Meta AI

Llama 2

責任ある使用のためのガイド

大規模言語モデル（LLM）を使用した
プロダクトの責任ある開発のための
リソースとベストプラクティス

目次

オープンイノベーション 1
このガイドの使い方 3
LLMを用いたプロダクトのリスク軽減 4
責任あるAIへの配慮 4
LLMを用いたプロダクトのリスク軽減 5
基盤モデルの開発 6
責任あるLLM製品の開発段階 7
ユースケースの決定 7
プロダクトのファインチューニング 8
責任ある微調整の流れ 9
ステップ1：コンテンツポリシーとリスク軽減策の定義 9
ステップ2：データの準備 10
ステップ3：モデルの学習 10
人間のフィードバックによる強化学習（RLHF） 11
AIのフィードバックによる強化学習（RLAIF） 11
ステップ4：パフォーマンスの評価と改善 12
レッドチームのベストプラクティス 13
プライバシー敵対的攻撃 14
入出力レベルリスクへの対応 14
入力レベルでのリスク軽減 15
出力レベルでのリスク軽減 16
効果の評価 17
ユーザーとのインタラクションにおける透明性と報告メカニズムの構築 18
フィードバックと報告の仕組み 18
透明性と制御のベストプラクティス 18
開発者向けリソース 20
責任ある生成AIの構成要素を組み合わせる 22
付録：Code Llamaの紹介 23
基盤モデルのユースケース 24
指示モデルのユースケース 25

Metaではオープンサイエンスに注力している。活気あるAIイノベーションのエコシステムが科学的発見を推し進め、教育から農業、ひいては気候管理からサイバーセキュリティに至るまで、幅広い分野に革命をもたらすと信じているからである。

私たちは、グローバルな課題に対処するためにAIの力を活用できると信じていますが、その可能性を責任を持って解き放つためには、アクセスの民主化とリスク管理に関する協力体制が重要になる。私たちは、世界のあらゆる開発者をエンパワーし、ブレークスルーを推進し、新しいプロダクトやソリューションを生み出し、技術進歩や経済成長の加速による恩恵を受けたいと考えている。

オープンサイエンスが達成してきたこと

すべての人の利益のためにAIの最先端技術を進歩させるというMetaの使命は、オープンな研究とコミュニティーの協力なしには達成できない。Metaは、機械翻訳、コンピュータビジョン、フェアネスエバリュエーションのコードとデータセットをオープンソース化しており、同時にPyTorch、ONNX、Glow、Detectronといったツールで AI開発者コミュニティのインフラに貢献している。また、最先端の大規模言語モデル（LLM）であるLlama 1とOPT-175Bも開発し、研究リリースを通じて科学コミュニティに提供している。これらのツールは、有数のAI研究機関や企業における探索的研究や大規模なデプロイメントを可能にしているが、エコシステム全体にわたる何千もの人々の貢献なしには不可能だったと考えられる。Metaによるオープンリリースは、モデル効率、医学、評価方法に関する会話の安全性研究、脱バイアス技術、LLMにおけるハルシネーションの原因といった研究を後押ししている。

この技術をより広く商業利用できるようにすることで、LLMのプロダクトへの統合をさらに加速させることができ、ひいては経済的・競争的利益をもたらすだろう。これらの目標を念頭に置いて、私たちは以前研究目的として立ち上げた基盤LLM「Llama」の新バージョンをリリースしている。LLMの事前学習の計算コストは、小規模な組織にとってはいまだ法外に高く、大規模モデルが普及すれば、この分野の二酸化炭素排出量はますます増えることになる。これらのモデルをオープンに公開することで、コストが統合され、参入障壁をなくし、中小企業はLLMのイノベーションを活用してテキスト生成のユースケースを模索、構築することができるようになる。最終的には、世界中のあらゆる規模の組織がAIの進歩によって約束された経済成長の恩恵を受けられるよう、より公平な競争条件が整うことになると信じている。

アクセスの民主化によって、これらのモデルはより多くの人々が利用できるようになるので、この技術は世界全体に利益をもたらすための正しい道だと私たちは信じている。さらに、このアプローチはAIコミュニティ全体の取り組みを促進し、LLMをさらに改善すると考えている。私たちの経験に基づいて言えば、オープンなアプローチは、AI開発者コミュニティの集合的な知恵、多様性、創意工夫を活用して、この技術の利点を実現するものだと考えられる。ゆえにコラボレーションの力によって、これらのモデルはより良く、より安全なものになるだろう。そうでなければ、今後の開発や安全への取り組みは、一部の大企業だけが行う試みに限られてしまう。Llamaの商用ライセンスは、特定の違法行為または有害な使用行為を禁じているものの、基盤モデルを活用する開発者が、新しいプロダクトが責任を持って構築、配備されるよう、重要な役割を果たすと信じている。

このガイドの目的は、「基盤モデルを活用したLLMを用いた機能」の責任ある開発のためのリソースとベストプラクティスを提供することで、開発者コミュニティをサポートすることである。私たちは、責任あるAIを構築し、プライバシーやコンテンツに関連する潜在的なリスク、そして社会的な影響を認識することに真剣に取り組んでいる。これらのシステムを使用する開発者も、これらのリスクを真剣に受け止め、適切なリスク管理プロセスを導入すべきである。このテクノロジーが私たちの仕事と創造活動の中心となっていくにつれて、私たち全員がパフォーマンスと安全性を向上させる役割を果たすことになるだろう。

ガイドの使い方

本ガイドは、各レベルのLLMを利用した責任ある製品を作るための一般的なアプローチを概説した開発者向けのリソースである。本ガイドでは、開発者が特定のユースケースや市場との関連において評価すべきベストプラクティスや考慮すべき事項を取り上げている。また、開発者がシステム内のさまざまなポイントでリスクに対処するために利用できるリスク軽減策やリソースについても紹介している。あるレベルで採用された戦略は、システム全体に影響を与える可能性があるため、これらのベストプラクティスは総合的に見たうえで検討されることをお勧めする。

