
- 責任追跡性
  - LLM 利用 AI システムを構成するコンポーネント間の情報の流れをログとして保存する機能をデザインする。
  - ユーザーがログ情報を利用できる機能をデザインする。

[28] 4.4 節の注 18 を参照のこと。

6.1.5 安全性

- 入力制限性
  - エンドユーザーに課す利用条件の一部に、選定した LLM に付随する情報が示す制限条項の遵守を明示する（6.3.2 項を参照）。

6.1.6 性能効率性

- 時間効率性
  - システムテストに相当するソフトウェアテスティングで、システムが所与の機能仕様を適切に実現していることを検査し時間効率性を評価する。
- 資源効率性
  - システムテストに相当するソフトウェアテスティングで、システムが所与の機能仕様を適切に実現していることを検査し資源効率性を評価する。

6.1.7 移植性

- LLM 利用 AI システムのコンポーネントアーキテクチャは、再利用部品である LLM の置き換えが容易になるようにデザインする。LLM を仮想化する「言語モデル」コンポーネントを導入する。

6.2 コンポーネント全般

LLM 利用 AI システムが満たすべき品質ごとに、その品質の達成に向けて各コンポーネントに関して共通して実施するべき事項を以下に挙げる。

6.2.1 信頼性

- 成熟性
  - LLM 利用 AI システムのデザインの結果として得られた当該コンポーネントの機能仕様を適切に実現していることを検査し成熟性を評価する。
  - LLM 利用 AI システムを構成する全てのコンポーネントを対象とした単体テストに相当する。
  - ブラックボックステスティングによる確認を実施しなければならない。
  - 必要に応じてホワイトボックステスティングによる確認を実施する。
- ロバスト性

- 機械学習モデル特有のモデルロバスト性とは別に、プログラムロバスト性をすべてのコンポーネントで満たすように実現する。

6.2.2 保守性

- モジュール性
  - 本ガイドラインが前提とするコンポーネント中心デザインは、モジュール性に寄与する。
  - LLM 利用 AI システムの概念的な情報の流れ（2.1.1 節）にしたがって、LLM 周辺コンポーネントに明確な役割・責任（Responsibility）を与えることで、当該コンポーネントのモジュール性が高まる。
  - コンポーネントのデザインでは、入出力ポートの定義を明示する。
- 修正性
  - モジュール性（4.10.1 節）は、モジュール内に閉じた修正を可能にすることから、修正性に寄与する。
  - LLM 利用 AI システムを構成するアーキテクチャコンポーネントを明示することで、モジュール単位での追加・削除・置き換えに関わる修正性を向上させる。
  - アーキテクチャのデザインでは、トップレベルデザインからの追跡性を向上させることを目的としたパターン生成アーキテクチャのデザイン方法論などを活用する。
- 試験性
  - モジュール性（4.10.1 節）は、個々のモジュールに対して試験を行うことから、試験性に寄与する。
  - コンポーネントの入出力ポート定義をもとに、試験項目をデザインする。
  - LLM 利用 AI システムを構成するアーキテクチャが追跡性を満たすときようにすることで、階層的に試験対象を構成するようにする。

6.3 LLM の選定と追加学習

本書の想定では、LLM には第三者が開発済の基盤モデルを再利用部品として用いる。このため、LLM の開発時に行う管理策は取り上げず、用いる LLM の選定に関わる管理策を示す。

LLM 利用 AI システムが満たすべき品質ごとに、その品質の達成に向けて利用する LLM の選定と追加学習に関して実施するべき事項を以下に挙げる。

6.3.1 信頼性

- ロバスト性
  - モデルロバスト性は、訓練済み学習モデルに固有の技術的な性質であり、LLM 利用 AI システムでは、所与の再利用コンポーネントとして選択した LLM で決まる。LLM に付された情報（AI BOM 等）をもとにして、当該 LLM 利用 AI システムの目的（要求）と整合するかを事前に判断する。
- 可用性
  - エンドユーザーによる利用の制限条項 [29]を遵守しても LLM 利用 AI システムの機能の実現に支障がないことを確認する。制限条項に違反する利用を試みると、LLM はしばしば回答を拒否し、またネット上のサービスとして提供されている LLM の場合、利用権限が取り消されることもある。

6.3.2 安全性

- 入力制限性
  - 不適切な入力に対する振舞いを制限するように適切にアラインされているかを確認する。LLM に付された情報（AI BOM 等）をもとにして、当該 LLM 利用 AI システムの目的（要求）と整合するかを事前に判断する。また、 エンドユーザーに課す利用条件の一部に、選定した LLM に付随する情報が示す制限条項の遵守を明示する。

6.3.3 プライバシー

- 情報プライバシー権や著作権にしたがって利用制御されているデータをモデル訓練に用いる場合、利用制御に関わる制度上の問題を解決済みの LLM を選ぶ。AI BOM 等の LLM に関わる情報を参照して、適切な LLM を選定しなければならない。

6.3.4 公平性

- 要配慮属性を含むデータをモデル訓練に用いる場合、バイアスに関わる制度上の問題を解決済みの LLM を選ぶ。AI BOM 等の LLM に関わる情報

[29] LLM に付随するライセンス項目（Behavioral Usage Restrictions や User-based Restrictions）ならびに使用説明（Instructions for Use）を参照する。

を参照して、適切な LLM を選定しなければならない。

6.3.5　性能効率性

- 時間効率性
  - AI BOM 等の LLM の時間性能に関わる情報を参照して、適切な LLM を選定しなければならない。
- 資源効率性
  - 選定した LLM によって必要な計算資源が決まることから、AI BOM 等の LLM の資源効率に関わる情報を参照して目的に対して適切な規模の LLM を選定する。

6.3.6　移植性

- LLM は、同一の開発組織によるシリーズ版であっても学習パラメータ規模によって能力が異なり、同一プロンプト入力に対して、同じ機能振舞いを保証するわけではない（機能振舞いの揺らぎ）。LLM への入力に関わるプロンプトならびに出力フィルターに関わるデザインと組み（トリプレットパターン）にして移植性を評価しなければならない。
- 「LLM 機能振舞いの揺らぎ」が理由となり、プロンプト等を工夫しても、LLM 置き換え前後で同じ出力を得られるわけではない。要求満足性（4.1 節）から再評価しなければならない。

6.3.7　学習データ品質

本書の想定では、基盤モデルは他組織で開発されたブラックボックスの再利用部品であり、その学習データの品質確保に直接関わることができないが、学習データの品質に関して充分な情報が開示されており、それによって学習データの品質が確保されていたことが確認できるものを選んで用いることができる。

また、本書の想定においても、基盤モデルをファインチューニングする際に使う学習データの品質確保には関わることができる。

6.3.7.1　個々のデータ点の品質観点

教師あり学習で用いる学習データに関しては、AIQM ガイドライン第 4 版の内部品質 B-3（データの妥当性）のデータ点の品質観点が対応する。「機械学習-WITH 開発」のファインチューニングに用いる「データ」の取り扱いを含む。

- 完全性
- 正確性
- 一貫性
- 信憑性
- 最新性
- 精度
- 利用性
  - 機微情報を含むデータに対して、期待される保護強度を示すデータ加工（データ匿名加工）を施す。データの利用目的との整合性からデータ匿名加工の方法を選定しなければならない。
- 標準適合性
  - 所与のデータに対する事前処理によって、標準適合性を満たすデータに加工する（5.1.8 節）。さまざまなデータ匿名加工の方法（集約・仮名化など）が知られている。遵守する規則に応じて、適切な方法を選択する。

6.3.7.2　データセットの品質観点

教師あり学習で用いる学習データに関しては、AIQM ガイドライン第 4 版の「データセットの被覆性（B-1）」を参照する。

- 代表性
  - 「データセットの被覆性（B-1)」は「対応すべき状況の組み合わせ」を基準としたカテゴリーに対して代表性を満たすデータセットを得る。
  - 稀な状況の取り扱いに関心が高い場合、外れ値をカテゴリーに含め、当該カテゴリーからデータを積極的に抽出する。「データセットの被覆性（B-1)」の特別な場合になる。
- 標本選択性
  - データ全体の経験分布を基準としたカテゴリーに対して代表性を満たすデータセットを得る。
  - データ全体の経験分布以外の方法を基準としてカテゴリー分けされていて、カテゴリーごとに固有の経験分布が既知の場合、代表性を満たすと同時に、カテゴリー固有の経験分布からの標本選択性を満たしたデータセットを得る。
- 一貫性
  - 教師あり学習で用いる学習データに関しては、AIQM ガイドライン第 4 版の「データの妥当性（B-3)」の「ラベリングの適切性」と関わる。データの構築に際して、データセットを通じて一貫した方針の下にラベリングを行う。

- 最新性
  - データセットをモデル訓練に利用する際に、事前に、当該データセットの最新性を調べて、利用目的と整合していることを確認する。

6.3.7.3　学習データセットの利用時品質観点

教師あり学習で、独立に整備された複数のデータセットを組み合わせてモデル訓練に用いる場合に、組み合わせ利用時の品質観点を調べる。考慮しうる組合せとしては、基盤モデルの開発時の訓練に使われたデータセット同士、基盤モデルの開発時のデータセットと追加学習に使うデータセットの間、さらに追加学習のデータセット同士がある。

- 文脈整合性
  - ファインチューニング（転移学習）では、ベースモデル訓練に用いた学習データセットと追加訓練に用いる学習データセットの関係が、学習目的と整合していることを確認する。
- 操作整合性
  - マルチモーダルモデルでは、データの種類ごとにデータセットを準備し、組み合わせて利用することがある。データセットごとに異なる処理が定義されている場合、組み合わせ操作を実現する。

6.4　プロンプトとその周辺

プロンプトは組成としても役割としても多様な側面を持つ要素であり、これに関わるコンポーネントは具体的な LLM 利用 AI システムごとの違いが大きいと考えられるが、本書で想定する LLM 利用 AI システムにおける、プロンプトに関わる主な事項として以下を本節で取り上げる。

- システムプロンプト
- プロンプト補完コンポーネント
- データとしてのプロンプト

6.4.1　システムプロンプト

システムプロンプトはテキストデータであるが、LLM に入力として与えることで、LLM の動作を規定するものである。それゆえ、本節では、システムプロンプトをその動作が実現する機能を持つコンポーネントと同一視してその役割を論じる。

LLM 利用 AI システムが満たすべき品質ごとに、その品質の達成に向けてシステムプロンプトに関して実施するべき事項を以下に挙げる。

- 機能要求満足性
  - プロンプトを入力した LLM の出力を検査することで、LLM 利用 AI システムの明示された要求仕様が規定する機能を LLM が実現可能かを調べる。
  - プロンプトと LLM の組み合わせを対象とした試行錯誤的な繰り返し（一種のプロトタイピング開発スタイル）を通して、機能要求満足性を評価する（4.2 節・4.4 節参照)。
- ロバスト性
  - ユーザー入力に揺れがあっても、LLM が同等なコンテンツを出力するように、その揺れを意味的に吸収するようなシステムプロンプトをデザインする。
- 進行性
  - プロンプトが進行性を満たさない記述になっていないことを、インスペクション等の文書レビュー手法により確認しなければならない。
- 説明性
  - LLM が出力を生成するために利用した情報を、出力の正当化理由として出力するようにシステムプロンプトをデザインする。LLM が利用する情報は、LLM が学習により取得したモデルに内在する情報である場合も、LLM への入力の一部として与えた情報である場合もある。
- アクセス制御性
  - LLM がシステムプロンプトを無視してユーザー入力に従うよう LLM に指示する(いわゆる jail break を試みる)ユーザー入力を無効化するようにシステムプロンプトをデザインする。Jail break が成功すると、LLM が訓練時に取得した情報や、運用時に LLM に入力された情報が出力に出ていく恐れがある。システムプロンプトだけで jail break を 100%阻止するのは難しいので、入力フィルター(6.7 節)や出力フィルター(6.8 節)等の他の対策を併用する。
- 入力制限性
  - ユーザーや外部サービスからの入力テキスト情報（プロンプト）を「データ」として取り扱うことで、入力フィルターの役割を果たすようにデザインする。
- 時間効率性、資源効率性
  - 利用する LLM の能力に応じた冗長さのないプロンプトをデザインする。LLM によって、ビジネスロジックを実現するためにプロンプトとして与える必要がある情報の詳しさが異なる。プロンプトが詳しいほどプロンプト

が長くなり、LLM による処理に時間がかかり、電力消費量も増える。

- 保守性
  - LLM 利用 AI システムの保守性を向上させるため、システムプロンプトのモジュール性を高めなければならない。プロンプトの流れを段落などの観点から構造化し各々の段落の役割・責任（Responsibility）を明確化する（3.4.1 節）。システムプロンプトのモジュール性は、段落間の関連が明確になることから、システムプロンプトの修正性にも寄与する。

6.4.2 プロンプト補完コンポーネント

プロンプト補完コンポーネントは、他の様々なコンポーネントから得られる情報をもちいてプロンプト全体を構築し、LLM に入力として与える。プロンプト補完コンポーネントがプロンプト構築に利用する主な情報としては以下が挙げられる。

- システムプロンプト
- ユーザー入力（いわゆるユーザープロンプト）
- ファイルデータ（RAG で用いるファイルデータベースにあるデータ）
- 外部連携コンポーネントからの応答

ユーザー入力や、外部連携コンポーネントなど、LLM 利用 AI システムの外から来る情報については、入力フィルター（6.5 節）を通した結果を利用する。

LLM にプロンプトを渡す際に、LLM の出力のサンプリング方式とそのパラメータを指定できることが多い。この方式とパラメータ値の選択は、LLM の出力が機能仕様を満たすかどうかに大きく影響し、どのような選択が有効かは、システムプロンプトに大きく依存する。このため、システムプロンプトの設計と合わせて検討する必要がある。

LLM 利用 AI システムが満たすべき品質ごとに、プロンプト補完コンポーネントで実施するべき事項を以下に挙げる。

- 機能要求満足性
  - LLM 出力のサンプリング手法やそのパラメータ（A1.1.3 節）を指定できる場合、システムプロンプトの設計のための試行錯誤の中で合わせて検討する。
- 説明性
  - ファイルデータや外部連携コンポーネントからの応答など実行時に選択してプロンプトに組み込む情報については、その来歴を示す情報を添えて組み込む。これと合わせ、システムプロンプトで来歴情報を活用するよう

指示する必要がある。

6.4.3 データとしてのプロンプト

プロンプトの役割は様々であるが、LLM に対する入力となるテキストデータであるので、データ品質を満たす必要がある。ただし、プロンプトを構成する各テキスト断片が担う役割に応じてより重視するべき品質観点は異なる。

6.4.3.1 LLM の処理を手続き的に指示する部分

LLM に期待する処理を、作業手順の記述として、または入力と出力の関係の記述として、指示するテキストについては、主に以下の品質が必要となる。

- 正確性
- 完全性
- 一貫性
- 利用性
  - 当該 LLM 利用 AI システムにおけるデータの利用目的から利用性を満たすか否かを決定しなければならない。

6.4.3.2 データや例示として与えられた部分

実行するべき機能を入力とそれに対し期待される出力の対の例示により示すテキストや、その機能で前提や参考情報として扱うべき情報を与えるテキストについては主に以下の品質が必要となる。

- 正確性
- 完全性
- 一貫性
- 信憑性、最新性
  - 当該データの来歴に関わるメタ情報を確認する。
  - データ量が膨大になると、信憑性、最新性を満たさないデータが混入する可能性が大きくなることに注意しなければならない。
  - 生成 AI が出力したデータコンテンツデータは信憑性に劣る場合を排除できないことから、そのようなデータの利用には注意しなければならない。
- 精度
  - 「トークン化（Tokenizers）」や「埋め込み（Embeddings）」の選択に注意しなければならない。例えばトークン化が多言語に対応していないと、対応言語以外の言語について不適切なトークン化が行われる（A1.2.1.4 節）。

- 利用性
  - 当該 LLM 利用 AI システムにおけるデータの利用目的から利用性を満たすか否かを決定しなければならない。
  - 情報プライバシー・公平性が関わる機微情報を表すデータの利用性は特に注意しなければならない。

6.4.3.3　データセットの例示として与えられた部分

プロンプトの中に与えられた同一の事項に関する複数の例示が、データセットとして主に満たすべき品質を以下に示す。

- 代表性
  - 例示対象の問題分析を行なって区分けしたカテゴリーを網羅するように、各カテゴリーから具体例をひとつずつ抽出し、全体を仮想的なデータセット（デモセットと呼ばれることがある）を得る。
- 重複性
  - 問題分析の結果であるカテゴリー間での同一情報の共有を避けることで、有用性を高める。つまり、重複性を避ける。
- 一貫性
  - 一貫した方針のもとに、例示に含む情報を決定する。
- 来歴性
  - LLM 利用 AI システムの外から運用時に取得した情報に由来するデータセットについては、来歴を示せるようにする。
- 時制整合性
  - 最新データの役割を果たすプロンプト段落は、一般には、内部知識が矛盾する場合があり、プロンプトの内容が優先されるようにする。

