Engineer

大規模言語モデル

💬 文章で訓練し、次のトークンを予測するニューラルネットワーク

どのようなものか

大規模言語モデルは、大量の文章で訓練したニューラルネットワークです。行うのは一つの処理で、そこまでの文章を受け取り、次に来る可能性のある各トークンに確率を与えます。トークンは短い文章の断片で、多くの場合は単語全体ですが、単語の一部や句読点一つの場合もあります。回答を書くときは、この一段階の処理を繰り返します。ソフトウェアが、その確率に従ってトークンを選び、文章に加えて再び問い合わせます。モデルが終了を示すトークンを出すか、長さの上限に達するまで続きます。2024年7月に公開されたNISTの生成AIプロファイルは、「LLMは、文や句の次のトークンまたは単語を予測する」と一行で説明します。プロファイル自身が使う略称LLMも、同じものを指します。

近年のモデルを支える構造は、2017年の機械翻訳の論文で説明されたTransformerです。それまでのモデルを構成したリカレント層と畳み込み層をなくし、文章中の各位置から、ほかのすべての位置を直接参照できる、注意機構だけを使いました。そのため、多数のプロセッサーに学習を分散しやすくなり、論文のモデルは、従来よりはるかに短い時間で学習しました。

「大規模」は、学習でネットワークが獲得する重み、つまりパラメーターの数を指します。2020年の論文は、1,750億個のパラメーターを持つモデルを説明しました。それまでのスパースでない言語モデルの、どれよりも十倍多い数です。その規模では、重みをまったく変えずに、入力に示した少数の解答例から新しい課題に取り組めることも示しました。

文章の続きを書くためだけに訓練したモデルは、そのとおり、文章を続けます。それは、頼まれたことを行うのとは違います。2022年の論文は、そのモデルをアシスタントに変える段階を説明しました。人が、一連の依頼に対する望ましい回答を書き、それを使ってモデルをファインチューニングしました。その後、人がモデル自身の回答に順位を付け、その順位でさらに学習させました。人間のフ

ィードバックによる強化学習と呼ぶ方法です。論文の評価では、この方法で訓練した13億パラメーターのモデルの回答が、百倍のパラメーターを持つ元のモデルの回答より、人に好まれました。

概念を判断に生かす

「大規模言語モデルとは？」を実際の判断に生かしましょう。その概念を、開発または購入するシステムに結び付けて考えるお手伝いをします。 使用中のシステムに接続し、自社の専門家が用意した事例でテストし、自社の担当者が監督できる、特定のモデルに縛られないエージェントをつくります。

サービスを見る お問い合わせ →

用語の背後にある前提を検証する

解説とリンク先の実績を参考に、プロジェクトについての質問を整理してください。ツールを選ぶ前に、前提を一緒に検討できます。 268検査を持つリポジトリの検査基盤を構築 — 268 コミット検査と、検査自体の146テスト

268検査を持つリポジトリの検査基盤を構築 お問い合わせ →

なぜ重要なのか

言語モデルは、固定した規則だけでは対応しにくい仕事を担います。自由な文章のメールを正しいフォルダーに分類すること、スキャンした納品書から項目を取り出すこと、顧客の言語で返信の下書きを作ること、長いやり取りを要約することです。以前は人が一件ずつ読む必要があった仕事でも、今はソフトウェアが下書きを作り、人が確かめられます。

役に立つ理由でもある性質が、それを使う機能の開発を難しくします。出力は、通常、確率に従って選ぶため、同じ入力でも、実行ごとに答えが変わり得ます。正しいかどうかにかかわらず、流暢です。NISTは、自信を持って示す誤った出力を、作話と呼びます。よりよく知られる呼び名は、ハルシネーションです。モデルの作り方から自然に生じるものと説明します。学習に使った文章の統計に合う文章を出すため、多くの場合は正しく、ときには正しくありません。回答を正当化しているように見える推論や引用も、作り出すことがあると指摘します。誤答も正答とまったく同じように見えるため、自分の仕事でどれほど正しいかを知るには、その仕事で計測するしかありません。

一つのモデルを選んで終わりでもありません。新しいモデルは定期的に現れ、同じ入力に対して二つのモデルが大きく違う動きをすることもあります。アプリケーションが呼ぶモデルを、いつ変えるかは、その提供者が決めます。言語モデルを使うチームは、自分で制御できない部品に依存しています。長く残るのは、モデルの周囲を整える仕事です。計測するためのケース、呼び出してよいツール、モデルが単独で行ってよいことを定める規則です。

NISTのプロファイルは、作話、データのプライバシー、情報セキュリティー、知的財産など、生成AIの十二のリスク領域を挙げます。開発者よりも、リスクを管理する組織に向けた文書ですが、モデルを使う機能に何を扱わせてよいかを決めるチームの確認表として使えます。

実際の活用例

言語モデルを使う機能は、プロンプトではなく、書き残したケースから始めます。各ケースは、実際の入力と、その仕事を知る人が受け入れる回答です。コードと並べて、バージョン管理するファイルに置きます。

```
# cases/delivery-notes.yaml
- id: note-0142
  input: scans/note-0142.txt
  expect:
    supplier: "Example Freight"
    reference: "DN-58213"
    pallets: 6
- id: note-0157
  input: scans/note-0157.txt
  expect:
    supplier: "Sample Haulage"
    reference: "DN-59002"
    pallets: 14
```