このガイドに含まれる推奨事項は、責任ある生成AIに関する現在の研究を反映したものである。この分野が進歩し、基盤モデルへのアクセスが増えるにつれて、AIの安全性に関するイノベーションがさらに進むと予想している。ベストプラクティスを実施するかどうかの決定は、各社の製品がデプロイされる地域の管轄権に基づいて評価されるべきであり、各社の社内の法的およびリスク管理プロセスに従う必要がある。

責任あるAIとシステム設計の概要

責任あるAIへの配慮

生成AI技術が不当な実害をもたらさないようにすることは、何よりも重要なことである。生成AIは急速に発展しており、研究、オープンなコラボレーション、そしてこの技術を世界中の人々の手に届けたいと考えるプロダクトのリリースによってさらに推進されつつある。この規模での成長は、責任あるAIのデプロイメントにおいて新たな課題をもたらす。しかし責任に関する原則の多くは、既存のAI技術と変わらない。このような配慮はMetaの責任あるAIへのアプローチの核となるもので、公平性と包括性、堅牢性と安全性、プライバシーとセキュリティ、そして透明性と管理、ガバナンスと説明責任のメカニズムを含むものである。LLMは数あるAIツールのひとつであり、そのリスクは「どのように使用されるか」という視点から評価されるべきである。

基盤モデルと生成AIシステムは、先行技術と比べて、より大きな影響力と、高い性能を誇る。これらのモデルの性能、実用性、柔軟性の向上は、それらが既存のユースケースにもたらす価値がシステム導入の運用コストを上回る可能性があるため、ユビキタス化につながる可能性が高い。完全に新しいコンテンツを生成する能力は、新しいユースケースを生み出すことにもなるので、それらがもたらす可能性のあるリスクを評価する必要がある。本技術の悪用については潜在的なリスクとして、すでにオンライン上で表面化しているものもあり、例えば違法なコンテンツの作成または拡散、好ましくないまたは憎悪的なコンテンツ、または不適切なアドバイスの提供につながる可能性のあるコンテンツなどがある。このような事例は、生成AIツールがより身近になるにつれて増える恐れがある。

私たちのプラットフォーム上の生成AIにおける文脈固有のリスクに対処するための安全対策

これらの予防策は、基盤モデルにおいて評価・軽減されうるものだけでなく、さまざまな介入ポイントにわたって重層化されている。当社のLlama 2に関する研究論文で述べられているように、開発プロセスの初期段階で適用されたいくつかのリスク軽減策は、モデルの下流での性能と安全性に悪影響を及ぼす可能性があるため、一部のリスクに関しては、プロダクト開発サイクルの後半で対処した方が良い場合もある。オープンソースモデルを活用した生成AI機能の開発者たちも同様に、プロダクト開発サイクル全体にわたって責任あるAIの全体的な視野に立ち、プロダクトが安全でエンドユーザーに利益をもたらすことを保証する必要がある。

LLM搭載プロダクトの
リスク軽減策

基盤モデルとは汎用的なAI技術のことであり、一方、LLMを搭載したプロダクトは、明確なユースケースを持ち、ユーザーインターフェイス（製品に組み込まれていることもある）を通じて意図された用途や機能を実現する為に、特定のタスクを実行するものである。LLMを搭載したシステムは、基盤モデルと、複数の製品固有のレイヤー両方を包含する。プロダクト開発ライフサイクルの様々な段階で、開発者は機能の目的と機能を決定しなければならないが、これには潜在的なリスクをともなう。一方でこれらの決定は潜在的なリスクを軽減する機会にもなり得る。ゆえに開発者は、製品の各レイヤーを検証し、製品の目的と設計に基づいてどのような潜在的リスクが発生しうるかを判断すること、そして、それに応じて緩和戦略を実施することが重要である。

次のセクションでは、LLM製品開発のさまざまな段階における、責任あるAIの考察を示したい。それぞれのレベルにおいて、潜在的なリスクを軽減するためのベストプラクティスを紹介する。

基盤モデルの開発

Llama 2は、Llama 1モデルの新バージョンであり、以前は研究用に公開されていたものである。このモデルの新しい事前学習済みバージョンとファインチューニング済みバージョンは、商用リリース用に更新されている。また、当社モデルの潜在的な能力と限界を理解するために、様々な事前学習のデータレベルの調査を行う以外にも、教師ありファインチューニング、人間のフィードバックによる強化学習(RLHF)、および反復的なレッドチーミングによって、モデルの微調整バージョンに安全性緩和を適用した(これらのステップについては、「プロダクトのファインチューニング」のセクションを参照)。

事前学習データ、モデル・アーキテクチャとパラメータ、および事前学習の評価に関する情報は、Llama 2に関する研究論文を参照のこと。この論文では、詳細な安全性調整作業や評価結果などを含む、ファインチューニングされたバージョンの開発手順についても詳しく説明されている。追加情報は、リリースに付属のモデルカードに記載されている。研究論文とモデルカードは、モデルの能力と限界に関する情報を説明しており、開発者が新しいユースケースに向けてLlamaをより安全にチューニング、評価、導入するのに役立つ。

事前学習中、モデルは学習データに含まれる人間の言語サンプル全体において統計的パターンを理解することができる。Llamaのトレーニングデータセットは、一般に公開されている様々なオンラインデータから収集されているが、このトレーニング・コーパスはほとんどが英語で記されており、現在のモデルの使用目的に合致している。トレーニングに使用した各データセットは、Metaの標準的なプライバシー・レビュープロセスに従っている。また事前学習のデータについては、個人情報が大量に含まれている特定のソースからのデータを削除するように努めた。事前学習の後、モデルは単純な文法規則から文脈、感情、比喩的な表現などの複雑なニュアンスまで再現できる。しかし、このモデルは人間のように知識を得たり、世界に関する信念を新たに生成するものではなく、学習データのパターンに基づいて、文中の次の単語を予測する学習のみを行う。

事前学習されたモデルを使う場合には、次のセクションで説明するテクニックを使ってチューニングし、あなたが意図したユースケースやタスクに反する、安全ではない出力を生成するリスクを減らすことを推奨する。顧客とあなたのLLMとのインタラクションに関して適用される利用規約やその他の関連ポリシーがある場合は、それらのポリシーに合致するようにモデルをファインチューニングすることも推奨する。また、LLMに特化した新たな利用規約やポリシーを設定したり、提供されたデータやフィードバックがファインチューニングにどのように使用されるかについては、ユーザーに通知する必要性も考慮する必要がある。

