

人文社会ビジネス科学学術院 ビジネス科学研究群 2025年度 春C

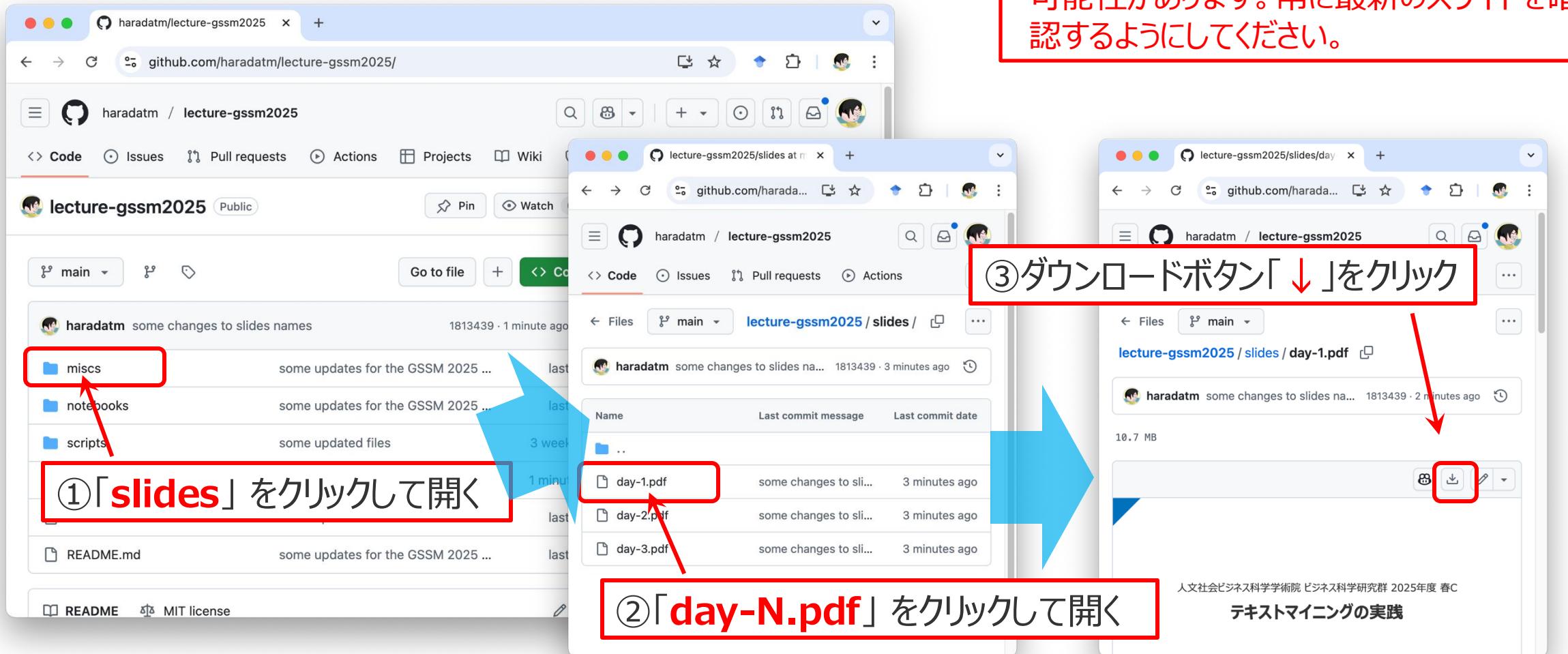
# テキストマイニングの実践

# 講義概要

# 教材について — 講義スライド

- URL: <https://github.com/haradatm/lecture-gssm2025>

注: 講義資料は、講義開始前まで更新する可能性があります。常に最新のスライドを確認するようにしてください。



# 演習の題材 — ホテルのクチコミ分析 (by 楽天トラベル)

● ホテルのクチコミ数: 1,468万件 ※年間約60~80万

The screenshot shows the Rakuten Travel website at <https://travel.rakuten.co.jp/review/>. The main heading is 'お客様の声' (Customer Reviews) with the number '14,648,306' displayed prominently. Below this, there is a search bar for reviews and filters for domestic and overseas stays. A section titled '新着！最新のクチコミ' (Newest! Latest Reviews) lists two entries: 'ダイワロイネットホテル那覇国際通りのクチコミ (1580件)' and 'プレミアホテルーCABIN PRESIDENTー函館のクチコミ (3289件)'. The page also features a banner for '楽天カード入会で2,000ポイントプレゼント' (Rakuten Card membership gets 2,000 points) and various travel-related links.

経年変化:

- 780万件 (2015)
  - 836万件 (2016)
  - 900万件 (2017)
  - 973万件 (2018)
  - 1,042万件 (2019)
  - 1,098万件 (2020)
  - 1,165万件 (2021)
  - 1,237万件 (2022)
  - 1,325万件 (2023)
  - 1,393万件 (2024)
  - **1,468万件 (今回)**
- ※ 2025/5/24現在

## スケジュール

### day 1

- 講義(後半) – 自然言語処理の最新動向

### day 2

- 講義 – テキストマイニングの手順
- 講義&演習 – データ理解
- 講義&演習 – テキスト解析 (1)

### day 3

- 講義&演習 – テキスト解析 (2)
- 講義&演習 – テキスト分析 (1)

### day 4

- 講義&演習 – テキスト分析 (2)