どの種類の仕事をどのモデルが受け持つかは、設定です。そのため、小さく速いモデルに抽出を任せ、より大きなモデルに、判断を必要とする返信の下書きを任せられます。

```
# routing.yaml
jobs:
  extract-delivery-note:
    model: small-fast
    cases: cases/delivery-notes.yaml
  draft-claim-reply:
    model: large
    cases: cases/claim-replies.yaml
```

プロンプト、ツール、モデルのどれを変えても、公開する前に、すべてのケースで実行します。各ケースに評価点を付け、履歴を残すため、一つの修正が、ほかのケースを気付かれずに壊すことを防げます。新しいモデルが出たときは、

`routing.yaml` の一行を変え、ケースを実行するだけで試せます。変更の判断は、発表ではなく、評価点に基づきます。

このサイトは、業務に合わせたAIエージェントを、そのように説明しています。モデルの選択は、コードではなく設定です。分類と抽出には小さく速いモデルを、判断には大きなモデルを使います。仕事は、三十から五十の実例と正答を書き出すことから始め、プロンプト、ツール、モデルのどれを変更しても、公開前にすべてを実行します。モデルは交換できる状態に保ち、コード、ケース、データは顧客に属するとも説明します。使うモデルが変わっても、エージェントは顧客自身のものです。

問題が起きるところ

自信のある回答が、間違っています。 手元にない契約条項について問われたモデルは、正答と同じ確かな口調で、その条項を説明してしまい、実在しない出典を引用するかもしれません。回答の根拠にする文書を渡し、引用を求め、すべての引用が実在する文書と箇所を指すことを、コードで検査します。書いたケースで誤答率を計測し、実際の判断や行動に影響する回答は、それに基づく処理を始める前に、人へ回します。

データの中に命令が隠れています。 メール、ウェブページ、アップロードしたファイルをモデルに読ませるアプリケーションは、その文章の作成者にも、モデルへ話しかける機会を与えます。2023年の論文は、実際のアプリケーションで、それを示しました。モデルが読みそうな内容へ埋めた命令で、データを漏らし、アプリケーションの動作を変えさせました。NISTも、生成AIのセキュリティーリスクに、間接的なプロンプトインジェクションを挙げます。モデルが読むものは、すべて信頼できない入力として扱い、権限は、プロンプトではなくツールで強制します。ツールには、モデルが代理する本人の権限だけを与え、取り消せない操作には、人の承認を求めます。

使っているモデルが、手元の変更なしに変わります。 新版や別の提供者は、同じ入力に違う答えを返すことがあり、自分のコードを変えていなくても、先月は合格した回答が、今日は失敗します。提供者が用意している場合は、モデルの版を厳密に指定します。モデル変更も含め、どの変更でも、利用者へ届ける前に、ケースのファイル全体を実行します。

コードが解析できない出力になります。 JSONを求めたモデルが、JSONの後に親切な一文を加えたり、項目を欠かしたり、数値を文字列として書いたりします。期待する構造をJSON Schemaで記述し、すべての回答を照合します。合わない回答は、データではなく、再試行するか人へ渡す、エラーとして扱います。

プロンプトとログに、個人データが残ります。 外部で運用されるモデルへ送ったものは、自身のシステムから外へ出ます。プロンプトと回答のログは、要求が終わっても、長く個人データを残します。NISTは、モデルが学習データ内の機微な内容を繰り返すこともあり、データの記憶と呼ぶ問題だと述べます。仕事に必要な項目だけを送り、プロンプトのログには、その中のデータと同じ保存期間とアクセス規則を適用します。個人データを一つでも送る前に、提供者の保存と学習に関する条件を読みます。

参考資料

- NIST AI 600-1：生成AIプロファイル — AIリスク管理フレームワークを補う、NISTの2024年7月の文書です。作話など、生成AIの十二のリスクと、それぞれへの対策案です。
- 注意機構だけでよい — 現在の言語モデルを支えるTransformerを紹介した、2017年の論文です。
- 言語モデルは、少数の例から学べる — 入力中の例から新しい課題に取り組む、1,750億パラメーターのモデルについての、2020年の論文です。
- 人間のフィードバックで、言語モデルを命令に従うよう訓練する — 書いた回答と、順位を付けた出力による追加学習です。文章予測器からアシスタントへ移る段階を説明します。
- 想定していた使い方とは違う：間接的なプロンプトインジェクションで、LLMを組み込んだ実際のアプリケーションを侵害する — モデルが読むデータに命令を埋める攻撃を、実際のアプリケーションで示しています。

用語集

実際の制約に合わせて選ぶ

業務の流れ、既存のインターフェース、変更を必要とする利用者をお知らせください。検証可能な納品内容と、お客様のチームが保守を続けられる引き継ぎ内容を定められます。

料金を見る お問い合わせ →

用語を実務に結び付ける

「大規模言語モデルとは？」がお客様の組織にどう当てはまるかをご相談ください。現在のシステムの具体例が一つあれば、話を始められます。

サービスを見る

welcome@engineer.company +45 50 10 61 84 デンマーク、コペンハーゲン

このページを持ち歩く

連絡先カード

心を込めて丁寧に制作： D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

著作権 © 2025 – 現在 · Engineer ApS · すべての権利を留保します。

印刷元： https://engineer.company/ja/glossary/large-language-model/