6.5　RAG 関連コンポーネント

Retrieval Augmented Generation (RAG)の実現には一般に以下のコンポーネントが関わる。

- 元ファイル
- ファイルデータベース
- ファイル検索コンポーネント

元ファイルは LLM が訓練時に学習していないと想定される情報を保持するファイルである。ファイルデータベースには元ファイルの情報を加工・変換したデータ（以下ファイルデータ）を格納する。ファイル検索コンポーネントは、ユーザーからの入力に関

連が深そうな情報をファイルデータベースから検索して取得し、プロンプト構築コンポーネントを介して LLM に提供する。

RAG とみなせる手法にもさまざまなバリエーションがある。例えば、検索手法としてキーワード検索を使うもの、あるいは知識グラフ上でのグラフ検索を用いるものなどがある。本節では、それらの可能性も念頭に置きつつ、主には、現時点でよく取り上げられる、埋め込み(Embedding)とコサイン類似度に基づく検索を想定する。この場合、RAG で扱う追加情報を扱いやすいサイズのチャンクに分割した上で、各チャンクを埋め込みに変換した上でデータベースに格納する。ユーザーからの入力も埋め込みに変換した上で、これとのコサイン類似度が高いデータベース中の埋め込みを検索して LLM に提供する。

以下、これらについて実施できる管理策を示す。

6.5.1　ファイルデータベース

一般的なデータベース利用システムの場合と同様であり、生成 AI 特有の事項はあまりない。しかし、埋め込みを使う場合にはベクトルの距離に基づく検索を行い、知識グラフを用いる場合にはグラフ構造に基づく検索を行うなど、ファイルデータの持ち方に合わせた検索を行うことになるので、それに適したデータベースを選択する。

LLM 利用 AI システムが満たすべき品質項目ごとに、その品質の達成に向けてファイルデータベースに関して実施するべき事項を以下に掲げる。

- 時間効率性
  - 実行時性能に注意する。基本性能に加えて、期待するデータ取得に至る最終的な実行時性能に注意する。
- 資源効率性
  - 資源効率性を考慮してデザインする。

6.5.2　ファイル検索コンポーネント

ファイルデータベースに入っているファイルデータが種々のデータ品質を満たしていても、ファイル検索コンポーネントがどのような検索を行うか（検索戦略）によって LLM 利用 AI システムの品質は大きく変わる。検索戦略が悪いと検索精度が低くなる。有用なファイルデータを抽出できなかったり、実施中の処理に関して出力してはいけないファイルデータを抽出してしまったりする恐れがある。

ファイル検索コンポーネントが行う検索の目的には LLM 利用 AI システムによって様々な可能性があるため、検索戦略について一般的なことを言うのは難しい。例えば、比較的よく用いられる検索の目的は、LLM 利用 AI システムに入力として与えられた問いに「近い」データを検出することである。この場合、検索戦略として考慮しうる事項としては以下が考えられる。

- 問いとデータの距離をどのように測るか
- どのくらいの距離を近いとみなすか
- 最大何件のデータを検索結果として採用するか

また、ファイルデータベースから検索してプロンプト構築コンポーネントに渡す情報がプライバシー、著作権、公平性などに関する問題を起こさないよう設計しなければならない。

LLM 利用 AI システムが満たすべき品質項目ごとに、その品質の達成に向けてファイル検索コンポーネントに関して実施すべき事項を以下に挙げる。

- 機能要求満足性
  - LLM 利用 AI システムの機能要求水準を満たすのに十分な検索精度を確保する。多くの場合、想定される問題領域のなるべく多くの範囲で一定以上の精度が出ることがよいとされる。
- フェイルセーフ性
  - LLM 利用 AI システムの用途から想定される問題領域の中で、高い安全性が求められる部分に関して、十分な検索精度を確保する。
  - ファイル検索コンポーネント自体もしくは後段に置くプロンプト構築コンポーネントにおいて、検索結果が LLM に渡してよい情報かどうかを判定し、不可の場合、後段に渡さないようにデザインする。
- プライバシー
  - ファイルデータが情報プライバシー権や著作権にしたがって利用制御されている場合、データ利用に関わる権利を侵害しないことを確認しなければならない。データ匿名加工の方法で、データの利用に関わる問題に対策してよい。
- 公平性
  - ファイルデータが要配慮属性を含むデータの場合、バイアスによって権利侵害が生じないことを確認しなければならない。データ匿名加工の方法で、要配慮属性の利用に関わる問題に対策してよい。

6.5.3 ファイルデータの品質

ファイルデータの役割は LLM 利用システムによって異なる。ファイルデータは、LLM 利用 AI システムの目的、要件、仕様に応じて決まるデータ品質を満たす必要がある。

社内規定など、明示的な指示を表すファイルを扱う場合、その一言一句が個別に重要なので、ファイルを分割してできる各ファイルデータの品質が重要であり、データセットとしての分布を考える余地がない。また、営業における顧客情報や、人事における業績査定情報など、組織の活動状況を表すデータの集積の場合も、各データが実態に即していることが重要であって、データセットの分布を操作する余地がない。

一方、一般的な知識を表す事例集として構成されたデータセットを扱う場合も考えられる。そのようなデータセットには、学習用データに求められるのと同様に、代表性と標本選択性のトレードオフが求められる。LLM 利用 AI システムが対処しようとする問題領域を分析した結果得られる問題領域の区分のうち、安全性や公平性の実現のため重要な区分については、安全性や公平性を実現するのに十分なだけの量のデータを確保する必要がある（代表性）。一方、そうすることにより、現実における事象の分布とデータセットの分布が乖離すると、LLM 利用 AI システムの判断の精度が下がる恐れがある（機能要求満足性）。

6.5.3.1 個々のファイルデータの品質

- 正確性、完全性、一貫性
  - 意図されたデータの性格(自然言語文なのかプログラム断片なのか、など)に応じて、これらの品質を保証する。たとえば、既存文書に由来するファイルデータの場合、既存文書の内容に照らし、正確で、欠損がなく、一貫していることを保証する。
- 信憑性、最新性
  - 当該データの来歴に関わるメタ情報を確認する。
  - データ量が膨大になると、信憑性や最新性を満たさないデータが混入する可能性が大きくなることに注意しなければならない。
- 精度
  - 「トークン化（Tokenizers)」や「埋め込み（Embeddings)」を用いる場合には、それに用いる具体的手法の選択に注意しなければならない。例えばトークン化が多言語に対応していないと、対応言語以外の言語について不適切なトークン化が行われる(A1.2.1.4 節)。
- 利用性

- 当該 LLM 利用 AI システムにおけるデータの利用目的に照らしてデータが利用性を満たすか否かを決定しなければならない。

- 標準適合性
  - 当該 LLM 利用 AI システムが準拠する規則にしたがわなければならない。プライバシーや著作権を保護する法律への適合性もここに含まれる。

6.5.3.2　ファイルデータのデータセットとしての品質

- 代表性、標本選択性
  - ファイルデータが一般的知識を表す事例集として使われるデータセットを含む場合、代表性と標本選択性のトレードオフにどう対処するかを、LLM 利用 AI システムの用途を考慮して決める。
- 重複性
  - 規定文書など、全体としてひと塊の情報を表す大量のデータを分割してファイルデータベースに保持する場合、ある区分と隣接する区分でデータの一部を共有し、問い合わせ応答に対する変化が滑らかになるようにする。
- 来歴性、最新性
  - 外部のデータセットをファイルデータとして利用する際に、事前に、当該データセットの来歴性および最新性を調べて、利用目的と整合していることを確認する。
- 文脈整合性
  - 外部の複数のデータ源から外部データを整備する場合、組み合わせる外部データの文脈整合性を確認する。
  - 訓練時に学習済みで LLM が保持する内部知識と、RAG で導入する外部データの文脈整合性を確認する。ファイルデータと内部知識が共通する話題を参照している可能性がある場合を考慮する。共通性がある場合、両者の意味内容がアラインしているか、互いに矛盾するか、などの影響が生じうることを考慮する。
- 時制整合性
  - LLM 本体のモデル訓練時の学習データセットが陳腐化している可能性があるとき、データ最新性の問題を解決する外部データを RAG で導入する。一般には、ファイルデータと内部知識が矛盾する場合があり、ファイルデータの内容が優先されるようにする。

6.6　外部連携コンポーネント

外部連携コンポーネントは、LLM からの出力を受け取って、そこに外部サービスの呼び出し要求が含まれている場合、そのサービスの呼び出しを行う。そして、呼び出し

の結果として得られた応答を、後段のコンポーネントに渡す。多くの場合、後段のコンポーネントを介して応答が同じ、または別の LLM に渡るようにする。外部サービスの呼び出しを行うため、外部連携コンポーネントはいくつかのプラグインツールを備える。

外部連携コンポーネントによって呼び出す外部サービスの役割には様々なものが考えられるが、以下のように大別することができる。

- 外部から情報を取得する。
- 外部に情報を提供する。
- 外部の状態を変更する。

これらの役割を踏まえて、LLM 利用 AI システムの品質に対する影響を考慮し、管理策を実施する必要がある。外部から情報を取得する場合、その情報がシステムに問題を引き起こす可能性を考慮する。外部に情報を提供する場合、その情報が守秘義務を伴う可能性や、提供先で問題を引き起こす可能性を考慮する。外部の状態を変更する場合、変更対象に与える影響だけでなく、その変更が LLM 利用 AI システムのユーザーや運用者にもたらす責任について考慮する。

外部連携コンポーネントによる入出力が問題を生じる可能性への対策は、設計時の他のコンポーネントでの取り組みに加えて、外部連携コンポーネントにおいて実行時に行う必要がある。外部連携コンポーネントを介した入出力は、LLM 利用 AI システムの入出力の一環であるが、他の入出力と違い、どの連携先とどんな情報をやりとりするか、などの入出力条件を実行時に LLM に決めさせるのが特徴である。LLM の振る舞いに不確定性があるため、LLM が決める入出力条件が問題を生じないことを設計時には保証できない。そのため、実行時に外部連携コンポーネントで対策する必要がある。

LLM 利用 AI システムが満たすべき品質項目ごとに、その品質の達成に向けて外部連携コンポーネントに関して実施するべき事項を以下に挙げる。

- 進行性、可用性
  - 外部サービス呼び出しにタイムアウトを設ける。
- 説明性
  - 外部データを、後段で LLM によるコンテンツ生成に使う場合、生成コンテンツの正当化理由として使えるよう、外部データにその出自を添えて後段のコンポーネントに渡す。
- アクセス制御性
  - プラグインツールの実行（利用）に際して、当該プラグインツールに期待

される利用許可条件にしたがったアクセス制御をデザインする。
  - 事前に当該プラグインツールに付された情報（SBOM 等）を確認し、外部連携としての利用可否を判断しておかなければならない。
- プライバシー、公平性
  - 外部データが情報プライバシー権や著作権にしたがって利用制御されている場合、データ利用に関わる権利を侵害しないことを確認しなければならない。データ匿名加工の方法で、データの利用に関わる問題に対策してよい。
  - 外部データが要配慮属性を含むデータの場合、バイアスによって権利侵害が生じないことを確認しなければならない。データ匿名加工の方法で、要配慮属性の利用に関わる問題に対策してよい。
- 安全性
  - 外部連携コンポーネント自体または後段に置くプロンプト構築コンポーネントにおいて、外部データを、LLM に入力するかどうかを判定し、不可の場合、後段への入力を抑止するようにデザインする。入力フィルター(6.5 節)の適用を検討する。
  - 外部に提供するデータが提供してよいものかどうかを判定し、不可の場合、外部への提供を抑止するようにデザインする。出力フィルター（6.8 節）の適用を考慮する。提供してよいかどうかは、データの内容だけでなく、提供先や提供時期など様々な条件に依存することに留意する。
- サイバーセキュリティ―
  - 第 7 章を参照。

6.7 入力フィルター

入力フィルターは LLM に入力するべきでない情報を検知し排除する。入力フィルターは、プロンプトに組み込むために LLM 利用システムの外から運用時に取得する情報に適用する。例えば、ユーザー入力と、外部連携コンポーネントが返す値が該当する。

しかし、入力フィルターに検知させたい種類のユーザー入力を高精度に検知するのは一般に難しい。その原因の一つは、検知させたい種類のユーザー入力を明確に定義するのが難しいことにある。その問題を克服する手段として、LLM にその検知を行わせる手法が用いられる。うまくいけばかなり有効であるが、入力フィルター自体の品質マネジメントが難しくなる難点がある。

また、どんな情報を入力フィルターで検知し排除するべきかは、LLM 利用 AI システムの用途や要件に依存する。

LLM 利用 AI システムが満たすべき品質ごとに、入力フィルターで実施するべき事項を以下に挙げる。

- プライバシー、公平性、安全性
  - LLM 利用 AI システムの用途や要件に照らして必然性がないプライバシーに関わる情報を検知し、排除する。
  - LLM 利用 AI システムの用途や要件に照らして必然性のない、プライバシーに関わる情報の出力、バイアスを反映した出力、危険な出力に対する要求を検知し、排除する。
- サイバーセキュリティ
  - 第 7 章を参照。

6.8 出力フィルター

出力フィルターは、LLM が生成した出力から、LLM 利用 AI システムの外に出すべきでない情報を検知し排除する。LLM の動作に不確定性があるため、他のコンポーネントでの取り組みだけでは LLM の出力に LLM 利用 AI システムの外に出すべきでない情報が含まれないことを保証するのが難しい。そこで、最後の砦として出力フィルターを用いることを考慮する。

LLM 利用 AI システムが満たすべき品質項目ごとに、その品質の達成に向けて出力フィルターに関して実施するべき事項を以下に挙げる。

- 説明性
  - 生成コンテンツに正当化理由を補う。
- 否認防止性
  - LLM の直接あるいは間接的な出力コンテンツに対して、「AI 技術利用」を示すメタ情報を付加し、後段コンポーネントに出力する。対象コンテンツが画像の場合、透かしを画像に挿入する方法がしばしば採用される。
  - 当該の LLM 利用 AI システムに関わる規制法にしたがって、否認防止性を達成しなければならない。
- 安全性、プライバシー、著作権、公平性
  - LLM 内部知識として保持されている情報が、LLM 利用 AI システムの用途に照らして安全性を損なう情報、パーソナルデータ、著作権にしたがうデータ、あるいはバイアスと関わる場合、当該データの漏洩を防止する一般的な対策を講じることは難しい。出力コンテンツに対するフィルター機能を導入し、それぞれ、安全性を損なう情報、想定外のパーソナルデータ

漏洩、著作権者の権利侵害に相当する出力、あるいは想定外のバイアスの出力を防ぐようにデザインする。

- 移植性
  - LLM は、同一の開発組織によるシリーズ版であっても学習パラメータ規模によって能力が異なり、同一プロンプト入力に対して、同じ機能振舞いを保証するわけではない（機能振舞いの揺らぎ）。LLM への入力に関わるプロンプトならびに出力フィルターに関わるデザインと組み（トリプレットパターン）にして移植性を評価しなければならない。

6.9 HMI コンポーネント

LLM 利用 AI システムが満たすべき品質項目ごとに、その品質の達成に向けて HMI コンポーネントに関して実施するべき事項を以下に挙げる。

- インタラクション性
  - 自然言語インタフェースが主となる LLM 利用 AI システムでは、エンドユーザーが習得している、あるいは使っている言語の違いを考慮してデザインする。
  - カラー画像インタフェースが主となる LLM 利用 AI システムでは、エンドユーザーの色彩への知覚の度合いを考慮してデザインする。
- 可制御性
  - システム運用者であるユーザーが、LLM 利用 AI システムの動作振舞いを中断あるいは停止可能なようにデザインしなければならない。
  - エンドユーザーが、LLM 利用 AI システムの動作振舞いを中断あるいは停止可能なようにデザインする。
- 説明性
  - エンドユーザーの要求がある場合には、可能な限り出力の正当化理由を提示できるようにデザインする。
- 習得性
  - LLM 利用 AI システムでは、UX からの HMI デザインが関わる。HMI を実現するコンポーネントでは、GUI 対話部品の提供機能と目的を明示したデザインを行う。
  - 一般に、自然言語インタフェースは、他の手段を用いるシステムに比べて、習得性が向上する。一方、習得している、あるいは使っている言語が、エンドユーザーによって異なる場合、アクセシビリティ（4.3.2 節）の観点から多様な言語に対応したデザインを行う。
- アクセス制御性、真正性