責任あるLLM製品の開発段階

開発者は、リリースされたモデルの特定のプロダクトのユースケースを確認し、そのユースケースに関連するリスクを評価し、安全性を確保するためのベストプラクティスを適用する責任を負う。このセクションでは、製品開発と展開の各段階で利用可能な考慮事項とリスク軽減策について概説したい。

これらの段階には、高いレベルで以下のようなものが含まれる：

1. ユースケースの決定
2. プロダクトに対するファインチューニング
3. 入出力レベルのリスクへの対応
4. ユーザーとのインタラクションにおける透明性と報告メカニズムを構築

ユースケースの決定

開発プロセスにおける重要な決定とは、「どのユースケースに焦点を当てるか」にある。このガイドを使用する開発者のほとんどは、カスタマーサポート、AIアシスタント、社内生産性ツール、エンターテインメント性の高いエンドユーザー体験、研究用アプリケーションなどのユースケースをすでに念頭に置いているはずだ。もしあなたが開発者であっても、そのモデルを使いたいと考えるユースケースが決まっていないのであれば、さまざまな倫理原則や価値観を考慮しつつ、人々や社会の生活を向上させるユースケースに焦点を当てることを検討することをお勧めする。内部リスク評価プロセスの構築又は採用は、特定のユースケースの潜在的なリスクを特定するのに役立つので、製品のエンドユーザーなどがどのような影響を受けるかにフォーカスして行うべきである。これはプロダクトリリースにおける状況に応じた安全性を評価するためには不可欠であり、潜在的なユーザーに対する調査やインタビュー、あるいは類似のプロダクトアプリケーションの市場分析といった形をとることができる。

AIの開発・ローンチにおける価値観を初めて検討する場合は、学術機関や専門機関が発表しているリスク管理に関する、例えば以下の様な原則やガイダンスを参照のこと。

- OECDのAI原則
- NISTの信頼できる責任あるAIリソースセンター

2

プロダクトのファインチューニング

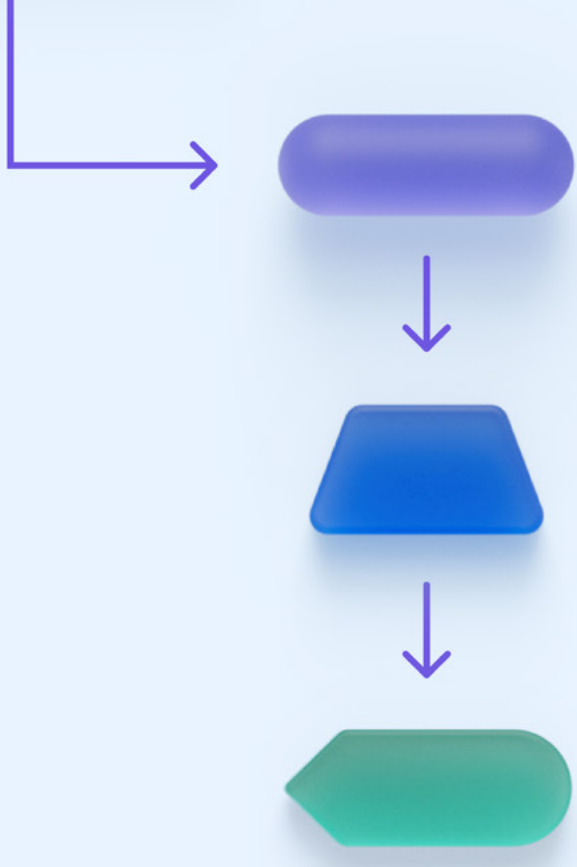
プロダクトに特化したファインチューニングのおかげで、開発者は、事前学習されたモデル、あるいはある程度ファインチューニング済みのモデルを使い、限られたデータとリソースしか必要としない特定のタスクを実行することができる。Metaによって実行された最初のファインチューニングでも、ドメイン固有のデータセットでモデルをさらに学習させることで、定義されたユースケースでの性能を向上させることができる。

ファインチューニングによって、モデルをドメインまたはアプリケーション固有の要件に適合させ、安全性を向上する

事前学習されたLLMのファインチューニングの例としては、以下のようなものがある：

- **テキストの要約：**事前学習された言語モデルを使用することで、長い文章とそれに対応する要約のペアを含むデータセット上で、モデルをファインチューニングすることができる。ファインチューニングされたことで、新しい文書に対して簡潔な要約を生成することができる。
- **質問応答：**SQuAD（Stanford Question Answering Dataset）などのQ&Aデータセットで言語モデルをファインチューニングすることで、モデルは与えられた文脈の段落に基づいて質問に答える方法を学習する。このようにファインチューニングされたモデルは、さまざまなトピックに関する質問に答えることができるようになる。
- **センチメント分析：**モデルは、ラベル付けされたテキストレビュー（肯定的または否定的な感情）のデータセット上でファインチューニングされ、感情を認識し、ユーザー満足度を知るための分析を実行することができる。このタスク特化型のデータセットでモデルをトレーニングすることで、テキストのセンチメントを正確に予測することができる。

これらの例は、LLMをファインチューニングすることで、特定のユースケースに対してモデルの能力を特化させ、パフォーマンスを向上させることで、特定のアプリケーションにより適したモデルにすることができることを示している。基盤モデルとタスク固有のデータセットの選択は、望ましい結果を得るために重要な役割を果たす。



責任あるファインチューニングの流れ

以下は、LLMを責任を持ってファインチューニングするために必要な一般的ステップであり、Metaの「責任あるAI」のフレームワークによって高レベルで実践されている：

