Ashiras, inc.

AI / WEB / DATA ENGINEERING & CONSULTING

NATIONWIDE PROJECTS

FROM AI USAGE TO AI ENGINEERING

生成AIを使うから設計するへ

AI・モダンWeb技術・データマネジメントの知見を、研究だけで終わらせない。生成AIの振る舞いを理解し、業務で再現できる仕組みとして設計・実装します。

OUR APPROACH

生成AIの最適解は「プロンプト」から「生成軌道」へ

生成AIは確率的に振る舞います。だから重要なのは「完璧な一発回答」を期待することではなく、入力・制約・検証・人間の介入点を含めた外側の実行構造を設計することです。

RESEARCH → PRACTICE

仕組みを理解し、実装可能な設計へ。

BERT意味空間 / Semantic Drift / AI Orchestration / Human Gate

01

AIソリューション

AI SOLUTIONS

AIパイプライン、プロンプト設計、複数AIの実行構造まで。**生成AIを業務で安定して使う仕組み**を設計します。

02

Webアプリ開発

MODERN WEB DEVELOPMENT

React / Angularを中心に、変更に強い構造とAI活用を両立。**設計から実装まで一気通貫**で支援します。

03

技術コンサルティング

ENGINEERING CONSULTING

DDD、アジャイル、データベース、データマネジメント。**技術を使いこなせる組織づくり**まで伴走します。

ASHIRAS ENGINEERING

AIを「便利な道具」で終わらせない。

研究・設計・実装・検証をつなぎ、再現性のある開発プロセスへ。

UNDERSTAND

生成AIの振る舞いを理解する。

<Transformer、意味ドリフト>

BERT / Softmax / Drift

→

DESIGN

入力、制約、役割、停止条件、人間の確認点を設計する。

Prompt / Agent / Gate

→

IMPLEMENT

Web・DB・AIを接続し、現場で運用できるシステムに落とす。

React / API / Data

ASHIRAS

アシラス株式会社 / Ashiras, inc.