- LLM 利用 AI システムの目的と用途によっては、ユーザーを認証して、認証を通らないユーザーの利用を拒否しなければならない。これによって、ユーザー入力プロンプトの来歴が明らかになる。

- 介入性
  - ユーザーが介入指示できるような対話インタフェースをデザインする。
- 入力制限性
  - ユーザー入力情報を制限あるいは限定することで、入力フィルターの役割を果たすようにデザインする。

7 LLM 利用 AI システムのセキュリティ対策

第 2 章と 4.4 節に記載されている内容を元に実際の設計開発におけるセキュリティ対策に向け留意しておくべき事項を述べる。セキュリティリスク分析や管理策立案のために考慮・検討するべき項目は製品やサービスによって千差万別であるため、本書では「考慮・検討すべき項目」を考える上でのヒント・道標となる情報を紹介するに留める。

なお本章では、情報セキュリティにおいて継承されてきた「攻撃によって被害が発生すること」を前提として記述する。LLM 利用 AI システムでは、生成 AI の特性によって攻撃がなく脆弱性のみによって被害が発生する場合がある。そのような問題は、安全性(4.5 節)の一部として捉える。

7.1 LLM 利用 AI システムにおけるセキュリティリスクの分析

LLM 利用 AI システムにおけるセキュリティリスクを紹介する。以下の観点や攻撃事例を元に脅威と脆弱性を洗い出し管理策を施行すればよい。なお、対象とするシステムに固有に想定される攻撃事例を追加・選択する必要がある。

7.1.1 生成 AI 固有のセキュリティリスクの存在

生成 AI の特性に強く依存するリスクについては個別に脅威と脆弱性を分析し、脅威と脆弱性それぞれに対して管理策を講じる必要がある。本書で定義しているコンポーネント品質・データ品質の各項目が攻撃対象となり得る。

7.1.2 注意するべき攻撃界面

LLM との入出力 I/F が攻撃界面となる。LLM と、入力データ・プロンプト（システムプロンプト・ユーザプロンプト両方）・RAG のために導入するソフトウェア・プラグインの間の I/F が攻撃界面である。入力データ・プロンプトの監査は「サニタイジング」「フィルタリング」と呼ばれる。入力の対象が HMI・プラグイン・そのほかのライブラリやプログラムのいずれについても十分なサニタイジングが必要である。

7.1.3 被害の出方の違い

従来の情報セキュリティの基礎概念である「情報資産に脆弱性があるときに、脅威が脆弱性を襲うことを攻撃と呼ぶ」のみではなく、脅威がなくても脆弱性のみで被害が発生することがあり得る。本章では従来からの「脆弱性があるところに脅威が襲いかかり攻撃する」ことを前提とし、「脆弱性のみで被害が発生する」場合については安全性として第 6 章で取り扱い、(7.3 節を除き)本章では取り扱わない。

7.1.4 レッドチーミングに対する要請

本ガイドラインは、第三者が開発・提供する LLM（基盤モデル）を再利用部品として用いて、別途与えられた要求仕様を満たすべく開発する LLM 利用 AI システムに関わる品質マネジメントを取り扱っている。LLM 自体の設計開発・設定やチューニングに関わる分析は、FOR-LLM に関わることから、当該 LLM の品質マネジメントに直接関与することができない。これは、セキュリティリスクアセスメントの結果を用いて機械学習要素やモデルに関わる品質マネジメントを行なってきた従来 AI の場合と全く状況が異なる。つまり、LLM を用いる WITH 開発において、セキュリティリスクアセスメントを実施し、その結果を元にレッドチーミングテストを行うことになる。

この時、LLM に対するレッドチーミング（モデル・レッドチーミング）と LLM 利用 AI システムに対するレッドチーミング（システム・レッドチーミング）を行うことに注意して欲しい。前者のモデル・レッドチーミングは、LLM 提供者から得られる情報（おそらく提供者によるモデル・レッドチーミングの結果に基づく）に加えて、現開発対象の LLM 利用 AI システムの要求仕様に即して実施したセキュリティリスクアセスメントの結果から、追加して行うべきテスト（LLM を対象とするペネトレーションテスト）を実施する [30]。後者のシステム・レッドチーミングは、LLM 利用 AI システムの構築後に、当該システム全体に対して実施する [31]。

---

[30] 選択した LLM は、FOR-LLM の段階で、一般的なレッドチーミングテストが実施されていても、WITH-LLM の文脈で期待するレッドチーミングテストに合格するとは限らない。たとえば、欧米文化圏で開発された LLM をアジアなどの異なる文化圏で利用・運用する場合には、倫理・道徳上の懸念に関わる追加テストを必要とする。

[31] たとえば、LLM 利用 AI システムの作動・運用環境（実行インフラ）から生じる脆弱性を考慮した追加テストを必要とする。

なお、セキュリティリスクアセスメントやモデル・レッドチーミングテストは早い段階で実施することが望まれる。

7.1.5 生成 AI の性質に起因するセキュリティリスクの例

LLM 利用 AI システムにおいて注意するべきセキュリティリスク（攻撃）は色々あるが、その中で、AI 特有の性質に起因するものとして、プロンプトインジェクションがある。これは、プロンプトの一部として取り込まれる入力を工夫することにより、LLM に想定外の出力を行わせる攻撃である。とりわけ、倫理的、法的、その他何らかの意味で“不適切”な出力を防止するための施策を LLM に回避させようとする試みはジェイルブレイク (jailbreak)と呼ばれる。プロンプトに攻撃をどう埋め込むかにより、以下を区別できる。

- ダイレクトプロンプトインジェクション：攻撃者が攻撃内容をプロンプトに直接入力する攻撃。
- インダイレクトプロンプトインジェクション：攻撃者が攻撃内容を画像やファイルなどで間接的に攻撃内容を与える攻撃。
- ペイロード分割：プロンプトを複数に分けて入力することで検出を逃れようとする攻撃。分割されたプロンプトは一見攻撃用のものと判別できないよう加工されている場合がある。
- マルチモーダルインジェクション：攻撃プロンプトを書いた黒板を写真に撮り画像を入力すると攻撃プロンプトとして機能することを悪用した攻撃。音声でも同様の攻撃が可能である。

7.1.6 セキュリティリスクアセスメントに際して

脅威と脆弱性の導出に当たり、以下の点に留意するとよい。セキュリティリスクアセスメントの作業の流れについては機械学習品質マネジメントガイドライン第 4 版[AIQMv4]の AI セキュリティ章と同様である。脆弱性のみで被害が成立する場合は安全性のリスクアセスメントで取り上げること。

(1) アクセス制御性・介入性・真正性・責任追及性・否認防止性

これらの性質は主に運用時に関係する。7.2 節で、これらの性質を機密性・完全性・可用性[32]にブレークダウンすることができることを示す。ブレークダウンした内容を脅威・脆弱性の割り出しや管理策導出に用いるとよい。

(2) AI 固有でないセキュリティリスク・従来 AI セキュリティのリスク

AI 固有でないセキュリティリスクについては ISO/IEC 27005 の附表、従来 AI セキュリティのリスクに対しては機械学習品質マネジメントガイドライン第 4 版の AI セキュリティ章の定義を参考に脅威と脆弱性を割り出し、管理策を適用するとよい。本書では詳細に述べない。

(3) 生成 AI 固有のセキュリティリスク

7.1.5 節などを参考に攻撃事例を挙げ、脅威と脆弱性を列挙し管理策を導出する。直接 LLM に対策を施すことができないため、LLM の外部に入力を監視するシステムを設けて、攻撃の元となるデータやプロンプトの入力に対してサニタイジングを行う必要がある。サニタイジング技術は未成熟・これからの発展が期待されるものであり、技術動向に注目する必要がある。

リスクの事例については以下の文献を参考にするとよい。[OWASP 2024] [DHS 2023] [NIST 2023] [NIST 2023b] [Google 2024] [OpenAI 2024b] [Anthropic 2024] [Meta 2023]

7.1.7 生成 AI の出力に関する注意事項

生成 AI においては、ステークホルダーが想定し得ない出力が得られる場合があり、未知であった機密情報が検出・ばく露されるなど人の想定を超える被害が発生し得る。このため生成 AI を含めたシステム全体の脆弱性分析の対象として生成 AI からの出力に注意する必要があり、入力と同様に出力に対してもサニタイジングが求められる可能性がある。

7.2 セキュリティの品質特性の扱い方

4.4 節は生成 AI 利用システムが満たすべき品質項目としてのセキュリティを示す。項目としては、アクセス制御性・介入性・真正性・責任追及性・否認防止性を挙げている。

---

[32] 本節および 7.2 節で挙げている機密性・完全性・可用性は、情報セキュリティの文脈における意味で使っている。特に可用性は 4.2.5 節で挙げたシステムの機能振舞いに関する信頼性の副品質特性とは異なり、攻撃から守るべきアセットの性質について述べている。

これらは生成 AI 利用システムにおける生成 AI の特性・特徴を考慮した管理策として有効な項目となっており、これらの特性が侵害される場合を機密性・完全性・可用性の 3 要素から考え、セキュリティリスクアセスメントに組み込むとよい。

セキュリティリスクアセスメント手法そのものについては、機械学習品質マネジメントガイドライン第 4 版のものと同じでよく、脅威と脆弱性を洗い出す際に本書が定義しているセキュリティの品質特性から機密性・完全性・可用性に対する侵害を導出すればよい。

(1) アクセス制御性の侵害

利用許可条件にしたがって情報を保護し処理できること。

(2) 介入性の侵害

システム動作に対して外部から介入してシステムの状態を安定させるので、何らかの被害が発生している場合の管理策として外部から介入することが適用できることを検討する必要がある。

- 機密性の側面：なし
- 完全性の側面：外部からの介入できない動作の発生
- 可用性の側面：外部からの介入できない状況の発生

(3) 真正性の侵害

制御対象とアクセス主体の身元が明らかであること

- 機密性の側面：身元情報に機密情報が紛れ込む。
- 完全性の側面：取得した身元に関連する情報に抜け・漏れ・改ざんなどが発生する。
- 可用性の側面：身元に関連する情報を取得できない状況が発生する。

(4) 責任追及性の侵害

LLM 利用 AI システムの出力コンテンツを生成した理由や要因を再現し提示できること

- 機密性の側面：コンテンツを生成した理由や要因に機密情報が紛れ込む。
- 完全性の側面：コンテンツを生成した理由や要因に抜け・漏れ・改ざんなどが発生する。
- 可用性の側面：コンテンツを出力した理由や要因が取得できない状況が発生する。

7.3 想定外の情報が出力に混入する事例

ここに挙げるのは、攻撃なしに脆弱性のみで被害が発生するため、セキュリティではなく安全性(4.5 節)に関わるリスクであり、本章のスコープ外であるが、忘備のため挙げておく。

- 利用者に悪意がなくても蓄積された知識と入力されたデータから利用者が想定できない内容（虚偽情報・新たな知見）が出力に入り込む場合があり、利用者に想定しない被害をもたらす場合がある。
- プログラム・SQL クエリ・ファイルパス・送信する電子メールの文面を生成させた場合に、想定しない脆弱性が混入する場合がある。
- 学習モデルからステークホルダーが意図しない外部システムにアクセスする場合がある。
- 意図せず脆弱性が混入したプログラムが出力される場合がある。

8 おわりに

8.1 今後の改訂の方針

品質マネジメントの詳細は、対象 AI の技術的な特徴によって異なる。産総研の機械学習品質マネジメントガイドラインは、識別 AI と予測 AI を主な対象とする。それに対して本ガイドラインは、LLM を中心とする生成 AI を対象とし、LLM 利用 AI システムの品質マネジメントを取り扱う。今後、当該技術の発展に応じて、それぞれ改訂を進める。なお、応用セクター向けファインチューニングに関する品質マネジメントは、組織的な方策の比重が高いことから別に議論する。その他、強化学習に関わる品質マネジメントガイドラインの整備についても検討する。

8.2 LLM エージェントについて

2025 年初めまでに、最大手の LLM 開発者が、相次いで PC 内の操作や、Web ブラウザ経由でのネットワークサービスの利用を行うことができる LLM 利用 AI システムをサービスとして提供開始した。これらに留まらず、2025 年中には多くの企業でいわゆる「AI エージェント」の開発が盛んにおこなわれるだろうという見立てが広まっている。「AI エージェント」という表記は以前よりある AI ベースのマルチエージェントシステムの研究と紛らわしいため、ここでは LLM エージェント(2.1.1 節)と呼ぶ。

外部連携コンポーネント(6.6 節)を利用する LLM 利用 AI システムは、最も単純なケースを除いては LLM エージェントと見なせると思われる。しかし、LLM エージェントに特徴的な機能やその品質を主に担うのは、「オーケストレーション」であると考えられる。今後、オーケストレーションの高度化を踏まえて、LLM エージェントの品質マネジメントの考え方を整理することが、本書の今後の改訂に向けた重要な課題の一つである。

付録1 大規模言語モデルの特徴

本付録章では再利用開発モデル（3.2 節）の「機械学習技術」に関連する事項を参考情報として整理する。LLM の原理的な性質（A1.1.1 節）と経験的な知見による LLM の特徴（A1.1.2 節）を説明し、プロンプトによる LLM 制御の方法（A1.1.3 節）を概観する。

A1.1 大規模言語モデルの仕組み

A1.1.1 確率モデル

LLM の典型的な使い方は、質問を入力し回答を得ることである。LLM の中核をなす言語モデルは確率的な振舞いを示す。自然言語処理の問題に確率モデル[岡崎 2022]からアプローチする。

一般には、文の構造的な構成を文法規則で表す。文を単語の列（並び）とすると、文法規則は単語を並べる規則である。ここで、単語よりも一般的にトークン（Token）とし、文をトークンの列とする。

自然言語では、簡明な文法規則を定式化することが難しい。膨大な文例（コーパス）を収集し、実際の文を構成するトークンの共起関係を表す確率モデル（言語モデル）を求める。複数のトークンが文の構成要素として同時に出現するとき、そのトークンには共起関係があるという。共起確率が高いトークンから構成される文は、背景にある未知の文法規則にしたがうと考えられる。LLM の生成は、大まかには、質問と回答を結合したトークン列が大きな確率になる内容を出力することである。

A1.1.2 訓練学習

LLM は極めて大きなニューラルネットで表現した言語モデルであり、複雑な学習モデルを用いることで、膨大な文から得られる多様な言語表現を総括的に表す。学習パイプラインの模式的な説明図を示す（図 9）。

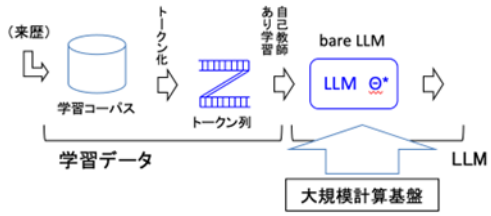

図 9 学習パイプライン

次のようなステップで訓練学習し訓練済み学習モデル（bare LLM）を得る。

1. 訓練学習の土台となる学習コーパスを収集する。インターネットのクローリング（Crawling）によってデジタルテキストを集め、事前処理加工を行って学習に用いるコーパスを得る。
2. 学習コーパスからトークン列に変換して学習データを構築する。
3. 自己教師あり学習の方法で学習モデルを訓練する。GPU クラウド環境などの大規模計算基盤上で実行する。

LLM の訓練学習は、自己教師あり学習（Self-Supervised Learning）で行われ、代表的な方法として、自己回帰学習の次トークン予測[GPT 2018]がある。まず、訓練データとして長大なトークン列を与える。先頭から処理していく方法で、走査済みトークン列の次に出現するトークンが、訓練データ中の実トークンに一致する確率が大きくなるように、学習モデルを訓練する[33]。この処理を訓練対象トークン列全体に行うことで、言語モデルを学習結果として得る。なお、自己教師あり学習の問題設定は、次トークン予測の他にも可能であり、マスク言語を訓練データとする穴埋め問題による学習方法[Devlin 2019]が知られている。

A1.1.3 出力生成

出力生成は、原理的には、言語モデルからサンプリングの方法でデータ（トークン列）を得ることである。ランダムサンプリングの場合、何らかの文が生成されるだけで、言語モデルの有用な使い方とはいえない。出力の促進指示（プロンプト）として、テキスト断片を表すトークン列を与え、言語モデルから後続トークンをサンプリングする。出現確率が大きい後続トークンを選べば、確率的に尤もらしいトークン列になる。求めた生成トークンを列に追加し、これを新たなテキスト断片として、さらに続く文を完成する過程を繰り返す。