1. コンテンツポリシーと緩和策の定義
2. データの準備
3. モデルの学習
4. パフォーマンスの評価と改善

ステップ1：コンテンツポリシーと緩和策の定義

コンテンツポリシーは、各社の製品の使用目的およびユーザーに基づいて、どのようなコンテンツが許容されるかを定義し、違法、暴力的、または有害なコンテンツの制作に関する安全上の制限を定めることもできる。特定の分野や地域では法律や基準が異なる場合があるため、これらの制限はプロダクトカテゴリーに照らして評価されるべきである。さらに、年齢に応じたプロダクトエクスペリエンスの開発など、コンテンツ・ポリシーを設計する際には、特定のユーザーコミュニティのニーズを考慮する必要がある。このような方針を策定することで、必要なデータ、アノテーションの要件、安全なファインチューニングの目標が決定される（実施されるリスク軽減策の種類を含む）。これらのポリシーは、RLHFを行う際のデータのラベリングや、例えばユーザー入力やモデル出力の実施判断などを行うプロダクトレイヤーを追加する際に用いられる。

ステップ2：データの準備

LLMの下流アプリケーションの開発は、LLMの潜在的な制限、プライバシーへの影響、特定のユースケースに対するデータの代表性を考慮することから着手すべきである。つまり、対象領域を代表するクリーンなデータセットを準備し、前処理することから始めるということになる。このプロセスには、テキストをトークン化し、特殊文字を処理し、不要な情報を削除すること、データセットを「トレーニングセット」、「検証セット」、「テストセット」に分割することが含まれる。このステップは、データがデプロイ状況におけるエンドユーザーを反映するものであることを確認する作業も含まれるだろう。例えば、あるプロダクトを非英語圏市場で展開する予定であれば、関連する言語の例を十分に収集することによってそれを達成できる。データの代表性はユースケースに依存するので、それに応じて評価されるべきである。

特定のユースケースのためにファインチューニングする場合、性別、人種、言語、文化、その他のバイアスについてトレーニングデータを調べることが有益である。このようなパターンを理解することは重要である一方、問題のあるコンテンツを学習データからすべてフィルタリングすることは、その後のパフォーマンスや、プロンプトエンジニアリングなどの安全対策に予期しない結果をもたらす可能性があるため、必ずしも最適とは限らない。データを削除するのではなく、データの代表性に注目することで、ファインチューニング済みモデルがその出力生成において偏ることを防ぐことができる（何が「代表的」とみなされるかについては、プロダクトがデプロイされる状況によって異なる）。また、開発者はデータに対する人間のフィードバックやアノテーションが、主観的な意見という意味で、ファインチューニングされたモデルをさらに偏向させる可能性があることに注意を払うべきである。そのためにも、アノテーションガイドラインにバイアスがかかることを防ぎ、アノテーター（ラベル付けを行う人）のバイアスの影響を緩和するための措置を講じることを推奨する。このトピックに関するリソースは以下の通り：

- アノテーターを責めないで：バイアスはすでにアノテーションの指示から始まっている
- アノテーターのあるべき態度：アノテーターの信念とアイデンティティが有害な言語の検出にどのようなバイアスを与えるか

過学習、プライバシー、セキュリティなど、考慮すべきリスクは他にもいくつかある。このようなリスクを軽減するためにも、ユースケースを代表する質の高いデータセットを収集し、厳密な評価を実施し、ファインチューニングしたモデルの使用可能性をレッドチーミングによってテストする（ステップ4「パフォーマンスの評価と改善」を参照）など、ファインチューニングのプロセスを慎重に設計すべきである。

ステップ3：モデルの学習

ファインチューニングでは、限られたイテレーション数でモデルをトレーニングする。事前学習されたモデルがファインチューニングのために環境に読み込まれると、学習プロセスが、エポック数、バッチサイズ、学習率などのハイパーパラメータを設定する。データがモデルに渡され、損失が計算さ

れ、バックプロパゲーションによって重みが更新される。トレーニングの進捗は検証セットを使ってモニタリングされ、必要に応じてハイパーパラメータが調整される。

安全性追求のためにLLMをファインチューニングするには、多くのテクニックが必要であるが、その多くはLlama 2の研究論文で詳しく説明されている。これらのテクニックには以下のようなものがある：

- **教師ありファインチューニング（SFT）：**有用性と安全性に関する注釈が付けられたデータを用いた、教師ありのファインチューニング。
- **人間のフィードバックによる強化学習（RLHF）またはAIのフィードバックによる強化学習（RLAIF）：**RLHFテクニックをサポートする為のトレーニングの安全性と有益性の報酬モデルは、繰り返しモデルを改善し、ジェイルブレイク（あらかじめ設けられた制限の非正規な解除）に対してよりロバストになる。
- **目標となる安全性のための文脈蒸留：**安全性のための文脈蒸留（context distillation）を用いると、敵対的なプロンプトに「あなたは安全で責任あるアシスタントです」といった安全な事前プロンプトを付け、その後に新しい出力でファインチューニングを行うことによって、モデルは敵対的なプロンプトと安全な応答を関連付けることができる。

人間のフィードバックによる強化学習（RLHF）

LLMの出力をユーザーの期待や価値観に一致させる場合、開発者が考慮すべきアプローチの1つは、人間のフィードバックによる強化学習（RLHF）メカニズムを実装することである。これには、トレーニングされたアノテーターまたはユーザーから順位付けされたデータを収集し（モデルの入力と生成された複数の出力が与えられた場合、ポリシーに従ってそれらを「最善」から「最悪」に順位付けすること）、人間のフィードバックの代わりとして機能する報酬または有益性モデルを学習し、強化学習によって報酬／有益性モデルのスコアを最大化するようにLLMを最適化するプロセスが含まれる。

AIフィードバックによる強化学習（RLAIF）

報酬モデルは、AIフィードバックによる強化学習（RLAIF）を使用することで、特定のポリシーに合わせて改善することもできる。ファインチューニングされたLLM自体は、報酬モデルの学習のための順位付けによる合成データを作成するために使用することができる。モデル入力、応答ペア、関連するガイドラインが与えられると、LLMはどの対応がガイドラインに最も沿っているかを予測し、合成報酬モデルのデータが、報酬モデルのトレーニングデータを補強するために使用される。

ステップ4：パフォーマンスの評価と改善

最終段階では、ユースケースに従って、特定のタスクや安全性のベンチマークに対する性能を測定するために、ファインチューニングされたモデルをテストセット上で評価する。これには、評価結果に基づいてモデルの長所と短所を分析し、性能と安全性をさらに高めるためにさらにデータを収集し、ホールドアウトテストのデータセットを使ってモデルの性能に満足するまで反復するプロセスが含まれる。