AI・Web技術・データマネジメントの研究開発 / システム開発 / 技術コンサルティング

全国各地のプロジェクトで、設計から実装まで伴走しています。

078-798-5203 | 090-7200-4530

PROJECT / CONSULTING

ashiras.io

EXHIBITION DEMO

2歳児AI

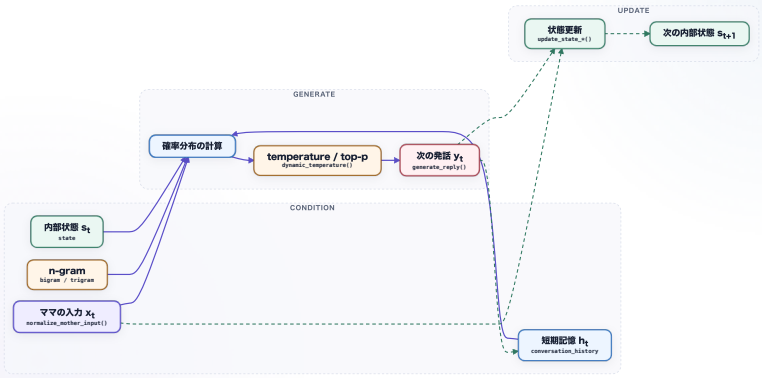
小さな生成AIで見る「状態・記憶・確率」

同じ質問でも、毎回答えが違うのはなぜ？

POINT

入力だけでは決まらない

全体の流れ



返答は、「ママの入力」だけでは決まらない。

今の入力、直前までの会話、内部状態、言葉のつながりを合わせて**「次に何を言いそうか」**の確率分布を作り、そこから発話を生成します。

$$y_t \sim P(y_t \mid x_t, s_t, h_t, T_t)$$

x_t：入力　/　s_t：内部状態
h_t：短期記憶　/　T_t：temperature

1 ママの入力 x_t `normalize_mother_input()`

「ご飯→まんま」「犬→わんわん」のように、2歳児AIが扱える小さな語彙へ入力を寄せます。

普段の生成AI　**Tokenizer / 入力表現への変換**

2 内部状態 s_t `state`

おなか・眠気・甘え・好奇心・不機嫌を数値で持ち、同じ入力でも状態によって出やすい言葉を変えます。

普段の生成AI　**文脈依存の内部表現を見える形に単純化**

3 短期記憶 h_t `conversation_history`

最近の会話ほど強く残し、少し前に出た言葉を次の発話にも出やすくします。古い履歴ほど影響は弱まります。

普段の生成AI　**会話コンテキスト（生成中はKV cacheも利用）**

4 n-gram `bigram / trigram`

「おなか→すいた」「ママ→だっこ」のように、コーパスから次に続きやすい語を数えた小さな言語モデルです。

普段の生成AI　**Transformer が担う次トークン予測の超簡易模型**

5 確率分布の計算

n-gramの基本確率に、現在の入力・短期記憶・内部状態を反映して、次の語の候補と重みを作ります。

普段の生成AI　**logit → softmax → 次トークン分布**

6 temperature / top-p `dynamic_temperature()`

確率分布の「鋭さ」と候補の範囲を調整。完全なランダムではなく、もっともらしい範囲で揺らぎます。

普段の生成AI　**生成時のサンプリング制御**

7 次の発話 y_t `generate_reply()`

確率分布から語を一つずつ選び、短い発話を作ります。返答は固定値ではなく、サンプリング結果として現れます。

普段の生成AI　**Autoregressive な next-token generation**

8 状態更新 `update_state_*()`

入力と自分の発話によって次の状態 s_{t+1} を作ります。そのため会話を続けると出力傾向も変化します。

普段の生成AI　**出力が次ターンの文脈になる ※重み学習ではない**

このデモで見てほしいこと

生成AIは「正解を取り出す」のではなく、条件から確率分布を作り、そこから次の言葉を選び続ける。

2歳児AIは、その振る舞いを「状態・記憶・確率」が目に見える形まで小さくした観察用モデルです。

詳しい解説・コード

zenn.dev/albatrosary/articles/03c0d57df1ba03

EXHIBITION DEMO

BERTで見る意味空間

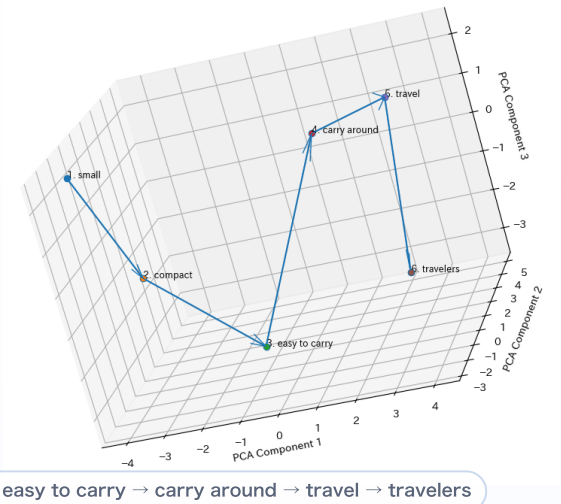
768次元の意味表現を3Dで「見える化」

文章を少しずつ変えると、意味はどこへ動く？

POINT

小さな変化は、積み重なる

3D意味空間 - semantic trajectory



small → compact → easy to carry → carry around → travel → travelers

一文ずつは自然でも、意味の中心は少しずつ移動する。

BERTは文章を高次元のベクトルとして表現します。このデモでは文章を少しずつ変え、 **意味表現が空間上をどう移動するか**を観察します。

$$v(x) \in R^{768}$$

$$d(x_o,x_i) = 1 - cos(v_o,v_i)$$

3本の軸はPCAで作った可視化用の成分です。軸そのものに「大きさ」「旅行」といった直接の意味はありません。 距離はPCA後ではなく、元の高次元表現で計算します。

1 文章をBERTへ入力 `BertTokenizer`

文章をトークンに分け、BERTが扱えるID列へ変換します。単語そのものではなく、文脈を含む入力として処理します。

ここを見る **意味表現の出発点を作る**

2 768次元の文脈表現 `last_hidden_state`

各トークンは周囲の文脈を反映した768次元のベクトルになります。同じ語でも文脈が変われば内部表現も変化します。

BERT **Transformerによる文脈依存表現**

3 文全体を1ベクトルに `mean pooling`

特殊トークンを除き、各トークンの最終隠れ状態を平均して文ベクトルを作ります。このデモ用の単純な代表値です。

注意 **BERT標準の唯一の「文ベクトル」ではない**

4 コサイン距離で測る `cosine distance`

最初の文章ベクトルとの向きの違いを測り、どれだけ意味表現が離れたかを元の高次元空間で確認します。

重要 **「見た目」ではなく元ベクトルで測定**

5 768次元 → 3次元 `PCA(n_components=3)`

人間には768次元を直接見ることができないため、PCAで3次元へ圧縮します。3D表示は「意味の移動」を見るための地図です。

可視化 **PCA軸そのものを意味として読まない**

6 意味ドリフトの軌跡 `semantic trajectory`

small → compact → carry → travel のような小さな意味変化をつなぐと、最初の話題から別の意味領域へ移っていく様子が見えます。

このデモ **BERTは生成者ではなく、意味移動の観測役**

このデモで見てほしいこと

意味は固定されたラベルではなく、高次元空間上の位置として表現される。小さな変化でも、連鎖すれば最初の意味から離れていく。

BERTの内部表現を使い、文章系列の「意味の移動」を3Dで観察するデモです。

デモ構成

`bert-base-uncased`
Transformers / PyTorch
PCA / cosine distance