[33] 誤差関数（ Loss Function ）を最小化する。

原理的には、後続トークン選択に際して、出現確率をもとにトークンを決めれば良い。しかし、予測確率の高いトークンを順番にサンプリングして繋げるだけでは、出力文（トークン列全体）の内容が尤もらしいかという点で不十分なことが多い。そこで、計算効率を考慮して、デコード手法のヒューリスティクスを考える。具体的な方法として、貪欲探索（Greedy Search）、ビーム探索（Beam Search）による制御などがある。また、出力層の確率計算に温度（Temperature）付き softmax 関数を用い、温度パラメータ値 T によって出力制御する方法が使われる。

温度パラメータ T を 0 にすると出力層の計算が確定的になるが、言語モデルの推論実行に関わるさまざまな要素から唯一に定まることがない。GPU オペレーション、メモリ参照パターン、数値計算精度などによる非決定性の影響を受ける[Riach 2019]。また、サンプリングのランダム性も非決定性を生じる原因となる[Gawlikowski 2022]。

LLM の振舞いについて考える場合、プロンプトを質問や問い合わせと解釈すると、生成文が回答で、チャットボットや Q&A システムの雛形になる。また、プロンプトに英語の文を与えて対応する日本語の文を得れば、英日自動翻訳の機能を示す。さまざまな自然言語タスクを実現することができる[McCann 2018]。

A1.1.4 基盤モデルとしての LLM

LLM は、自然言語処理のさまざまな応用を実現する後段タスク（Downstream Tasks）で使われる、すなわち再利用される基盤モデル（Foundation Model）[CRFM 2021]の役割を果たす。

基盤モデルの LLM を FOR 開発の成果物とする時、特定の応用機能を提供する後段タスクの開発は、事前訓練済みモデル（Pre-trained Model）である LLM を流用し、当該タスク向けの学習データを用いた転移学習[Tan 2018]による WITH 開発になる。

また、LLM 自身を単独で用いる文脈内学習（In Context Learning）の方法[Brown 2020]の場合、LLM は厳密な意味での基盤モデルではない。一方で、文脈内学習を微調整（Fine-tuning）とみなすこともできる[Dai 2023]。さらに、LLM の技術的な特徴が禍いし、LLM 単独で、一般に期待する応用機能を実現することができない場合がある。後段での開発作業（LLM 利用ソフトウェア開発）によってユーザーが利用するアプリケーション開発を実施する必要性があるという意味で、広義の基盤モデルといえる。

A1.1.5 学習アーキテクチャ

LLM はニューラルネットワークの学習モデルとしてトランスフォーマー（Transformer）[Attention 2017]の亜種のデコーダー・トランスフォーマー（Decoder-only Transformer）を用いる[GPT 2018]。模式的な学習アーキテクチャを図 10 に示した。

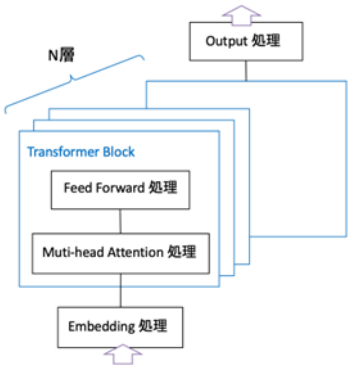

図 10 デコーダー・トランスフォーマー

この学習アーキテクチャの構成要素は、埋込み処理、トランスフォーマーブロック、出力処理に大別でき、また、トランスフォーマーブロックは、マルチヘッドアテンション処理とフィードフォワード処理からなる。N 層に積み重ねたトランスフォーマーブロックに続く最後の出力層が生成列を構成するトークンの出現確率を返す。

既存 LLM をモデル規模で比較することが多い。LLM のモデル規模はトランスフォーマーを構成する学習パラメータ数で表現される。図 10 を参照して、規模の見積もり法を示す。LLM 規模の概算値に影響する主なパラメータを

- トークン（語彙）数：V
- 内部状態数：D
- トランスフォーマーブロックの層数：N

とする時、標準的なデコーダー・トランスフォーマーの学習アーキテクチャでは、

- 埋込み処理：$V \times D$
- マルチヘッドアテンション処理：$4 \times D^2$
- フィードフォワード処理：$8 \times D^2$
- 出力処理：$V \times D$

となる。LLM モデル規模という場合、出力処理を含めないことが多い。そこで、

$$概算値 = V \times D + 12 \times D^2 \times N$$

で、たとえば、V=4×10^4、D=2^{12}、N=32 だと、6.6×10^9（6.6B）になる [34]。

A1.2 大規模言語モデルの特徴

A1.2.1 経験的な知見

LLM の機能振舞いについて、原理的な面からの理解が進んでいるわけではないが、多くの実験を通して経験的な知見が整理されている。

A1.2.1.1 定量的な分析

定量的な分析に関連して、スケーリング則が成り立つとされている[Kaplan 2020]。多層のトランスフォーマーブロックから求めたモデル規模を M（$=12\times D^2\times N$）、訓練データに用いたトークン列の長さを T、訓練学習に費やす計算実行時間を C とする [35]。誤差関数 L が、各々のパラメータの冪乗にしたがって改善される、つまり、$L\propto 1/X^{\alpha}$（$X\in\{M, T, C\}$）の形になるという経験則である。

LLM の規模 M がある閾値を超えると、小さい言語モデルに見られなかった機能振舞いを示すという観察もある。ベンチマーク問題でプロンプトに対する評価指標 P を測定したところ、$P\propto M^{\alpha}$ に比べて P の値が向上する。多層のトランスフォーマーブロックを構成する学習パラメータが示す自由度が組み合わさり、全体が揃って振る舞うと考え、創発的な振舞い（Emergent Behavior）と呼ぶ[Wei 2022]。一方で、この創発性は、規模 M に対して非線形性を示す評価指標 P を用いたことが原因で現れた「蜃気楼」であるという研究報告もある[Mirage 2023]。現時点で、産業界では、スケーリング則と創発性が成り立つと仮定し、より規模の大きい LLM 開発を行うというビジネス上の意思決定の拠り所にしている。

A1.2.1.2 大規模化の限界

規模を大きくすると、当然のごとく、計算の手間が膨れ上がる。たとえば、現在の計算機性能を基準として考えるとき、現実的な経済コストで、どのくらいの規模のモデル訓練まで可能かという疑問が生じる。また、一般にソフトウェア開発が知識集約型であ

---

[34] GPT-3（6.7B）の概算値[Brown 2020]

[35] $C\sim 6\times M\times B\times S$、B はバッチサイズ、S はエポック数で、C の単位は FP-day（8.64×10^{19} floating-point operations）。

ることを考えると、研究開発スタッフの人的なコストも経済的な観点からの議論では重要になる。

文献[Cottier 2024]は、2015 年 10 月から 2023 年 12 月の期間に開発された大規模言語モデルの公開情報から、モデル訓練に要する計算機コストと開発の人的コストを抽出し、フロンティア基盤モデルの開発コストを見積もった。計算機コストはクラウドレンタルの場合（OpenAI 社・Anthropic 社など）と独自 TPU 基盤の場合（Google 社など）で共に年に 2.4 倍の増加率を示す。前者の場合、2027 年には 10 億ドル規模のコストに達することがわかった。また、開発の人的コストは、試行錯誤を含む実験・微調整・評価の開発全般にわたる工数から推定した。その結果、全体に対して人的コストが 30 から 40%の割合を占める [36]。技術市場の寡占化が進む要因の一つになる。

もう一つの観察として、入手可能で有用なデータを使い尽くしてしまうという予測がある。新たな学習データを得ることが難しくなり大規模化が限界を迎える[Villalobos 2024]。素朴には、生成 AI で合成したデータをモデル訓練に用いることで、この問題を解決できるとする意見がある。しかし、合成データは、訓練データの特徴（確率分布）を近似的に学習した生成モデルからのサンプリングによって得られることから、尤度が大きいデータが合成される確率が高い。この生成・学習の過程を繰り返すと、学習内容の多様性が失われ、モデル崩壊が生じることが指摘されている[Shumailov 2024]。一般に知られている「エコーチェンバー」と似た状況であり、その訓練結果として得られるモデルは、多様性を失い、理論的に常に同じデータを生成するようになる。

技術的には、大規模化と逆の傾向として、モデルマージ（Model merge）手法の研究が進んでいる。これは、異なる機能で良い性能を示す中規模モデルを組み合わせて、複合的な機能を提供するモデルを合成する方法、「3 人寄れば文殊の知恵」である。開発コスト見積もり上、中規模モデルの開発であれば経済的に実行可能であり、また、モデルマージ処理は訓練を伴わないことから、計算機コストを削減できる。一方で、技術的には、どのようにマージ（重ね合わせ）するかの系統的な方策はなく、試行錯誤によるマージと評価の繰り返しにならざるを得なかった。進化的モデルマージ[Akiba 2024]は、進化計算を利用して、マージと評価を自動化する枠組みを導入した。特に、オープンソ

[36] AWS 社からの Anthropic 社への出資（Claude 開発）は 2024 年 4 月までに 40 億ドルにのぼる、と報道されている。

ースとして開発された多彩なモデルを、目的に応じて、適宜選択しモデルマージするという新しい開発パラダイムにつながると期待されている。

大規模モデルが目的に適したサブモデルの集まりとして構成されるという発想は MoE（Mixture of Excellence）にも見られる。MoE は学習モデル全体が専門性の高いサブネットに分割されるという仮説に基づいて、訓練に関わる計算の効率向上からのアプローチである。訓練時に更新対象の大規模モデルを局所化するというアイデアに基づく。膨大な学習トークンの中から当該サブネットの訓練に有用なトークンを選択し、訓練対象（学習パラメータ数）の規模を適正に保つことである。トークンを基に対象サブネット候補を選ぶ戦略[Shazeer 2017]とサブネットが対象トークンを選ぶ戦略[Zhou 2022]がある。後者によってトランスフォーマーアーキテクチャのファインチューニング時に良い性能を示したことが報告されている。また、中国のベンチャー会社 DeepSeek は、MoE の考え方をトランスフォーマーアーキテクチャに対して全面的に採用し、さらに、計算効率を向上させるモデルアーキテクチャ上の様々な工夫を加えて、LLM 訓練に関わる計算資源（GPU）を大幅に低減した [DeepSeek-AI 2024]。

LLM の「推論」実行時 [37]に、生成するコンテンツのデータ分布を動的に適応させる方法が提案されている[Snell 2024]。入力プロンプトの内容に応じた適応の方策を決定するテスト時計算を追加する。規模の異なる訓練済みモデルを準備し、数学問題に対して、テスト時計算を補った小規模モデルと 14 倍の規模のモデルを比較したところ、同等の正解率が得られることを実験で確認した。推論能力の向上に大きく寄与する。

A1.2.1.3 定性的な分析

次に、表 2 は定性的な観点の特徴 を整理したものである[Chang 2023]。さまざまなプロンプトを入力し LLM の出力を調べたもので、外部観測による振舞い評価といえる。

表 2. LLM の定性的な特徴

1. 一般に文法的に正しいテキストを生成する。
2. 主語と動詞の一致を学習するが、節や特定の単語が間にあると影響を受けやすい。
3. 訓練の早い段階で構文規則を学習する。
4. 明示的な位置情報がなくても語順を学習できるが、多くの例で語順は必要ない。
5. 論証構造、同義語、多義語など、個々の単語の意味的複合的特性を学習する。

[37] 機械学習分野の習慣で「テスト時（Test-time）」と呼ぶ。訓練データに対するテストデータと同じニュアンスの「テスト」である。

6. 否定をうまく扱えず、モデルの規模が大きくなるにつれて悪化することが多い。
7. 一貫性はあるものの脆弱な状況モデルを作るように思える。
8. 基本的なアナロジー、メタファー、比喩表現を扱うように振舞う。
9. テキストの登場人物の心理状態を推測できるが、含意された意味や用法の扱いには苦戦する。
10. モデルの規模が大きくなるにつれて、事実や対象の常識的な性質を学習するが、物理的な性質は人間ほど敏感ではない。
11. 事実の学習は、コーパス中の文脈だけでなく、事実の出現頻度に影響を受けやすい。
12. 限定的であるが、アクションやイベントに関する常識推論の自明でない能力があるように振舞う。
13. プロンプトを工夫することで基本的な論理推論を行えるが、複雑な推論には依然として苦戦する。
14. 特定の入力に限って、数値推論や確率推論の基本的な能力を示す。
15. 規模が大きくなるにつれて、学習コーパスから丸暗記したテキストを生成する可能性が高まる。
16. 入力プロンプトが指定する文脈と整合性があり、学習コーパスにないテキストを生成する。
17. 目的を定めたプロンプトによって、攻撃的なテキストやヘイトスピーチを生成することがある。
18. 私的な情報を晒す可能性があるが、多くの場合、特定の個人に結びつくことはない。
19. 人口統計グループ間で、実性能および有害テキストの発生確率が異なる。
20. 性別、性的指向、人種、宗教、などの人口統計学上の属性に関わる有害なステレオタイプを反映する。
21. 事実と異なる内容や、安全でない助言を強い説得力で生成できる。
22. 生成テキストを人間が作成したテキストと区別することは難しい。
23. LLM の“個性”や政治性は、入力プロンプトが指定した文脈次第である。

表 2 で、1 から 5 と 22 は LLM が基本的な言語処理を実現していることを示す。6 から 10 は語用法や意味的な処理に関わる。11 から 14 は推論能力が限定されていることを示す。15 は逐語記憶、16 と 23 はプロンプトの役割が大きいことを表す。17 から 20 はプライバシーや公平性に関わる倫理的な負のリスクが生じること、21 はハルシネーションを指す。22 は出力内容がフェイク情報の場合に負のリスクになり得る。6 や 15 はスケーリング則から期待される振舞いと異なり、規模が大きくなると共に不具合が増す現象である。スケーリング則は好ましい結果を導くとは限らない。LLM を利用する立場から考えると、規模と素朴な好ましさはトレードオフ関係にあることを示唆する。

闇雲に大規模化が良いとするよりも、実現すべきアプリケーションやサービス機能の実現に必要な振舞いを示す LLM を利用すれば良いだろう。

A1.2.1.4 数と漢字の難しさ

表 2 に示されていないが、ここでは、2 つの点を追加する。第 1 は、「数」の取り扱いである。対象文書に現れる「数」は「数」のテキスト表現であり、別途、数学的な対象の「数」として解釈を適切に与えなければならない。しかし、通常のトークン化（図 9）は、「数」としての性質を考慮していないことが多い[Thawani 2021]。そこで、数値演算が関わるようなプロンプトの処理は期待通りにならず誤答を生じることがある。

第 2 は、日本語などで現れる問題で、特に、日本語では、漢字の異体字の取り扱いがある。異体字は、人名や地名に現れ、たとえば、「さいとう」の「さい」の漢字は、一説によると 85 種類ある[森岡 2017]。初対面の場で、「『さいとう』は、漢字で、どのように書くのですか」という質問が繰り返される。

トークン化（図 9）が対象とするデジタル文書は文字コード（Unicode）の並びである。ところが、手書きで 85 種類ある「さい」の漢字全てにユニークな文字コードが対応するわけではない。そのいくつかは同じ意味を表すとして同一の文字コードにマップされている。つまり、デジタル文書になった時点で、85 種類の違いを区別できない。これは、呼び方（音）で同姓同名であっても、元の手書き文書で異なる漢字を使っていた 2 人が、デジタル文書の取り扱い上、同一人物になることを示唆する。すると、LLM は、別人を同一人物とする情報に基づいたハルシネーションを生じる。

以上 2 つは、必ずしも LLM 自身の問題というわけではないが、利用者からみると、LLM が奇妙な振舞いを示すように見える。したがって、LLM の定性的な特徴として説明した。

A1.2.2 ハルシネーション

標準的な知識に照らして不適切な内容をハルシネーション（Hallucination）と呼ぶ[Siren 2023]。一般には、3 つの種類に分けることが多い。

- 入力が指定する文脈から逸脱する（Input-conflicting）
- 文脈内で矛盾が生じる（Context-conflicting）
- 事実に反する（Fact-conflicting）

この中で、LLM では事実に反するハルシネーションは、LLM が自動的に生成するフェイク情報ともいえ、読み手の人間側に誤解を生じさせることから社会的な問題が大きい。ハルシネーションの原因は、LLM の訓練学習から内容生成までの複数箇所に見出せる。