モデルのリスクの測定に有用な多くの種類の補足的な評価が存在し、自動ベンチマーク、人間の評価者による手動アノテーション、LLM自体を評価者とする評価などが含まれる。言語モデルの全体論的評価（HELM）では、一般的に使用されている自動ベンチマークについて議論している。

パフォーマンスを向上させるための評価戦略とプロセスには、以下のようなものがある：

- **自動評価**では、自動ベンチマークと分類器を活用し、特定のリスクカテゴリーに関する出力を判断する。
- **手動評価**では、人間のアノテーターまたはモデルのアウトプットを判断する専門家を活用している。
- **レッドチーミング**とは、望ましくない動作や出力を引き出す可能性のあるプロンプトを作成することで、モデルの脆弱性や顕在化したリスクを特定する体系的な取り組みである。このようなモデルの操作は、セーフガードや「ジェイルブレイク」の試みをテストするために行うことができる。

レッドチームのベストプラクティス

レッドチームでは、可能な限り実世界の行動や脅威のベクトルを推定しながら、テストと測定に体系的なアプローチを採用すべきである。

- **多様性：**レッドチームには、潜在的なユーザーとその層を幅広く代表するようなさまざまな専門的背景を持つ多様な人々を含めることが望ましい。レッドチームのメンバーは社内の従業員、専門家、コミュニティのメンバーで構成することもできる。
- **専門分野の知識：**専門分野のエキスパートは、特定されたリスクカテゴリーに対する理解度に基づいてモデル回答を判断、各カテゴリーに該当する回答をラベリングすべきである。
- **定期的なテスト：**このモデルでは、攻撃に対するリスク軽減策が有効かどうかを判断するために、定期的なテストを行うべきである。これには、人間によるラベル付け（高いコストがかかる）、またはリスクカテゴリーに該当する回答を認識するように学習された識別器を用いるなど、何らかの形の自動評価が必要である。

プライバシー敵対的攻撃

プロダクトをリリースする際には、プライバシー保護の追加を検討し、悪意のある者が情報を不正に引き出せるかどうかをテストするべきである。プライバシー敵対的攻撃とは、攻撃者がモデルからデータを流出させることである。例えば、敵対的攻撃には、特定のサンプルがモデルのトレーニングデータに含まれているかを予測するメンバーシップ推論攻撃や、サンプルのサブセットの代表的なビューを再構築するモデル反転攻撃などが含まれる。プロンプト・インジェクション攻撃は、コンテンツの制限を回避して特定の出力を生成しようとする攻撃を指す。

各企業が実施するレッドチームのプライバシー敵対攻撃は、そのような攻撃の実現可能性を示すことができる。企業が（適用される個人情報保護法に従って）データを使用してモデルをファインチューニングするシナリオでは、モデルが特定のデータを記憶したかを確認するために出力をテストすることをお勧めする。

このアプローチは、AIアシスタントやエージェントとしてデプロイされることを想定したモデルのテストに特に有効である

入出力レベルのリスクへの対応

入出力のレベルで適切な保護措置が取られない場合、モデルが敵対的な入力に適切に対応し、コンテンツポリシーや保護措置を回避する攻撃（「ジェイルブレーキング」）から身を守るのは非常に難しくなる。出力レベルでのリスク軽減措置は、リスクの高いコンテンツやポリシー違反のコンテンツの生成に対するセーフガードとしても機能する。コンテンツポリシーの実施は、自動化されたシステムや、サンプルやレポートの手動分析によって管理することができる。自動化されたシステムにはプロンプト入力やシステム出力をフィルタリングする為の、機械学習やルールベースの識別器が含まれる。ユーザーがそれらのポリシーに繰り返し違反した場合に備えて、使用法または結果のポリシーを定義できる。

入力レベルでのリスク軽減

「入力」とは、ユーザーからシステムに渡される情報を指す。たとえ開発者であっても、ユーザーの入力をコントロールすることはできない。入力フィルタとセーフガードの実装が無い場合、たとえ高度なモデルであっても、有害な出力や誤解を招く出力を生成したり、コンテンツポリシーに違反するように操作されてしまう恐れがある。プライバシーを保護し、潜在的な危害を防ぐためのセーフガードは、モデルのチューニングにより開発が可能であるが、たとえ厳格な設計とテストを行ったとしても、そのセーフガードが完璧ではなく、侵害される可能性があることを想定すべきである。追加的なセーフガードとして、入力の直接フィルタリングとエンジニアリングなどが挙げられる。これらを効果的に利用するには、モデルの入力を適切にフォーマット化しなければならない。これらのアプローチには以下のようなものがある：

- **プロンプトフィルター：**入力がコンテンツポリシーに違反していなくても、モデルが問題のあるエンゲージメントや出力を生成する場合がある。このような場合、モデルが意図した応答ができるようになるまで、いくつかの入力に対してフィルタリング、ブロック、ハードコーディングされた応答を行うことが適切である。この戦術は、システムにおけるユーザーの経験や主体性にトレードオフをもたらすことがある。したがって、よりロバストな解決策が開発されるまでの間は、こうした制限や修正による安全上のメリットとコストを比較検討する必要がある。
- **プロンプトエンジニアリング：**ユーザーの入力の直接修正は、出力の生成中に背景知識とガイドラインを確立するためのプロンプトにコンテキスト情報または制約を含めることにより、モデルの動作を制御し、責任ある出力を促すための選択肢です。修正については、自動識別や分類、LLMそのものへの手助け、ルールエンジンなど、さまざまな方法で行うことができる。モデルの多様性と表現力を高めることで、ユーザー体験を向上させるのに役立つ。例えば、プロンプトエンジニアリングは、より広範な参考文献を含め、特定の口調や視点を適用するようモデルを指示するために活用することができる。プロンプトエンジニアリングのルールは、ハードコーディングされている場合もあれば、確率に基づいている場合もある。

プロンプトと並行して、理想的な責任ある行動を示す手本となる入力と出力のサンプルの提供も有効

出力レベルでのリスク軽減

下流のユースケースに基づいて、いくつかのアプローチを適用して、問題のあるコンテンツやポリシー違反のコンテンツのモデルが生成した出力を検出、フィルタリングすることができる。ここでは、出力のフィルタリングに関するいくつかの考慮すべき事項とベストプラクティスを紹介する。出力フィルタのリスク軽減には、プロダクトを購入できる地域で使用されているすべての言語を含める必要がある。