- 学習コーパスが相矛盾する情報（Misinformation）を含む
- 訓練学習過程で不適切な相関（Spurious Correlation）が生じる
- トークン予測機構の確率的な振舞いが原因となる

最初の 2 つは、従来、公平性に関連したバイアス除去を論じる際に考えた状況と同じで、学習データならびに訓練学習機構がブラックボックス化していることと関連する。一方、LLM でハルシネーションが生じる原因は、出力生成過程にもある。以下で直感的な説明を示す。

出力途中時点での後続予測では、その文脈内で出現確率の高いトークンが選ばれる。しかし、真実さ（Veracity）の確率が大きいとは限らない[Azaria 2023]。文脈上は整合しているが、事実に反する不適切なトークンの確率が大きいことがある。これ以降の生成過程では、この不適切なトークンを起点として整合性の高いテキストが続く。つまり、不適切なトークンの早期選択確定（Early Commitment）に引きずられて、雪だるま式にハルシネーションが膨れ上がる（Hallucination Snowballing）[Snowball 2023]。

上記からわかるように、LLM が事実に反するハルシネーションを生じる理由にはトークン列予測の仕組みが関わる。したがって、さまざまな対策が講じられているものの、完全にハルシネーションをなくすことは、実際上も原理上も不可能といえる。

A1.2.3 アラインメント

ハルシネーションへの対応には、アラインメントの考え方で、LLM を再学習する方法がある（図 11）。ハルシネーションを生じるプロンプトに対して「回答できない」または「知らない」といった正直な内容（Honest Samples）を出力するように、適切な訓練データセット（インストラクション学習データ）を準備し、教師あり学習の方法で微調整（Supervised Fine-tuning）する[InstructGPT 2022]。元の基本 LLM（bare LLM）に対して、アラインした LLM（aligned LLM）を得る。前者を得る過程を事前学習（pre-training）、後者に対して事後学習（post-training）と呼ぶことがある。

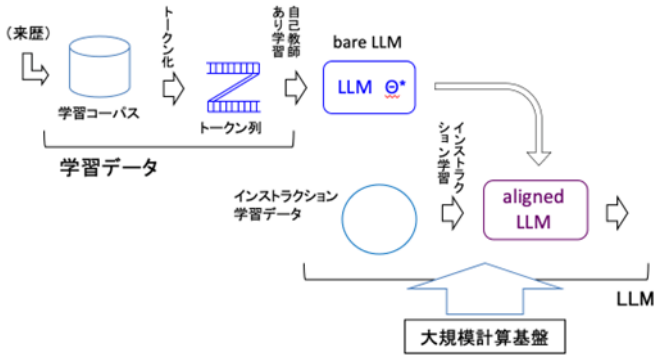

図 11 インストラクション学習

一般に、利用者の意図に合う内容を出力するように調整することを、アラインメント（Alignment）と呼ぶ。絶対的な基準はなく、アプリケーションに依存した対策法であるが、一般的な指針として、有用なこと（Helpful）、危険でないこと（Harmless）、正直なこと・誠実なこと（Honest）の頭文字をとった 3 H の考え方[3H 2021]による。LLM に対する具体的な方法として、RLHF（Reinforce Learning from Human Feedback）があり、開発者の価値判断を報酬（Rewards）とし、強化学習（Reinforce Learning）の方法で訓練する[RLHF 2023]。このハルシネーション対応には、次のような問題がある。

- インストラクション学習データ整備が困難
- 正直さと有用さのトレードオフ（Honest-Helpful Trade-off）

また、アラインメントが、アプリケーションの目的から見て期待通りかは、別途、評価が必要になる。AI 安全性の観点からアラインメントの標準データセットを整備する動きが活発になっている（A1.4.2 項）。

A1.2.4 訓練データ記憶

一般に、ニューラルネットワーク機械学習では、訓練データ記憶（Memorization）の問題がある[Feldman 2021]。分類学習タスクなど識別 AI を対象とし、特定データが学習に用いられたかを判定するメンバーシップ推定（Membership Inference）が可能なことが実験的に確認された[Shokri 2017]。

LLM では訓練に用いたトークン列の漏洩が生じる[Carlini 2021]。その理由は、学習パラメータに現れる情報（Parametric Knowledge）が訓練データを忠実に再現する逐語記憶（Verbatim Memorization）にある[Carlini 2023a]。これは、生成コンテンツが著作権違反になるかの議論において、依拠性肯定に関する技術的な説明となる。同様な記憶の問題

は画像生成 AI でも起きることが実験的に知られており[Carlini 2023b]、生成 AI 一般の問題といえる。

LLM の逐次記憶は、訓練学習機構ならびに後続トークン予測機構と密接な関係にある。一連のトークンが塊として強く結びつくとき、そのトークンの並びを再現するように学習パラメータが決まる。生成時には、その一連のトークンが再現される確率が大きくなる。また、学習データの中に、ユニークな定数のトークン列がある場合、この一連のトークンが塊として強く結びつくので、定数文字列は逐語記憶されやすい[Carlini 2019]。逐語記憶が直感に合った働きを裏付ける。

一般に、学習パラメータに現れる情報は、訓練に用いた学習データの経験分布に依存する。同じテキスト（トークン列）が何度も学習データに出現すると、そのトークン列が塊として強く結びつき逐次記憶されやすい。したがって、逐語記憶の対策は、学習データの事前処理によって、重複除去（De-duplication）することである[Lee 2022] [Nasr 2023]。

重複除去は、逐語記憶を減らし訓練データ漏洩抑止に効果がある一方で、事実と反するハルシネーションを増やす可能性がある。素朴に考えて、事実情報の出現頻度は、虚偽情報よりも大きく、学習コーパス中で重複して出現しやすい。事実情報が重複除去されると、相対的に虚偽情報の割合が増え、学習コーパスがそもそも抱えていた矛盾の度合いが増幅される。

このようなトレードオフ関係は、利用アプリケーションに即した評価が不可欠である。また、適切な方法を実施したかを知るには、事前処理内容を、学習データの来歴情報として明示しておく必要がある。

A1.2.5 学習コーパスの来歴

LLM においても、一般のニューラルネットワーク機械学習の場合と同様に、訓練データが訓練済み学習モデルの機能振舞いを決めるベースとなる。適切性が重要であり、矛盾する内容を含まないように訓練データを構築すれば良い。しかし、さまざまな応用に使うことを想定し、汎用性を目指して学習コーパスを整備する場合、適切性を担保した訓練データの構築作業は難しい。

LLM 構築では日常使っている文の用例を広範に収集することから、インターネット上でアクセス可能な文書をクローリング（Crawling）で自動収集する方法を採用する[C4

2020]。電子書籍、技術文書・特許文書、オンライン百科事典、ニュースメディア、オープンソースソフトウェア（Open Source Software, OSS）のプログラムなどが、収集、整理されて学習データに使われてきた。訓練データの特徴を簡明に文書化することは自明の作業ではない [Datasheet 2020] [C4 2021]。

このような文書のソースは多様で、内容も玉石混交であり、そのすべての信憑性が確認されているわけではない。膨大な文書の中には、矛盾する内容、相反する結論などが混在する。訓練データ整備段階の事前処理によって、適切な文書だけを選び出せば良い。しかし、何を適切とするかは、開発者の判断が関わる問題である。また、学習コーパス規模が大きくなると共に、膨大な作業になり現実的でない。学習コーパス内文書のソース、事前処理の判断基準などを整理し、訓練データの来歴を明らかにしなければならない。

LLM を含む生成 AI の訓練データでは、著作権に関わる問題が生じる。従来の識別 AI 開発では、著作物を学習データとして利用することは、フェアユースあるいは著作権の目的外使用として、著作権者の許諾を必要しないとされている。一方、生成 AI の場合、逐語記憶の問題から、訓練データの素材に極めて高い類似性を持つコンテンツが出力され、著作権侵害の可能性が生じる。著作権のあるデータの取扱いを訓練データの来歴として示すことで、依拠性の議論の客観性が高まると期待できる。

なお、ここでの著作権の話題は、生成コンテンツが著作物であるか否かとは異なる。現時点では、多くの国で、生成 AI は法的に著作者ではないし、生成コンテンツ単独では著作物にならない[Copyright 2023b]。

A1.3 プロンプトによる制御

A1.3.1 プロンプトの役割

LLM の基本的な振舞いは、入力プロンプトに対する出力テキスト生成である。プロンプトは字義上、処理促進のことを指す。LLM の技術分野では、入力として与えたトークン列（テキスト断片）そのものをプロンプト（Prompt）と呼ぶ。プロンプトはトークン列の生成条件を規定し文脈（Context）を指定する。つまり、トークン列は、与えたプロンプト下での条件付き確率で生成される。

LLM から期待通りの有用な出力を得るには、適切なプロンプトを与える必要がある。プロンプトの内容を構造化し、期待する出力が得られるように付加情報を与える。プロンプト中に与えた文脈の情報をヒントとして出力生成の方法を学習するように見えることから、文脈内学習（In-Context Learning）と呼ぶ[Brown 2020]。特に、少数例示学習（Few-shot Learning）は、質問と回答の組の例をヒントとして与える。このヒント情報を教師あり学習の訓練データと見なし、少数例示学習が教師あり学習をエミュレートすると考える[Dai 2023]。ヒント情報は実効的に訓練データと同じ役割を果たすと考える。

A1.3.2 プロンプトエンジニアリング

文脈内学習あるいは少数例示学習の方法によって、LLM の使い方が大きく変化した。最早、LLM は後段タスク構築の基盤モデルではない。単独の機能コンポーネントとして、事前訓練済みモデルの LLM を利用する方法が広まった。同時に、目的に応じてプロンプトを工夫する技法、プロンプトプログラミング（Prompt Programming）[Reynolds 2021]あるいはプロンプトエンジニアリング（Prompt Engineering）[White 2023] に関心が移った。多くの場合、入力プロンプトが期待する出力を生成するか、試行錯誤的探索的に LLM の挙動を調べる。

これまでに経験的に得られたプロンプト作成方針が整理され、5 つに分類されている[Bsharat 2023]。

- プロンプト構成のわかりやすさ
- 具体性と情報の補足
- ユーザーとのやり取りとユーザー関与
- プロンプト内容と表現スタイル
- 複雑なタスクとプロンプト書法

一方、プロンプトの具体的な表現は対象 LLM に依存する。用いる LLM ごとに、きめ細かなチューニング、つまり、試行錯誤が必要なことに注意する。

複雑なタスクを実行させる時、プロンプトを系統的な方法で構造化する方法が有効な場合がある。Chain-of-Thoughts（CoT）は、答えを導く思考過程の具体例をヒントとして示すプロンプト書法であり、簡単な推論を必要とする問題に有効である[Wei 2023]。Chain-of-Verification（CoVE）は複数のプロンプトを組み合わせて、LLM と対話的に、回答を得る系統的な方法である。対象 LLM がハルシネーションを生じるかを調べるこ

とを目的とて考案された。本質的に同じ内容を質問の仕方を変えて実行させ、一連の出力を総合して確認する[Dhuliawala 2023]。

一般には、LLM はブラックボックスの機能コンポーネントとして提供される。この時、プロンプトを通して、LLM の振舞い検査あるいは動作を確認することになる。通常のソフトウェアテスティングの場合と同様に、正常系に加えて例外系のテストケースを用いる。目的に合わせて工夫することで、さまざまなテストケースに対応するプロンプトを作成することになる。例外系テストケースでは、意図的に誤りを混入させたプロンプトを用いる方法を利用できる。

A1.3.3 外部情報の利用

学習パラメータに現れる情報は、訓練学習以降、更新されないことから、情報の新しさ（Newness）の問題がつきまとう。学習コーパス整備以降の新たな知見は、LLM に反映されていない。この新たな知見が、一般に正しさを客観的に確認可能な科学的な知識に関わると、LLM は最新情報を持たないことから、事実に反する正しくない応答を返すことになる。情報の新しさの問題は、事実に基づく情報（Factual Knowledge）の不備と関わり、ハルシネーションの原因になる。

文脈内学習の考え方によると、プロンプトは単なる出力の指示ではなく、期待するコンテンツの生成に必要な情報を与える、とする。プロンプトに新規の情報を埋め込めば、LLM が持たない外部情報を出力生成に利用できるだろう。LLM の学習パラメータに埋め込まれた内部情報をパラメータ情報（Parametric Knowledge）と呼ぶことがあり、これに対して、LLM の外部から与えた非パラメータ情報（Non-parametric Knowledge）という。この外部情報は内部情報と組み合わせられて、出力生成に用いられるように見える。ハルシネーション低減の効果を得られる場合がある[Shuster 2021]。

ところが、外部から与えた情報が目的とする内容出力に有用かは自明でない。外部情報と内部情報の関係によって、生成出力が影響を受ける。

- 外部情報が内部情報にない追加の新規内容
- 外部情報が内部情報の補足となる内容
- 外部情報と内部情報が相反する内容

1 番目の場合、出力生成に際して、外部情報が優先されることが望ましい。2 番目は外部情報が補足的な役割を果たし、期待出力の生成効率向上を目的とする。3 番目の場合、一方が事実で他方が虚偽となる知識衝突（Knowledge Conflicts）が生じる[Mallen 2023]。

LLM 内部に埋め込まれた情報は所与であり、これを前提とし、アプリケーションの目的に応じて、外部情報を選ぶ必要がある。しかし、訓練学習の結果である LLM の内部情報を調べることは難しく、実際の状況を知ることは容易ではない。

そこで、適切な入力プロンプトに対する LLM の出力を調べる外部観測の方法に頼らざるを得ない。プロンプトに適切な外部情報を埋込み、その生成出力から内部情報を推定する。仮に、訓練データに関わる情報が既知であれば、調査プロンプトを絞り込むことができ、実験作業の負荷を軽減できる可能性がある。訓練データの来歴を明示することの重要さを示す。

A1.4 基盤モデルとしての LLM（再考）

ここまで、経験的な知見から得られた特徴を概観し、直接 LLM を利用するプロンプトの役割をみてきた。以下、LLM をベースとして転移学習の方法で後続タスクを構築する場合、すなわち、LLM が狭義の基盤モデルの役割を果たす場合について補足する。

A1.4.1 ファインチューニング

転移学習の基本的な方法では、事前訓練済み学習モデル M0 に対して、目的となる後続タスク構築に必要な学習データセット D1 と、後続タスクを実現する学習モデル M1 を準備する。LLM の転移学習では、M0 と M1 の関係によって 2 つの代表的な方法がある。

- チェックポイント（Check-point）：M0 と同じ M1 に対して D1 の訓練を継続する方法。M0 が M1 を訓練する際の初期値の役割を果たす。
- ファインチューニング（Fine-tuning）：M0 に後続タスク固有の層を追加した M1 に対して D1 の訓練を行う方法。M0 に対応する層の学習パラメータは更新しない。

実務上、チェックポイント後にファインチューニングするというように 2 つの方法を組み合わせることがある。

後続タスクを実現した訓練済み学習モデルは、期待するアプリケーション機能を提供する。このようなドメイン特化 LLM をベースとして LLM 利用 AI システムを構築することで実用化する。一方で、振舞いの本質は LLM の特徴を示す。したがって、一般論として、LLM が持つ不具合をなくすことは難しい。提供アプリケーションの目的に応じて、適切な利用制限（Usage Restrictions）を明らかにすることが大切になる。

先に紹介したアラインメントでは、bare LLM を M0 とし、aligned LLM を M1 とする転移学習を強化学習の方法で行った（図 11）。bare LLM を基盤モデルとする時、このアラインメントという後続タスクに対応する aligned LLM は、ハルシネーション対策機能を施した LLM である。通常、公開される LLM は自明な不具合に対するアラインメントを実施済みである。LLM 利用ソフトウェア開発という場合、アラインされた LLM（aligned LLM）を用いるとする。

LLM 利用 AI システムの技術開発では、提供された LLM をブラックボックスの機能部品として活用することから、当該システムの目的あるいは要求仕様を基準として、ガードレールとなる LLM 周辺コンポーネントのデザインの役割が大きい。

A1.4.2 オープンなデータセット整備

ファインチューニングの方法は、「ソフトウェア-WITH 開発」（3.2.2 節）で使われ、適切な学習データ整備が成否に関わる。実際、アラインメントに際して、インストラクション学習データセットを整備した。最近では、応用セクターごとに標準的な学習データあるいは評価ベンチマーク用データを整備する取組みが進められている。

AI ガバナンスの文脈で論じられている「安全性」は、AI がもたらす社会的倫理的なリスクの低減に関わる。不適切な機能振舞いを抑止するように、注意深く構築されたデータセットを整備することが大切である。社会的倫理的なリスクと関わることから、広範なステークホルダーの協業による学習データ収集・整備が、AI 安全性に関わる国際的な活動として進められている。

提供された LLM が不適切な機能振舞いを示さないか、どのような状況（プロンプト入力）によって監獄破りが生じるかは、第 3 者によるレッドチーミング活動によるところが大きい。また、収集されたデータに適切なアノテーションを与えてデータセットを整備する過程では、さまざまな社会背景・文化背景からのステークホルダーによる協業が大きな役割を果たす。

AI 安全性という公益性から見て、このようなデータセットは共通領域の技術であり、実際、オープンな活動が活発になっている[Gallegos 2024]。国内でも大規模言語モデルの「安全性」に関わり、「要注意回答」を集めたデータセットの研究開発が進められている 。

実際、「機械学習-FOR 開発」では、「安全性」に関わるデータセットの整備と利用が重要である。言語モデルの開発時に、オープンな標準データセットを用いて、「安全性」を向上する作業を実施している事業者もいる。文献[OpenAI 2024]は、用いた「安全性」の向上に関わるデータセットを一覧している。

付録2 大規模言語モデル利用ソフトウェアシステム

本付録章では、LLM 利用ソフトウェア開発を対象として、エンジニアリング上の工夫（A2.1 節）、コンポーネント指向開発の方法（A2.2 節）を整理する。

A2.1　エンジニアリングの工夫

LLM（アライン LLM）単独の利用では、プロンプトの工夫が大きな役割を果たすものの、ハルシネーションや訓練データ漏洩といった基本的な不具合の回避が難しい。また、アプリケーションサービス構築に必要な機能が不足する。たとえば、出力結果をファイルに格納するなど、LLM 利用 AI システムの動作環境に副作用を与える機能を考える。LLM は、入力プロンプトに対して出力を求めるという関数的な振舞いを示す。ファイル書き込み操作を実現することはできず、LLM 外部でのプログラミングを必要とする。LLM を機能部品として用いるソフトウェアシステムをエンジニアリング上の工夫で実現する。

これまで、さまざまな LLM 利用アプリケーション開発の知見から、LLM の不備を補うように外部で工夫する機能が整理されてきた。主な 5 つの観点に対して、典型的な解決パターン（Solution Patterns）が整理されている。

4. プロンプト補完：出力生成の制御に必要な情報をプロンプトに追加
5. 出力フィルター：生成内容が期待通りかを判定
6. データ最新性：新しい情報を取り込み出力生成に活用
7. オープンツール：特定機能を実現した外部ツールと連携（プラグイン）
8. 履歴メモリ：出力結果をフィードバックし入力

一般に、オブジェクト指向デザインパターンでは、複数パターンを組み合わせたデザインからプログラムを作成する方法が実践されてきた。LLM 利用ソフトウェア開発でも複数パターンを適宜カスタマイズし組み合わせることは同じであるが、LLM の典型的な解決パターンは粒度が大きい「コンポーネント」に対応する。LLM を抽象化してコンポーネント（言語モデルコンポーネント）とみなすと、LLM 利用ソフトウェアのデザインは、自然言語テキストのメッセージをやり取りするコンポーネントモデルとして表現できる。

仮想的な LLM を表す言語モデルコンポーネントへの直接の入力はプロンプトである。他方、HMI からの入力情報の取得や外部ツール連携では、プログラムを作成して情報を

加工する必要がある。簡単な例では、たとえば、少数例示プロンプトの一部をプログラム処理したデータに置き換えるなどがある。そこで、プロンプト中に「空き場所（Place Holder）」を配置したプロンプトテンプレートを準備しておくと、コンポーネントモデルに基づくデザインの見通しが良くなる。

従来からのソフトウェアデザインならびにプログラミングを組み合わせて、必要な機能を構築することになる。LLM 利用ソフトウェアシステムの開発に際して、プロンプトの工夫とプログラムの「すり合わせ」が必要で、プロンプトとプログラムをコデザインする。

A2.2 コンポーネント開発

コンポーネントアーキテクチャをベースとする WITH-LLM の開発技術を概観する。

A2.2.1 トリプレットパターン

LLM の基本的な使い方は、入力プロンプトにしたがった内容を出力することだった。LLM 利用アプリケーションシステムでは、直接、LLM を使うことは稀で、LLM を言語モデルとした基本形（図 12）の構成をとる。ここで、言語モデルは LLM を仮想化したコンポーネントであり、具体的な LLM をプラグインすると考える。

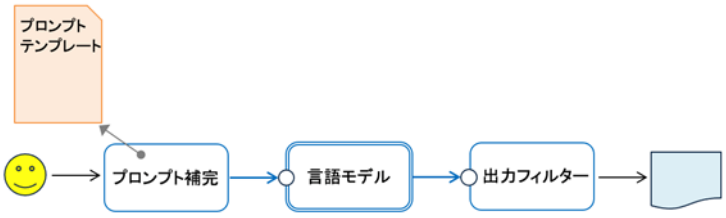

図 12 基本形となるトリプレット

この時、開発対象の中心は期待する出力を得るプロンプトを求めることである。機能要求を具体化し、探索的なプログラミング（Exploratory Programming）[Sheil 1983]と同様に、試行錯誤を繰り返す。プロンプトプロトタイピング（Prompt Prototyping）である。

プロンプトを定型化できると、プロンプトの骨格を決める共通部分を抽出したプロンプトテンプレートを作成し、可変情報をユーザー入力から埋めて言語モデルへの入力プロンプトを生成するプロンプト補完コンポーネントを導入する。プロンプト補完の具体

的なコンポーネントには、自動化のレベルに応じて、さまざまな方法が考えられる。たとえば、

- 埋込みプロンプト：事前に準備したプロンプトテンプレートにユーザー入力情報を埋め込む
- 生成的プロンプト：ユーザー入力からプロンプトを自動生成する

がある。ここで、プロンプト補完コンポーネントが利用するプロンプトテンプレートは外部情報の一種と考えられる。

次に、出力フィルターコンポーネントを導入し、言語モデルの出力を加工する機能を連結する。出力フィルターの具体的なコンポーネントは、加工の目的によるが、所与の基準から不適切な情報の出力を抑止する「情報選択」が有用である。また、出力フィルターコンポーネントによっては、変換の方法や不適切な情報の基準を、外部から与える方法を採用する。たとえば、事実と異なる知識であるかの判定には外部情報が不可欠である。ガードレール（A1.4.1 節）の実現がある。

以上のコンポーネントを逐次結合（Sequential Composition）することで、LLM 利用の基本形を仮想化したトリプレットパターン（図 12）を得る。特に、外部情報を参照するプロンプト補完や出力フィルターを用いることで、外部情報を活用したトリプレットパターンの機能振舞いを実現できる。

A2.2.2 RAG パターン

学習データ最新性の問題を解決するパターンとして、外部データから必要な情報を動的に獲得する RAG（Retrieval-Augmented Generation）がある[RAG 2021] [Gao 2024]。

RAG を構成する検索器（Retriever）は、問い合わせ情報をキーとして、関連する外部データを獲得する機能を実現する。検索器の具体的な構成方法は、機能要求からソフトウェアデザインを具体化する方式設計で決定する。問い合わせ情報と候補データ断片の類似性から検索する方法、検索器コンポーネントを訓練学習で獲得する方法[REPLUG 2023]など、さまざまな手法がある。

RAG デザインとして、知識グラフを活用する方法（グラフ RAG）がある。基本的な RAG ではフラットな文書として扱うことから、外部データが構造のある文書であっても構造に関わる情報が失われる。そこで、文書構造を反映した知識グラフと対象文書デ

ータの両方を用いて、RAG の検索機能を高度化する。文書が構造を持たない場合であっても、文書データから知識グラフを抽出して利用する方法を考えることができる。

グラフ RAG では、LLM との関係が多様になる[Xu 2024]。そもそも、RAG は LLM に外部情報を補う役割を果たしていた。グラフ RAG も同じで、LLM に対して「知識埋め込み」機能を果たす。また、文書から知識グラフを構築する際に LLM を利用することができ、LLM を「知識抽出」機能に用いる。さらに、知識グラフをグラフ DBMS で管理するアーキテクチャを採用する場合、問い合わせ情報から当該 DBMS の問い合わせ式を生成する「言語変換」機能に用いる。グラフ RAG をデザインする上で、自然言語処理を実現する LLM の使い方を考える。

RAG パターンは外部データを取り込む仕組みを実現するが、一方で、獲得したデータの内容が妥当かには触れない。その外部データが目的達成に有用かの評価が必要になる。経験的には、適切な外部データによってハルシネーション低減の効果を得ることができる[Shuster 2021]。一方で、知識衝突などに関わるデータ品質を別途、評価する必要がある[Xie 2023]。

A2.2.3 オープンツールパターン

オープンツールパターンの目的は、言語モデルと外部ツールの間での情報交換を実現することである。さまざまな機能を持つ外部ツールを利用することで、LLM 利用ソフトウェアの機能を拡大できる。たとえば、インターネット検索エンジンや特定応用のシミュレーションツールとの連携がある。外部ツールとの入出力データ形式が合致していること（インタフェース整合性）、外部ツール出力から必要な情報を抽出すること（出力フィルター）、に注意する。

オープンツール言語モデルによって外部ツールとの連携が可能になると、LLM の応用方法が広がる。ReACT パターン[Yao 2023]は、CoT などによる処理手順を制御するプロンプトと外部ツールのアクションを組み合わせる。手順の制御とアクション実行を自律的に繰り返す。

LLM エージェントシステム（2.1.1 節）は、この ReACT パターンを拡張することで実現可能になる。エンドユーザーが与えたプロンプトをゴールとする。このゴール達成に必要なサブタスクを立案し、各々のタスクを実行する外部ツールを適切に選び実行する自律的なソフトウェア基盤である[AutoGPT 2023] [Shen 2023] [Song 2023]。現時点では、

実験的な技術と考えた方が良いという考察もある[Santos 2024]。システム品質には、自律的なオーケストレーション（Orchestration）の仕組みを実現するソフトウェア計算基盤の性質（Safety や Liveness）が影響する。

A2.2.4 システム全体のデザイン

上記に示したパターンを組み合わせることで、LLM 利用ソフトウェアのコンポーネントアーキテククチャを整理することができる。この考え方は、都市計画デザインに関するパターン言語の考え方[アレグザンダー 1974]をオブジェクト指向ソフトウェアに応用した「パターン生成アーキテクチャ（Patterns Generate Architecture）」のデザイン手法[Beck 1994]を LLM 利用ソフトウェア開発に応用したものである。

LLM 利用ソフトウェアシステムは、このようなパターン指向コンポーネントアーキテクチャのみで構成されるわけではない。LLM 利用ソフトウェアはユーザーとのインタフェースを有し、HMI（Human-Machine Interaction）のデザインが重要になる。GUI を含む HMI の良し悪しはユーザーの満足性に寄与し、利用時の品質に影響する。

一般に、HMI は複数の GUI コンポーネント（コンセプトと呼ぶ[ジャクソン 2023]）から構成されることから、HMI デザインをコンセプトアーキテクチャとして表現する。各コンセプトはユーザーと直接情報交換し、各々がプロンプト断片を有すると考えられる。プロンプト補完コンポーネントはプロンプト断片を集約する役割を果たす。同様に、出力フィルターコンポーネントは生成出力メッセージテキストを分割し、コンセプトにフィードバックする。このようなメッセージテキストの分割は、コンセプトデザインのモジュラリティを反映する（図 13）。

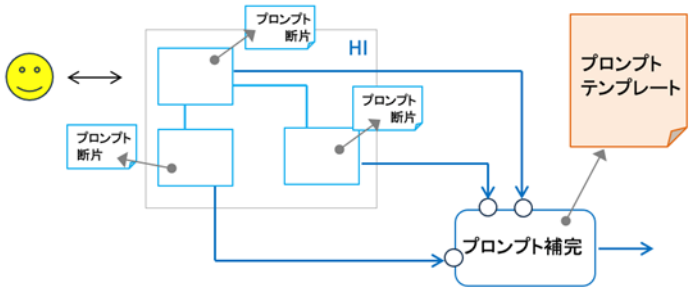

図 13 HI とプロンプトの集約

以上見てきたように、HMI を含む LLM 利用ソフトウェアシステムは、2 つの異なる役割を与えられたアーキテクチャから構成される。

- LLM を中心に構成したコンポーネントアーキテクチャ
- HMI デザインに関わるコンセプトアーキテクチャ

両者は、異なる視点から分析設計した結果である一方で、ひとつのソフトウェアシステムを表す。したがって、2 つのアーキテクチャは互いに関連する。視点の転換（Viewpoint Shift）によって結びつくが、独立に分析設計を進めることができる。

A2.2.5 開発の流れ

HMI を含む LLM 利用ソフトウェアシステム開発の大まかな流れを示す。

1. 要求分析
2. アーキテクチャのデザイン
3. コンポーネントのデザインと開発
4. コンセプトのデザインと開発
5. システム統合と利用時の検査

全体として、従来ソフトウェアと同様な V 字型開発モデル[中谷 2025]として整理できる。

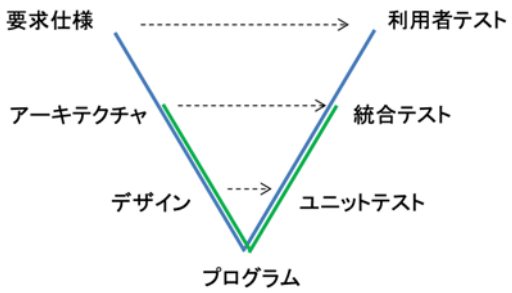

図 14 V 字型開発モデル

次に詳細に見ていく。要求分析は主として 2 つの作業に分けられる。

1. 要求の取り出し（Elicitation）と形式化または整理（Formalization）
   従来ソフトウェアの要求分析フェーズで、顧客要求をビジネスソフトウェアの要求仕様として整理する作業に相当する。
2. プロンプトのプロトタイピング（Prompt Prototyping）

想定する典型的な入力を、使用する LLM に即したプロンプトとして表現できることを確認する作業で、従来の要求分析における実現予見の一例に相当する。プロンプトエンジニアリングなどの知見を参考にし、本質的に試行錯誤を伴うラピッドプロトタイピングによる。

アーキテクチャ段階は、2 つのアーキテクチャ視点に関わる分析とデザインを進める。

1. アーキテクチャ分離
   システム全体は、LLM を中心に構成したコンポーネントアーキテクチャと HMI デザインに関わるコンセプトアーキテクチャからなる。標準的な MVC アーキテクチャでは、前者は M（モデル）で後者は VC（ビューとコントローラ）に対応する。
2. コンポーネントアーキテクチャ
   パターン生成アーキテクチャの手法で、機能要求の実現に必要なコンポーネントを選び組み合わせる。
3. コンセプトアーキテクチャ
   HMI デザインの実現に必要なコンセプトを定義し組み合わせる。
4. 擦り合わせ
   メッセージテキストが 2 つのアーキテクチャ視点を媒介する。コンセプトのモジュラー性から必要となるプロンプト集約の検討を中心に、2 つのアーキテクチャ視点の対応関係を調整する。必要であれば 2 や 3 に戻って検討する。

デザインとプログラム段階は、2 つのアーキテクチャを並行して進め、ユニットテスト相当の検査を行う。コンポーネントならびにコンセプトに固有の機能に関わる。コンポーネントに着目して入出力メッセージテキストの関係が期待通りかを検査する。コンポーネントアーキテクチャのプログラム実現する際には、既存の再利用可能なフレームワークを利用すればよい [38]。また、コンセプトアーキテクチャのプログラム実現では、GUI フレームワークなどを活用する。

コンポーネントならびにコンポーネントアーキテクチャの検査では、利用するパターンごとに検査を工夫する。RAG パターンやオープンツールパターンは外部の情報交換に関わる機能振る舞いが意図通りかを確認することから、検査の手間が大きくなる。

次に、2 つのアーキテクチャ個別に、全体の流れが期待通りかを検査する。正常系テストに加えてストレス機能に対応する反実仮想メッセージテキストのテストケースを検査する。最後に、統合テスト段階は、2 つのアーキテクチャからシステム全体を構成

[38] たとえば LangChain。

し、総合的なテストを行う。利用時の評価に相当する統合テストを通して、ユーザーに課す利用制限事項を整理する。

A2.3 品質特性とコンポーネント

LLM 利用 AI システムの開発に際して、再利用部品である LLM の選択、プロンプトプロトタイピング、コンポーネントならびに HMI のデザインを進める。この過程で考慮すべきコンポーネント品質およびデータ品質との関係を概観する（表 3・表 4）。