- **ブロックリスト：**リスクの高いコンテンツの生成を防ぐ最も簡単な方法のひとつは、どのような状況においても回答に含めてはいけないフレーズのリストを作成することである。問題を特定できる単語はたくさんある。例えば、中傷で使われる言葉は文脈に関わらず常に不快である。ブロックリストはシンプルで魅力的だが、ユーザーのモデルの使用を不当に制限するものではない。言葉には文脈に依存した特定の意味があることも多い。例えば、性的な暗示を与えるような用語が、医学的な文脈でも使われることもある。コンテンツポリシーは、許可されるトピックと禁止されるトピックの詳細をユーザーに明確に示すのに役立つ。
- **分類器：**より効果的ではあるが、より難しいアプローチのひとつは、選択された単語が伝える意味に基づく出力の検出を行い、フィルタリングする分類器を開発することである。分類器は、既知の感情や意味内容の例に基づいて適切に学習されると、その感情や意味を表現する新しい例を識別するのに非常に効果的となる。

効果の評価

プロンプトフィルタリングとエンジニアリングは重要で安全なリスク軽減策であるが、その有効性をモニタリングし、意図しない結果を避けることも重要

ベストプラクティスには以下のようなものがある：

- **意図しない結果にならないようテストする。**プロンプトエンジニアリングが意図せず他の問題を引き起こさないように注意するべきである。望ましい動作を保証するためにも、プロンプトエンジニアリングの後にエンドツーエンドのパフォーマンスをテストする必要がある。
- **セーフガードの効果を評価する。**一般に公開されているデータセットの多くは、データセットが入力して使用された場合の特定の懸念事項に対するベンチマークしてのプロンプトを提供する。モデルの応答を集めた後、標準化された測定基準によってそれらの応答を評価することができる。
- **異なる言語用に調整する。**プロンプトフィルタリングとエンジニアリングによるリスク軽減には、製品が購入できる地域で使用されているすべての言語を含める必要がある。これらのリスク軽減策の効果は、言語やコミュニティレベルのニュアンスに左右される可能性があるからである。Llamaは、その意図した使用目的により、主に英語のデータで訓練されている。したがって、他の言語でのリスク軽減策を慎重に評価することも重要である。

4

ユーザーとのインタラクションにおける透明性と報告メカニズムの構築

LLMを搭載した機能をリリースし、ユーザーと対話することで、新たなユースケースや新たな懸念事項が明らかになることがある。ユーザーとの対話は、強化学習（前セクションで説明）に使用できる重要なフィードバックをもたらす。これは、ユーザーに対して適切な告知、透明性、制御性を与える機会でもあり、それによって機能に対する満足度や信頼が高まる可能性がある。

フィードバックと報告の仕組み

適切なフィードバックや報告メカニズムによってユーザーとの対話を促進することは、質の高い出力を保証するために必要不可欠である。フィードバックの仕組みは、単純に肯定的か否定的か（例：親指を立てる、下げる）にすることもできるし、企業のユースケース（例：AIアシスタント）に基づいて予測可能な問題のタイプに合わせてフィードバックを調整することで、フィードバックの質を高めることもできる。このフィードバックを利用すれば、開発者はよりターゲットを絞った方法でモデルを改善することができる。自由形式のフィードバックを行えるような選択肢を報告のメカニズムに設けることで、ユーザーから寄せられた新たな懸念や予期せぬ懸念を明らかにすることもできる。さらにユーザーは、モデルだけでは認識できないようなエラーや安全でない行動、最適でない行動を特定し、強調することができる。一方開発者はこのフィードバックを用いてモデルをさらに訓練することで、パフォーマンスを向上させ、ミスを防ぐことができる。プロダクト開発者は、ユーザがモデル出力を報告する割合をモニタリングし、それらの報告およびモデル出力の選択されたサンプルを直接手動で確認することにより、フィードバックをレビューすべきである。

透明性と制御のベストプラクティス

質の高いフィードバックを保証し、エンドユーザーにAIとのインタラクションについての通知と選択肢を提供するために、開発者はユーザーとの対話について以下の事項を考慮する必要がある：

- **透明性：**開発者は、ユーザーとの対話に先立ち、あるいはユーザーとの対話中に、システムの潜在的なリスクや限界についてエンドユーザーに透明性を開示する方法も検討すべきである。例えば、AIを搭載したチャットボットとやりとりしていることをユーザーに開示することは、特定の市場において必須要件になりつつあるが、虚偽の情報や誤った情報に関連する懸念に対処するためのベストプラクティスにもなりうる。開発者は、AIエージェントが人間であると伝えたり暗示するべきではない。特に擬人化されたインタフェースを構築、デプロイする際にはなおさらである。いつ、どのように透明性を確保すべきかを見極めるには、偽情報の文脈、意図、敏感さ、可能性が重要な要素となる。ユーザーの回答がモデルのファインチューニングに使用される可能性があることを知らせるべきか否かも含め、専門アドバイザーと協力してユーザーに対して提供す

べきタイプの透明性を決定することを推奨する。開発者は、システムカードを使用し、AIシステムの基本的なアーキテクチャを理解し、特定のAIによる体験がどのように生み出されるかを説明することも検討すべきである。その他のベストプラクティスについては、AIによる合成メディアのための責任ある実践に関するパートナーシップを参照のこと。

- **制御メカニズム：**追加の制御には、ユーザーにLLMが生成する出力をカスタマイズする選択肢を与えることも含まれる。例えば、ユーザーは複数の選択肢から出力を選択することも、拒否することもできる。また、編集機能により、出力に対するユーザーの主体感（SoA）を高めることもできる。一方で、開発者はユーザーを成功に導くための教育フローを検討すべきである（例：出力を改善する為のプロンプトの提案や説明を行う）。

開発者向けリソース

現在、LLMの責任ある訓練と利用をサポートするバリューチェーンが生まれつつあり、ベストプラクティス、ツール、ベンチマークをよりオープンに公開・共有することで、さらに進歩するものと信じている。このエコシステムに貢献する研究者、プラットフォーム、企業、開発者コミュニティの数は増え続けている。私たちは、LLMの責任ある開発の為の多くのツールが今後利用可能になっていくことを期待し、開発者を支援するために、安全性に関する研究とツールのオープンな意見交換を促進している。

ベストな結果を得るためには、常に最新版のモデルを把握し、活用することが重要

私たちのパートナーシップにより、LlamaをAzure Model Catalogで利用できるため、Microsoft Azureを使用している開発者は、コンテンツフィルタリングと安全機能向けのクラウドネイティブのツールを活用できる。以下は開発者向けの有名なハブや実装リソースであるが、このリストがすべてを網羅しているわけではない。マイクロソフトはまた、責任あるAIリソースのリポジトリも提供している。

フィルターと識別器：

- Azureのコンテンツフィルタリングシステム（様々な言語に対応）：
  https://learn.microsoft.com/en-us/azure/cognitive-services/content-safety/overview
- 問題のある単語生成に対するフィルターリスト：
  https://github.com/LDNOOBW/naughty-words-js
- オープンドメインのチャットボットにおける安全性のためのヒント（センシティブなトピックに対する分類器を含む）：
  https://parl.ai/projects/safety_recipes/

ツールや評価のためのプラットフォーム：

- スタンフォード大学の基盤モデル研究センターによるLLMのベンチマーク、HELM：
  https://crfm.stanford.edu/helm/latest/
- EleutherAI LLM Evaluation Harness：
  https://github.com/EleutherAI/lm-evaluation-harness
- Huggingface Hubは、オープンソースのモデル、データセットをホストし、開発者がセーフガードを共有し、ベンチマーク情報にアクセスするための場所である：
  https://huggingface.co/docs/hub/index
- Credo.AIによるGenAI Ops Toolsデータベース：
  https://www.credo.ai/gen-ai-ops-landscape

報告リソース

問題、違反、トラブルに関する情報をお持ちの方は、報告リソースを利用してコミュニティの安全維持にご協力ありたい。

- モデルの問題を報告する：
  github.com/facebookresearch/llama
- モデルによって生成されたリスクのあるコンテンツを報告する：
  developers.facebook.com/llama_output_feedback
- バグやセキュリティ上の懸念を報告する：
  facebook.com/whitehat/info
- 利用規定違反またはLlamaの無許諾利用を報告する：
  LlamaUseReport@meta.com

責任ある生成AIの構成要素を組み合わせる

モデル開発の各段階には、AI機能の安全性を高められる機会が存在する。しかし、これらの段階が相互に関連していること、各段階での決定が他の段階にどのような影響を与えるかについて認識することは極めて重要なことである。責任あるAIエコシステムの構築には、各構成要素を改良し、それらが効果的に機能するようにする継続的な取り組みが必要である。

これらのコンポーネントを一体的に実装するための主な考慮事項を以下に示す：

- **全体的最適化：**各コンポーネントには特定の役割と最適化の目標があるものの、それぞれのコンポーネントは孤立した存在ではない。他のコンポーネントとの相互作用を考慮せずに、あるコンポーネントを過剰に最適化しすぎると、最適とは言えない結果を招く可能性がある。例えば、安全性のためにトレーニングデータを過剰にフィルタリングすると、モデルが安全でないコンテンツを適切に認識・処理できない可能性があるため、後のファインチューニングが効果的でなくなる恐れがある。それこそが、開発ライフサイクル全体を通じて各レイヤーで安全性に対するリスクを軽減することが、高性能で責任ある製品を生み出すために不可欠な理由である。
- **開発の各段階における目標の整合性：**開発者がターゲットとするユースケースに最適化したプロダクトを生み出すためには、プロセスの各段階を進む際の一貫した目標と成果が不可欠である。データ収集段階からユーザーからのフィードバックに至るまで、全体的な目標を常に念頭に置いてほしい。
- **フィードバック／エラーからの学習プロセスの標準化：**反復的にモデルを開発するという考え方を取り入れることは非常に重要である。新しい学びをその後のモデルトレーニングに取り入れるための、明確なプロセスを確立すべきである。このプロセスには、一貫したフィードバック分析、特定した問題の優先順位付けと、モデルトレーニングの次の反復における学習の体系的な適用が含まれるべきである。

生成AIの領域は複雑で、進化し続けており、可能性に満ちているが、リスクと隣り合わせでもある。デメリットを軽減しながらメリットを引き出す鍵となるのは、責任あるAIの実践である。この実践は、技術の複雑さ、ユーザーや社会への潜在的な影響、継続的な改善努力の重要性を理解することから始めるべきである。

透明性、説明責任、ユーザーへの権限付与の原則を受け入れ、継続的な学習と改善に取り組むことで、AI機能を革新的で有用なものにするだけでなく、責任と敬意にあふれた技術にすることができる。このガイドが、責任あるAIの実践のための貴重なツールとなることを願っている。

Code Llamaの紹介

Code Llamaは、Llama 2に基づくコード用の大規模な言語モデルファミリーであり、オープンモデルの中では最新の性能を発揮し、コード補完機能、大規模な入力の文脈のサポート、プログラミングタスクに対するゼロショット指示実行といった能力を持つ。幅広い用途に対応するため様々なモデルを用意している：基盤コードモデル（Code Llama）、Python特化モデル（Code Llama - Python）、および指示実行モデル（Code Llama - Instruct）などがあり、それぞれ7B、13B、34Bのパラメータを持つ。すべてのモデルは16kのトークン長で学習され、またトークン長を100kトークンに至るまで大きくすることで性能の改善が見られた。7Bと13BのCode LlamaとCode Llama - Instructのバリアントは、周囲のコンテンツに基づいたコード補完をサポートする。Code Llamaは、Llama 2を一般公開されているコードとコードに関連する自然言語データセットにより訓練することによって開発された。