表 3 デザインで考慮するコンポーネント品質

		再利用部品	プロンプト	主要コンポーネント				システム
		言語モデル		RAG	外部連携	フィルター	HMI	（計算基盤）
要求満足性	機能要求満足性		レ				レ	レ
	品質要求満足性							レ
信頼性	成熟性		レ	レ	レ	レ	レ	レ
	ロバスト性	レ	レ					レ
	出力一貫性	レ	レ				レ	
	進行性		レ					レ
	可用性		レ		レ			レ
	耐故障性							レ
	回復性							レ
インタラクション性	適切度認識性						レ	レ
	アクセシビリティ		レ				レ	
	可制御性						レ	
	説明性		レ	レ		レ	レ	レ
	習得性						レ	
セキュリティ	アクセス制御性		レ	レ	レ		レ	
	介入性						レ	レ
	真正性		レ		レ		レ	
	責任追跡性							レ
	否認防止性					レ		
安全性	入力制限性	レ	レ				レ	
	フェイルセーフ性			レ	レ	レ		レ
プライバシー		レ		レ	レ	レ		
公平性		レ		レ	レ	レ		
性能効率性	時間効率性	レ	レ	レ				レ
	資源効率性	レ		レ				レ
移植性			レ	レ	レ		レ	レ
保守性	モジュール性		レ	レ	レ	レ	レ	レ
	修正性		レ					レ
	試験性							レ

表 4 デザインで考慮するデータ品質

		再利用部品	プロンプト	主要コンポーネント				システム
		言語モデル		RAG	外部連携	フィルター	HMI	
データ点	正確性	レ	レ					
	完全性	レ						
	一貫性	レ		レ				
	信憑性	レ						
	最新性	レ		レ				
	精度	レ						
	利用性	レ		レ				
	標準適合性	レ		レ				
データセット	代表性	レ	レ	レ				
	重複性	レ		レ				
	標本選択性	レ						
	一貫性	レ	レ	レ				
	来歴性	レ		レ				
	最新性			レ				
データセット組み合わせ	文脈整合性			レ				
	時制整合性			レ				
	操作整合性	レ						

- 再利用部品の欄は、LLM 選択の際に、当該 LLM に対して期待する品質観点を表す。印付けした観点について適切な品質を持つことを LLM に付された AI BOM 等を参照して確認する。
- プロンプトの欄は、印付けした観点について適切な品質を持つことを、プロンプト構築（プロンプトデザインならびにプロンプトプロトタインピング）の際に確認する。プロンプト構築での目標品質観点となる。
- 主要コンポーネントの欄は、印付けした観点について適切な品質を持つことを、当該コンポーネント開発の際に確認する。目標品質観点となる。
- システムの欄は、印付けした観点について適切な品質を持つことを、LLM 利用 AI システムに対して総合的に確認する。目標品質観点となる。

A2.4 品質特性と SQuaRE との関係

本ガイドラインの品質特性は、SQuaRE（25010・25012・25059）が示す一般的で中立な品質モデルに、「デザインからの品質」の考え方に基づく解釈を与えて整理した。SQuaRE 品質モデルの基本的な考え方をベースとして、LLM 利用 AI システムの技術的な特徴を考慮している（表 5・表 6）。煩雑さを避けることから、大きく異なる項目についてのみ説明する。

表 5 コンポーネント品質と SQuaRE

	SQuaREの読み替え・SQuaREとの違い
要求満足性	機能適合性を再編成し、要求の満足度合いとして整理
機能要求満足性	＊機能要求の満足度合い
品質要求満足性	＊機能外要求の満足度合い
信頼性	
成熟性	ソフトウェア信頼性の達成度合い
ロバスト性	モデルならびにプログラムのロバスト性
出力一貫性	＊LLMで顕在化
進行性	＊自律実行する計算基盤で顕在化
可用性	
耐故障性	
回復性	
インタラクション性	
適切度認識性	
アクセシビリティ	
可制御性	＊外部からの制御インタフェース
説明性	
習得性	
セキュリティ	
アクセス制御性	機密性・インテグリティを統合
介入性	＊作動中システムへの外部介入
真正性	
責任追跡性	
否認防止性	
安全性	振舞い利用制限が前提
入力制限性	
フェイルセーフ性	
プライバシー	＊プライバシーに関わる機能を対象（セキュリティと同じ発想）
公平性	＊公平性に関わる機能を対象（セキュリティと同じ発想）
性能効率性	
時間効率性	
資源効率性	
移植性	LLMの置き換えが中心
保守性	
モジュール性	
修正性	
試験性	

表 6 データ品質と SQuaRE

		SQuaREの読み替え・SQuaREとの違い
データ点		SQuaRE 25012
	正確性	
	完全性	
	一貫性	
	信憑性	
	最新性	
	精度	
	利用性	機密性をデータ利用権の立場から再整理
	標準適合性	
データセット		＊SQuaREでは未考慮
	代表性	
	重複性	
	標本選択性	
	一貫性	
	来歴性	
	最新性	
データセット組み合わせ		＊SQuaREでは未考慮
	文脈整合性	
	時制整合性	
	操作整合性	

付録3 品質特性と LLM 関連 AI セキュリティ

本付録章では、[OWASP2023]を基に LLM に特有なセキュリティリスクのタイプに対応した、LLM 利用 AI システムのデザインからの対策を検討する。新たに発見されるリスクタイプへの対応が不可欠であり、本付録の内容が網羅的な対策を述べるものではない。

A3.1 デザインからの対策

LLM に特有なリスクタイプに関して、OWASP（Open Worldwide Application Security Project）が、LLM リスクのトップ 10 を整理している [39] [OWASP 2023][Santos 2024]。

各リスクの影響を低減するように、LLM 利用 AI システムをデザインするとき、さまざまな開発運用フェーズで行うべき対策として整理する（表 7）。WITH-LLM 開発フェーズを中心とし、サプライチェーン、FOR-LLM、USE に対しての期待を、提供情報あるいは期待する対策事項という観点で整理する。WITH-LLM 開発では、このような情報や講じられた対策を前提として、コンポーネントアーキテクチャのデザインによる対策を考える。

[39] 表 7 は 1.0 版に基づく。OWASP の LLM リスク・トップ 10 は 2025 年版[OWASP 2024]が公表されている。「システムプロンプト抽出」（A3.2 参照）が新しい問題として顕在化している。

表 7 OWASP の LLM リスクと対策

OWASPリスク	Supply Chain	FOR-LLM	WITH-LLM	USE
プロンプトインジェクション Prompt Injection		レ	レ	レ
安全でない出力の取り扱い Insecure Output Handling		レ	レ	
訓練データの毒化 Training Data Poisoning	レ	レ	レ	
モデルDoS Model Denial of Service			レ	レ
サプライチェーン脆弱性 Supply Chain Vulnerabilities	レ	レ	レ	
機微情報の漏洩 Sensitive Information Disclosure		レ	レ	
安全でないプラグインデザイン Insecure Plugin Design			レ	
過剰な代行 Excessive Agency			レ	レ
過度の依存 Overreliance			レ	レ
モデル搾取 Model Theft	レ	レ	レ	レ

また、LLM 利用 AI システムの開発に際して、各 OWASP リスクの低減に寄与するコンポーネントならびに当該コンポーネントのデザイン時に考慮すべき品質観点を整理する（表 8）。LLM 利用 AI システムの対策に関わるデザイン例と考えることができる。

表 8 OWASP リスク対策デザインの品質観点

OWASP リスク	デザイン対象（主な品質副特性）
プロンプトインジェクション	プロンプト（機能要求満足性・説明性） フィルター（成熟性） システム（責任追跡性） 外部連携（アクセス制御性・真正性）
安全でない出力の取り扱い	フィルター（成熟性） プロンプト（機能要求満足性・安全性）

訓練データの毒化	RAG（アクセス制御性） プロンプト（真正性）
モデルの DoS	システム（責任追跡性）
サプライチェーン脆弱性	成果物（真正性）
機微情報の漏洩	データ保護加工（利用性） RAG（プライバシー・アクセス制御性） 外部連携（プライバシー・アクセス制御性） フィルター（プライバシー）
安全でないプラグインデザイン	外部連携（可用性・アクセス制御性・真正性）
過剰な代行	システム（要求満足性・進行性・介入性） HMI（可制御性・介入性）
過度の依存	システム（機能要求満足性） フィルター（否認防止性）
モデル窃取	システム（責任追跡性）

以下、１０の OWASP リスクについて詳しく見ていく。

「プロンプトインジェクション」は、プロンプトが改変されて不適切な情報が LLM に入力されることである。不適切なプロンプトを直接入力する場合（直接攻撃）と、LLM への入力プロンプトを構築する外部サービスを改変する場合（間接攻撃）がある。

- 利用する LLM に基本的なアラインメントが施されていること（FOR-LLM）
- 入力から LLM を保護するようにプロンプトをデザインすること
- 不適切な生成コンテンツを検知しシステムからの出力を抑制すること
- 間接攻撃への対策として外部連携サービスを確認すること
- システムからの出力の根拠を示せること
- 運用時の利用監視を併用すること（USE）

「安全でない出力の取り扱い」は、想定外の LLM 出力を後段アプリケーションが適切に処理できないことである。

- 利用する LLM に基本的なアラインメントが施されていること（FOR-LLM）
- 後段アプリケーションの入力仕様を明らかにすること
- LLM 生成コンテンツ対する出力フィルターを導入すること。出力可否判断の根拠は、予め決められた基準による方法を、外部データを参照して決めても良い。
- プロンプトのデザインによって不適切な出力を抑止すること（間接的な対策）

「訓練データ毒化」は、「訓練データ」を改変することで当該データを学習したモデルの機能振る舞いを変更することである。「訓練データ」をどのように解釈するかで対策が異なる。LLM をファインチューニングする場合、訓練データがセキュアに格納保持されていることによって、このリスクを低減する。

次に、文脈内学習の方法を採用する場合を考える。訓練データと似た役割を果たすデータは、LLM に入力されるプロンプトの内容として現れ、2 つの場合を考えることができる。

- ユーザー入力プロンプトの場合、「プロンプトインジェクション」に準じる
- 外部データを取り込む RAG の場合、外部データがセキュアに格納保持されていること

「モデル DoS」は、LLM 使用に関わる計算資源を不適切に消費することで正常な運用を阻害することである。

- 運用監視によって利用状況の統計的な異常を検出すること（USE）
- システムが利用ログを保持管理すること

「サプライチェーン脆弱性」は、LLM 利用 AI システムの分業開発に際して、中間成果物・再利用可能コンポーネントを配布する際に生じ、成果物の品質を劣化させることである。

- 成果物の真正性を保証すること（サプライチェーン）
- LLM の来歴を保証すること（FOR-LLM）

「機微情報の漏洩」は、LLM 利用 AI システムが保持する機微情報の漏洩を防止することである。「機微情報」の在処の違いによって個別に具体的な対応策を考える。

- 基本はデータの保護加工（データ匿名加工）を行うこと
- LLM 訓練に用いた学習データが適切に保護加工されていること（FOR-LLM）

WITH-LLM の入力となるデータは、RAG 利用やプラグイン利用を含めると、ユーザー入力データ・RAG による補完データ・外部ツールからの入力データなどがある。

- ユーザー入力データを WITH-LLM システム内部で保持管理する場合、格納前に保護加工を施すこと
- RAG で取り込む外部データは RAG のデータ構築の段階で保護加工を施すこと
- 外部ツールからの入力データは、ユーザー入力データの取り扱いに準じる。ま

た、利用ツールの品質面で「安全でないプラグインデザイン」と関わる。

システムプロンプトはビジネスロジックを表すことから機微情報となる場合があり、「プロンプトインジェクション」と関わる。

「安全でないプラグインデザイン」は、連携する外部ツールが不適切な振る舞いを示すことである。従来の分散ソフトウェアで外部サービスを利用する際の品質問題と同等である。

- 外部ツールの真正性・アクセス制御性を確認すること
- 外部ツール出力のフィルターで確認すること

「過剰な代行」は、LLM 利用 AI システムが想定外の自律性を示ときに生じる。

- 要求仕様で、自律性の暴走を避けるようにシステムの振舞いを制限すること
- 利用の指示（Instruction of Use）を明示すること。これを遵守すること（USE）
- 運用監視によって想定外の振る舞いを検知すること
- システムを強制停止する介入性の仕組みを実現すること

「過度の依存」は、利用者が LLM 出力を無批判に受け入れることの危険性に関わる。

- エンドユーザーに対する利用上の注意、社会的弱者への利用制限などを通して、過度の依存が生じないような制度・運用面からの対策を講じること（USE）
- 第 3 者への影響を軽減する上で、コンテンツの否認防止性を考慮すること

「モデル窃取」は、訓練済みモデル自身あるいは等価な機能振る舞いを再現可能な情報が漏洩することである。後者について、繰り返し問い合わせによって情報を獲得する方法が知られている。

- 成果物（訓練済みモデル）をセキュアに保持管理すること（サプライチェーン）
- 訓練済みモデルの出力情報を制限・擾乱付加すること（FOR-WITH）
- 繰り返し問い合わせを避ける上で利用頻度の制限といった運用面の方策を講じること（USE）
- 利用状況を把握すること

A3.2　プロンプト周辺

プロンプトはエンドユーザーと LLM 利用 AI システムのインタフェースとなる重要な要素である。使い方によってはプロンプトインジェクションがシステムへの脅威とな

る。ここでは、標準的なプロンプト構成方法（図 15）を想定し、プロンプトに関わる問題のいくつかを整理する。

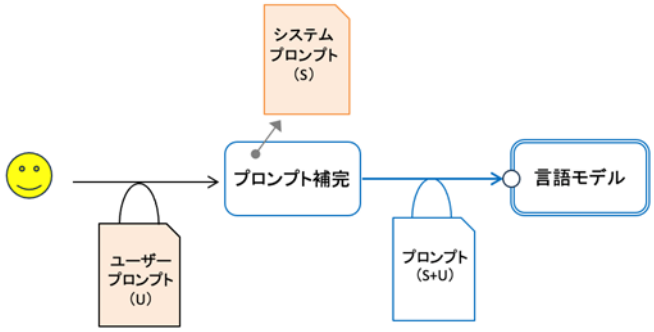

図 15 システムプロンプト

素朴には、ユーザープロンプトが LLM への入力となる。一方、標準的なシステム構成方法では、プロンプトにモジュラー化する。図 15 は、システムプロンプトとユーザープロンプトに区分けする方法を示した。ユーザーがさまざまな問い合わせを行う状況で、問い合わせに共通する情報をシステムプロンプトとして保持しておき、ユーザーが問い合わせごとに入力する情報を表すユーザープロンプトと組み合わせることで、LLM への入力プロンプトを合成する。

共通情報のシステムプロンプトは WITH-LLM の開発成果物であり、ビジネスロジックを実現すると考える。問い合わせごとに個別に与えられるユーザープロンプトは USE 時の入力情報である。WITH-LLM の品質マネジメントに関連して、2 種類のプロンプトの論理的な関係が問題となる。通常の使い方では、システムプロンプトが表すメタレベルでの制御下で、その共通情報とユーザープロンプトが表す個別のインスタンス情報を組み合わせて、LLM への入力プロンプトの全体を構成する。この時、ユーザープロンプトとして入力した情報が適切に取り扱われるかは、「機微情報の漏洩」と関わる問題である。

システムプロンプトで期待するユーザープロンプトが入力されるかは、USE 時の状況による。一般には、不具合を誘発する「プロンプトインジェクション」の可能性を排除できない。また、RAG を用いる場合、簡単には、ユーザー入力と RAG から得た情報を、図 15 のユーザープロンプトに対応させて考えればよい。RAG を併用したプロンプトインジェクション攻撃は、RAG によって取り込んだデータに、ユーザーが意図しない不適切な情報が埋め込まれている場合である。

対策として、システムプロンプトのメタレベル制御を利用することで、ユーザープロンプトが適切か否かを判断し、不適切な場合に、ユーザープロンプトを無効化（スキップ）することができる場合がある。この問題を一般化すると、プロンプトがモジュラー分割されている場合に、モジュール間に階層関係を導入すること、階層上位のモジュラープロンプトがメタレベルの役割を果たし、下位のモジュラープロンプト実行を制御すること、を如何にして実現するかである。Instruction Hierarchy[Wallace 2024]は、モジュラープロンプト間の強弱関係を取り込んだ LLM を、アラインメント微調整の方法で得る。監獄破り（Jailbreaks）を低減するアプローチになる可能性を持つ。