責任を持ってAIモデルを構築することは私たちにとって重要であり、Code Llamaをリリースする前にさまざまな安全対策（Llama 2と同様）を講じた。さらに特別に、サイバーセキュリティとマルウェア開発の専門家に、Code Llamaモデルが、熟練していない敵対者によるマルウェア開発を可能にする能力を備えているか評価するよう依頼した。モデルの訓練、アーキテクチャとパラメータ、評価、責任あるAIと安全性についての詳細情報は、研究論文と責任ある使用ガイドを参照のこと。

本付録では、下流コーディング関連機能の責任ある開発において、開発者が考慮すべきコーディング特有のベストプラクティスを概説する

Code Llamaの潜在的ユースケース

Code Llamaには大まかに言うと2つのユースケースがある：

1. **基盤モデルのユースケース：**より多くのデータで訓練し、Code Llamaを変異させたモデルを作成する。例えば、他のプログラミング言語（C++、Java）の追加、コンテキスト長の増加、モデルの量子化によるレイテンシの短縮など。

2. **指示モデルのユースケース：**より多くの指示データを追加して、指示に従うモデルの能力を向上させ、英語以外の指示を理解する能力を拡張し、スタンドアロンなコードボットの作成、既存のサードパーティ製品への統合など。

基盤コードモデルも指示コードモデルも、汎用的な大規模言語モデルとして使用するようには設計されていない。そのようなシナリオについては、Llama 2を参照のこと。

1

基盤モデルのユースケース

このモデルは、プログラミング向けに特化した基盤となる大規模言語モデルのさらなる研究の探求に利用できる。Code Llamaをファインチューニングする際には、下流モデルの責任ある開発に関する重要なガイダンスを概説するLlama 2の「責任ある使用ガイド」を参照のこと：(i) コンテンツポリシーとリスク軽減策の定義、(ii) データの準備、(iii) モデルのファインチューニング、(iv) パフォーマンスの評価と改善、(v) 入出力レベルリスクへの対応、(vi) ユーザーとのインタラクションにおける透明性と報告メカニズムの構築。さらに、開発者はCode Llamaの上に構築する際に、特定のユースケースに沿ったコード固有のベストプラクティスを考慮する必要がある。

ユースケースに応じたコンテンツポリシーを定義

- コンテンツポリシーは、意図された使用法とデプロイの文脈に基づいて、どのようなコンテンツが許可されるかを定義する。このコンテンツポリシーは、潜在的に問題のあるコンテンツを制作する際の制限を説明するものでもある。コードの分野では、モデルはマルウェア、ウイルス、悪意のあるコードを生成しないようにすべきである。また開発者は、悪質な行為者がどのようにしてモデルに上記の様な結果を出力させたのかを考えるべきである。
- これらの方針は、必要なデータ、アノテーションの要件、安全性に基づくファインチューニングの目標を決定するだろう。また、追加的な安全メカニズムとして、入力レベルおよび出力レベルのセーフガードに適用することもできる。

評価とベンチマーク

- モデルは、その使用目的とエンドユーザーの要求に対して評価されるべきである。具体的には、コードモデルはコード固有のベンチマークに対して評価されるべきである。参考リソースとして、以下のリンクからさまざまなベンチマークを参照できる：
  Papers with Code: Code Generation Benchmarks
- コードによらない安全性評価も推奨される、例えば、TruthfulQA、ToxiGen、BOLDなどのベンチマークでコードモデルを評価することができる。

レッドチームと微調整の考慮事項

- データはエンドユーザーの要求を代表するものでなければならない。例えば、モデルがJavascriptを生成することを意図している場合、ファインチューニングのために選択されるデータセットは、Javascriptに特化していなければならない。開発者はまた、データ中に存在する潜在的に悪意あるコードや悪質なコードに対する制限の調査・設定を検討すべきである。
- 開発者は、訓練に用いられるコードのデータセットのセキュリティとロバスト性の品質が、特定のユースケースに基づいて出力、及びその出力コードが統合されるシステムのセキュリティ要件に、適合していることを確認する必要がある。

- 開発者は、意図的なマルウェアの生成や、意図しない脆弱なコードの導入など、コード固有の領域について安全性調査を行うべきである。レッドチーム領域の専門家との連携により、開発者がプロンプトの意図が明確で、出力がインターネット上ですでに公開されているリソースを超えている場合における、モデルの「悪意のあるコード生成に対するハードルを下げる」能力を評価することができる。
- モデルの出力が本番システムで使用される場合、開発者は、モデルの学習に用いられるコードに関連するセキュリティ上の脆弱性がないことを確認すべきである。このモデルをソフトウェア開発のアシスタントとして使用する開発者やエンドユーザーは、引き続きセキュリティのベストプラクティスに従ってほしい。

特にCode Llamaを使用する場合、Code Llamaはコード関連のタスクに特化しており、他のタスクファミリー（例：一般的な言語モデル）の基盤モデルとしては適切でないことに留意すること。Pythonプログラミング言語と関連性の高い下流のタスクでは、Code Llama - Pythonモデルを使用する方がより適切かもしれない。Code LlamaとCode Llama - Pythonは、指示データでトレーニングされていないため、自然言語による命令に従うようには設計されていない。モデルの潜在的な誤用を避けるためにも、これらのモデルを一般的な自然言語タスクの実行に使用することは推奨されない。

2

指示モデルのユースケース

このモデルは、プログラミングに特化した大規模言語モデルのさらなる応用研究やテストに利用できるだろう。Code Llama - Instructは自然言語の指示を理解する能力を持つ。コード生成タスクにCode Llamaを使用する場合、ユーザーに役立つ安全な回答を生成するためにファインチューニングされたCode Llama - Instructを使用することをお勧めする。特定のユースケースに関連する、入力および出力レベルのリスクへの一般的な対応、ならびにユーザーとのインタラクションにおける透明性および報告メカニズムの構築に関するベストプラクティスについては、『責任ある使用ガイド』全文を参照のこと。

どちらのユースケースにおいても、ユーザーは利用規定を遵守しなければならない。

開発者向け報告リソース

問題、違反、トラブルに関する情報をお持ちの方は、報告リソースを利用してコミュニティの安全維持にご協力ありたい。

- GitHubレポでモデルの問題を報告する
- 開発者ポータルでモデルによって生成されたリスクのあるコンテンツを報告する
- バグバウンティ（バグ報奨金制度）でバグやセキュリティ上の懸念を報告する