チャットボットのような会話型 AI では、システムプロンプトに会話の流れを誘導するようなメタレベル情報を持たせることで、ユーザーから情報収集するような仕組みを実現することができる。対応はエンドユーザーに委ねられる。たとえば、ユーザーが不穏な気配を感じるなどによって、敵対的なチャットボットが欲する情報を与えなければ良い。

ところが、システムプロンプトのメタレベル制御を用いることで、敵対的なチャットボットを構築できる。また、プロンプトインジェクションによって、敵対的な会話を誘導するシステムプロンプトを導入することが可能な場合もある[Greshake 2023]。

WITH-LLM で考えるべきことは、開発対象の LLM 利用 AI システムを敵対的な外部から保護すること、エンドユーザーや第 3 者の権利を保護することである。一方、エンドユーザーから見て、LLM 利用 AI システムが敵対的な振る舞いを示さないことの確認も重要な課題となる。特に、上記で見たように、チャットボットのような会話型 AI では、システムプロンプトが敵対的な振る舞いを誘導する恐れがある。システムプロンプトを開示することで、敵対性がないことを主張できる。一方で、システムプロンプトにシステムの目的依存の技術ノウハウ、利用している LLM の特徴を考慮したノウハウがあり、会話型 AI の開発者からすると、システムプロンプトをビジネス上の機密情報（トレードシークレット）として保護したい。つまり、開示したくない。

このシステムプロンプト保護の問題は、技術的には、エンドユーザーに隠されているプロンプトを抽出できるかである。このプロンプト抽出（Prompt Extraction）[Zhang 2024]の基本的なアイデアは、ユーザープロンプトにメタレベル情報を与えてシステムプロンプトを制御下に置くことで、システムプロンプトと等価なテキストを逐語出力させる。会話型 AI に対して、敵対的なやり取りを繰り返すことで、システムプロンプトを推測する。このプロンプト抽出は、「安全でない出力の取り扱い」の問題であり、出力フィ

ルターによる対策を考えることができる。一方で、この防御が常に有効とは限らない[Zhang 2024]。

LLM からの情報流出（機密情報の漏洩）は、これまでは、訓練データに関わる権利侵害の問題として議論されてきた。LLM 利用 AI システムが、自身に埋め込まれたシステムプロンプト（図 15）と共に流通する状況を考えると、プロンプト抽出攻撃への対策がビジネス上、重要になる。

コラム：AI ガバナンスと品質マネジメントの役割の違い

システムの機能要求が、そもそも包括的基本権[芦部 2023]に関わる問題などを含むことがある。例としてダイレクトマーケティング（Direct Marketing）のサービスを実現するシステムを考える。推奨アルゴリズム（Recommendation Algorithm）の技術を応用して、個人向けにカスタマイズした広告情報をオンライン送付する。膨大な量の購買履歴などのアクティビティを事前にオンライン収集し、このインターネットビッグデータを学習データに用いて訓練した機械学習モデルが中核となる。次に、広告送付先の個人情報を利用することで、当該個人にカスタマイズした広告を機械学習モデルが生成する。

ダイレクトマーケティングは、アルゴリズミックな不正義（Algorithmic Injustice）を内在することから、規制対象となる場合がある。

- 対象者が社会的弱者の場合、心理や行動に影響を与えるダイレクトマーケティングの目的そのものが道徳的な規範に反する
- 広告送付先個人の機微情報に基づく判断が特定集団へのバイアスを含む場合、ダイレクトマーケティングの機能要求そのものが倫理面の公平性に反する
- 収集するアクティビティデータ利用への同意がないと情報プライバシー保護に反する

このようなシステムの影響を受けるステークホルダーは多岐にわたる。開発の可否・利用の可否は、社会から見た AI ガバナンスで議論すべきことである。

一方、品質マネジメントは、ダイレクトマーケティングの機能仕様が与えられたとき、デザインからの品質に関わり、以下のような対応策を実施する。

- 収集したアクティビティデータはパーソナルデータであり、情報プライバシー保護

の観点からデータ匿名加工を施す［プライバシー］
- 公平性に配慮した機構をデザインしバイアスを避ける［公平性］
- セキュリティ機能をデザインし、システムが管理するデータへのアクセスを保護する［セキュリティ］
- 安定して作動するようにシステムをデザインし構築する［信頼性］

AI ガバナンスも品質マネジメントも万能ではない。互いの役割を明確化し、多様なステークホルダーが損害を被ることがないようにする。

参考文献

公的ガイドライン

[AIGBiz] 総務省, 経済産業省. AI 事業者ガイドライン第 1.01 版. 2024 年 11 月 22 日
[AIQMv4] 産業技術総合研究所. 機械学習品質マネジメントガイドライン第 4 版. 2023 年 12 月 12 日
[AIQMv4Annex] 産業技術総合研究所. 機械学習品質マネジメントにおける AI の新潮流への対応. 機械学習品質マネジメントガイドライン第 4 版 Annex. 2023 年 12 月 12 日
[AISIeval] AI セーフティ・インスティテュート. AI セーフティに関する評価観点ガイド第 1.01 版. 2024 年 9 月 25 日
[AISIrt] AI セーフティ・インスティテュート. AI セーフティに関するレッドチーミング手法ガイド第 1.00 版. 2024 年 9 月 25 日
[Copyright 2023b] Copyright Office, Copyright Registration Guidance: Works Containing Material Generated by Artificial Intelligence, Federal Register, vol.88, no.51, pp.16190-16194 (2023).
[DHS 2023] U.S. Department of Homeland Security, Risks and Mitigation Strategies for Adversarial Artificial Intelligence Threats: A DHS S&T Study, December 2023
[EU 2019] European Union, Ethics Guidelines for Trustworthy Artificial Intelligence. April 2019
[NIST 2023] National Institute of Standards and Technology. NIST AI Risk Management Framework.
[NIST 2023b] National Institute of Standards and Technology. NIST AI Risk Management Framework Playbook, January 2023
[OECD 2019] OECD, Scoping the OECD AI principles: Deliberations of the Expert Group on Artificial Intelligence at the OECD (AIGO). OECD Digital Economy Papers, No. 291, OECD Publishing, November 2019

論文、書籍、Web 記事等

[Akiba 2024] Akiba, Takuya, Makoto Shing, Yujin Tang, Qi Sun, and David Ha, Evolutionary Optimization of Model Merging Recipes, arXiv:2403.13187 (2024).
[Anthropic 2024] Tracking Voluntary Commitments https://www.anthropic.com/voluntary-commitments
[Azaria 2023] Azaria, Amos and Tom Mitchell, The Internal State of an LLM Knows When It's Lying, arXiv:2304.13734 (2023).
[Beck 1994] Beck, Kent and Ralph Johnson, Patterns Generate Architecture, in Proc. ECOOP'94, pp.139-149 (1994).
[Bengio 2024] Bengio, Joshua et al., International scientific report on the safety of advanced AI:

interim report. DSIT research paper series number 2024/009. 17 May 2024.
[Brown 2020] Brown, Tom B., et al., Language Models are Few-Shot Learners, arXiv:2005.14165v4 (2020).
[Bsharat 2023] Bsharat, Sondos Mahmoud, Aidar Myrzakhan, and Zhiqiang Shen, Principled Instructions Are All You Need for Questioning LLaMA-1/2, GPT-3.5/4, arXiv:2312.16171 (2023).
[Carlini 2019] Carlini, Nicholas et al., The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks, Proc. 28th USENIX Security Symposium, pp.267-284 (2019).
[Carlini 2021] Carlini, Nicholas et al., Extracting Training Data from Large Language Models, arXiv:2012.07805v2 (2021).
[Carlini 2023a] Carlini, Nicholas et al., Quantifying Memorization across Neural Language Models, arXiv:2202.07646v3 (2023).
[Carlini 2023b] Carlini, Nicholas et al., Extracting Training Data from Diffusion Models, arXiv:2301.13188 (2023).
[Chang 2023] Chang, Tyler A. and Benjamin K. Bergen, Language Model Behavior: A Comprehensive Survey, arXiv:2303.11504 (2023).
[Cottier 2024] Cottier, Ben, Robi Rahman, Loredana Fattorini, Nestor Maslej, and David Owen, The Rising Costs of Training Frontier AI Models, arXiv:2405.21015 (2024).
[Dai 2023] Dai, Damai et al., Why Can GPT Learn In-Context? Language Models Implicitly Perform Gradient Descent as Meta-Optimizers, arXiv:2212.10559v3 (2023).
[DeepSeek-AI 2024] DeepSeek-AI. DeepSeek-V3 Technical Report, arXiv:2412.19437 (2024).
[Devlin 2019] Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, arXiv:1810.04805v2 (2019).
[Dhuliawala 2023] Dhuliawala, Shehzaad, Mojtaba Kmoeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston, Chain-of-Verification Reduces Hallucination in Large Language Models, arXiv:2309.11495v2 (2023).
[Feldman 2021] Feldman, Vitaly, Does Learning Require Memorization? A Short Tale about a Long Tail, arXiv:2004.05271v4 (2021).
[Gao 2024] Gao, Yunfan, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang, Retrieval-Augmented Generation for Large Language Models: A Survey, arXiv: 2312.10997v5 (2024).
[Gallegos 2024] Gallegos, Isabel O., Ryan A. Rossi, Joe Barrow, Md Mehrab Tanjim, Sungchul Kim, Franck Dernoncourt, Tong Yu, Ruiyi Zhang, and Nesreen K. Ahmed: Bias and Fairness in Large Language Models: A Survey, arXiv:2309.00770v3 (2024).
[Gebru 2020] Gebru, Timnit, Jamie Morgenstern, Briana Vecchione, Jennifer W. Vaughan, Hanna Wallach, Hal Daumee III, and Kate Crawford, Datasheet for Datasets, arXiv:1803.09010v7 (2020).

[Google 2024] A Guide to Google Gemini Security https://concentric.ai/google-gemini-security-risks/
[Greshake 2023] Greshake, Kai, Sahar Abdelnabi, Shailesh Mishra, Christoph Endres, Thorsten Holz, and Mario Fritz, Not what you've signed up for: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection, arXiv:2302.12173v2 (2023).
[Kaplan 2020] Kaplan, Jared et al., Scaling Laws for Neural Language Models, arXiv:2001.08361 (2020).
[Liu2024] Yue Liu, Sin Kit Lo, Qinghua Lu, Liming Zhu, Dehai Zhao, Xiwei Xu, Stefan Harrer, and Jon Whittle: Agent Design Pattern Catalogue: a Collection of Architectural Patterns for Foundation Model based Agents, 2405.10467v4 (2024).
[Mallen 2023] Mallen, Alex et al., When Not to Trust Language Models: Investigating Effectiveness of Parametric and Non-Parametric Memories, arXiv:2212.10511v3 (2023).
[Meta 2023] Building Generative AI Features Responsibly (2023)
https://about.fb.com/news/2023/09/building-generative-ai-features-responsibly/
[Nasr 2023] Nasr, Mirad, Nicholas Carlini, et al., Scalable Extraction of Training Data from (Production) Language Models, arXiv:2311.17035 (2023).
[OpenAI 2024] OpenAI, OpenAI o1 System Card (2024).
[OpenAI 2024b] Open AI Security Portal https://trust.openai.com
[OWASP 2023] OWASP Foundation, OWASP Top 10 for LLM Applications, Version 1.0 (2023)
[OWASP 2024] OWASP Foundation, OWASP Top 10 for LLM Applications 2025 (2024).
[Shen 2023] Shen, Yongliang, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang, HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face, arXiv:2303.17580v3 (2023).
[Yao 2023] Yao, Shunyu, et al., ReAct: Synergizing Reasoning and Acting in Language Models, arXiv:2210.03629v3 (2023).
[Santos 2024] Omar Santos and Petar Radanliev, Beyond the Algorithm - AI, Security, Privacy, and Ethics, Addison-Wesley 2024.
[Shuster 2021] Shuster, Kurt et al., Retrieval Augmentation Reduces Hallucination in Conversation, Proc. EMNLP, pp.3784-3803 (2021).
[Sindre 1995] G. Sindre, R. Conradi, and E.-A. Karisson: The REBOOT Approach to Software Reuse, Journal of System and Software, vol.30, no.3, pp.201-212 (1995).
[Snell 2024] Snell, Charlie, Jaehoon Lee, Kelvin Xu, and Aviral Kumar, Scaling LLM Test-Time Compute Optimally can be More Effective than Scaling Model Parameters, arXiv:2408.03314 (2024).
[Song 2023] Song, Chan Hee et al., LLM-Planner: Few-Shot Grounded Planning for Embodied Agents with Large Language Models, arXiv:2212.04088v3 (2023).
[Tan 2018] Tan, Chuanqi, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu, A Survey on Deep Transfer Learning, arXiv:1808.01974 (2018).

[Villalobos 2024] Villalobos, Pablo, Anson Ho, Jaime Sevilla, Tamay Besiroglu, Lennart Heim, and Marius Hobbhahn: Will we run out of data? Limits of LLM scaling based on human-generated data, arXiv:2211.04325v2 (2024).
[Wei 2023] Wei, Jason, Xuezhi Wang, Dale Schuuemans, Maarten Bosma, Brian Ichter, Fei Zia, Ed. H. Chi, Quoc V. Le, and Denny Zhou, Chain-of-Thought Prompting Elicits Reasoning in Large Language Models, arXiv:2201.11903v6 (2023).
[Wei 2022] Wei, Jason et al., Emergent Abilities of Large Language Models, arXiv:2206.07682v2 (2022).
[Xie 2023] Xie, Jian, Kai Zhang, Jiangjie Chen, Renze Lou, and Yu Su, Adaptive Chameleon or Stubborn Sloth: Revealing the Behavior of Large Language Models in Knowledge Conflicts, arXiv:2305.13300v3 (2023).
[芦部 2023] 芦部信善、高橋和之（補訂）：憲法[第 8 版]，岩波書店(2023).
[アレグザンダー 1974] C. アレグザンダー（著），平田翰那（訳）：パタン・ランゲージ，鹿島出版会 (1974).
[岡崎 2022] 岡崎直観、荒瀬由紀、鈴木潤、鶴岡慶雅、宮尾祐介：自然言語処理の基礎，オーム社 (2022).
[ジャクソン 2023] D. ジャクソン（著），中島震（訳）：優れたデザインにとってコンセプトが重要な理由，丸善出版 (2023).
[中谷 2025] 中谷多哉子、中島震（編著）：ソフトウェア工学 [改訂版]，放送大学教育振興会 (2025).
[森岡 2017] 森岡浩，名字でわかるあなたのルーツ，小学館 (2017).

機械学習品質マネジメント検討委員会メンバーリスト

2024 年度　（委員長、副委員長を除き、五十音順、敬称略）

中島　震(委員長)	産業技術総合研究所
妹尾　義樹(副委員長)	産業技術総合研究所
江川　尚志	産業技術総合研究所
越前　功	国立情報学研究所
大岩　寛	産業技術総合研究所
岡本　球夫	パナソニックホールディングス
小川　秀人	日立製作所
桑島　洋	デンソー
小林　健一	富士通
小宮山　英明	コニカミノルタ
新原　敦介	日立製作所
鈴木　知道	東京理科大学
高田　眞吾	慶應義塾大学
中島　裕生	NPO MedXML コンソーシアム
難波　孝彰	パナソニックホールディングス
浜谷　千波	アドソル日進
福島　真太朗	トヨタ自動車
宗像　一樹	富士通
安井　裕司	本田技術研究所
山田　敦	日本 IBM
若松　直哉	日本電気

ガイドライン詳細検討タスクフォース メンバーリスト

2024 年度　（五十音順、敬称略）

磯部　祥尚	産業技術総合研究所
江川　尚志	産業技術総合研究所
大岩　寛	産業技術総合研究所
岡本　球夫	パナソニックホールディングス
小川　秀人	日立製作所
川本　裕輔	産業技術総合研究所
北村　弘	IPA/AI セーフティ・インスティテュート
桑島　洋	デンソー
小西　弘一	産業技術総合研究所
小林　健一	富士通
小宮山　英明	コニカミノルタ
新原　敦介	日立製作所
妹尾　義樹	産業技術総合研究所
田部　尚志	日本電気
中島　震	産業技術総合研究所
中島　裕生	NPO MedXML コンソーシアム
難波　孝彰	パナソニックホールディングス
浜谷　千波	アドソル日進
林谷　昌洋	日本電気
三宅　和公	住友電気工業
三宅　武司	サイバー創研
若松　直哉	日本電気

本プロジェクトに関係して産業技術総合研究所に特定集中研究専門員として部分的に在籍しているメンバーについては、それぞれの出身母体で記載した。