### day 5

- テキストマイニングツール紹介 – TMS
- ラップアップ – Q&A

EXCEL

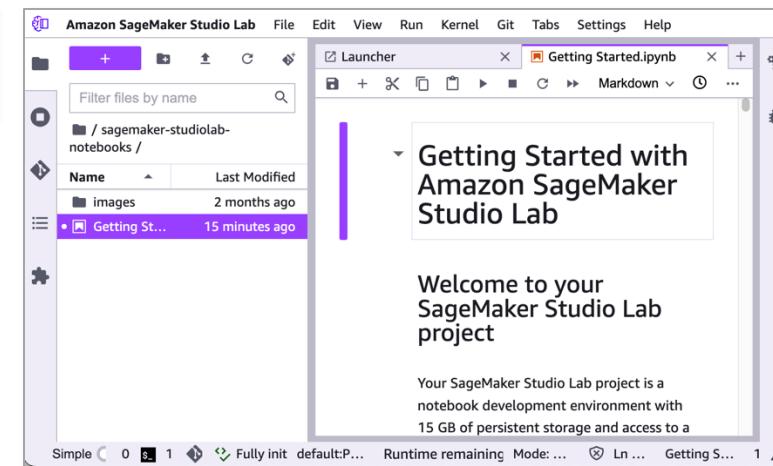
Google Colab

# 無償で利用できる機械学習環境

## ● 機械学習の教育・研究を目的とした研究用ツール

### Colaboratory

<https://colab.research.google.com>



### SageMaker Studio Lab

<https://studiolab.sagemaker.aws/>

演習で使用  
↓

	Clab(無償版)	Studio Lab
GPU	T4(16GB)	T4(16GB)
最長実行時間	12時間	CPU:12時間 GPU:4時間
メモリ	12GB	15GB
ディスク	CPU:100GB GPU:78GB	15GB (永続化)
ターミナル	×	○
ランタイムの保存と再開	×	○
費用	無償	無償
その他	Googleアカウントが必要	AWSアカウントは不要 (クレカ不要)

人文社会ビジネス科学学術院 ビジネス科学研究群 2025年度 春C

# テキストマイニングの実践

## day 1 (後半)

# スケジュール

## day 1

- 講義(後半) – 自然言語処理とのLLM

## day 2

- 講義 – テキストマイニングの手順
- 講義&演習 – データ理解
- 講義&演習 – テキスト解析 (1)

## day 3

- 講義&演習 – テキスト解析 (2)
- 講義&演習 – テキスト分析 (1)

## day 4

- 講義&演習 – テキスト分析 (2)

## day 5

- テキストマイニングツール紹介 – TMS
- ラップアップ – Q&A

# 自然言語処理と大規模言語モデル (LLM)

## 目次

- 大規模言語モデル(LLM)はどんなもの?
- 大規模言語モデル(LLM)の成り立ち
- テキストマイニングの未来予想

## 大規模言語モデル (LLM) はどんなもの?

## 動画生成ができる

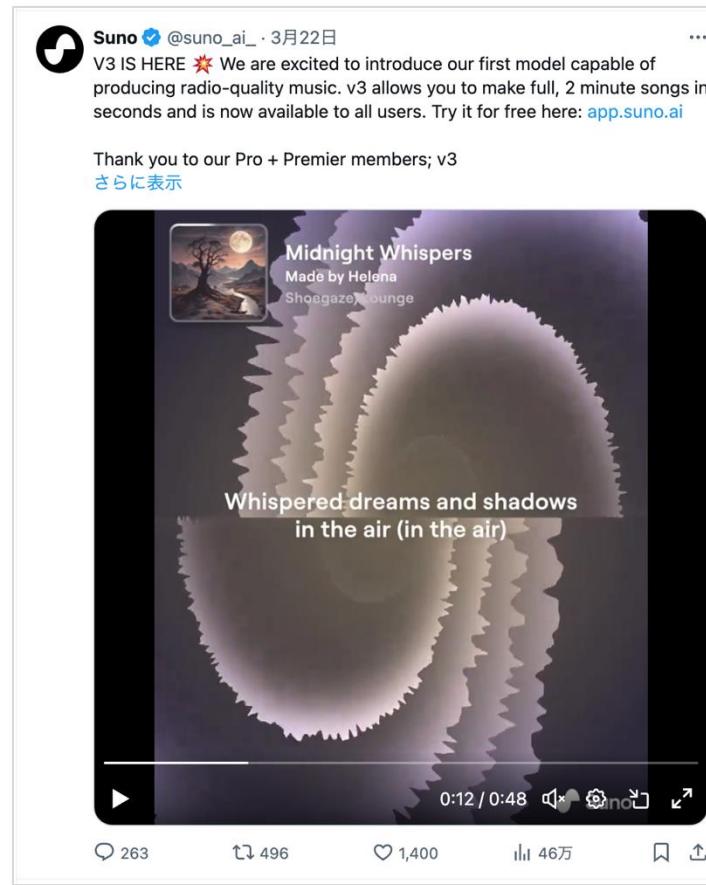


出所: [X \(@Luma AI\)](#)

台本生成ができる  
歌詞生成ができる

大規模言語モデル(LLM)

## 音楽生成ができる



出所: [X \(@Suno\)](#)

## CM が作れる

台本: 生成AI (LLM)  
動画: 生成AI (動画生成)  
歌詞: 生成AI (LLM)  
音楽: 生成AI (音楽生成)



出所: [X \(@CultureCrave\)](#)

※ 上の例は具体例ではなくイメージ。OpenAIが開発中の Soraで作成されたもので、AIが商業目的で活用された実例として歓迎する声が上がった一方で、不快な動画との批判も多数寄せられた。

- 動画生成は、高精細な4K映像生成、リアルな動きの表現、高速・長時間生成が進む
  - OpenAI Sora, Google Veo 3, Kuaishou Kling, Luma Ray2 …



出典: Peng et al. (2025), “[Open-Sora 2.0: Training a Commercial-Level Video Generation Model in \\$200k](#)”

- OpenAI GPT-4o: 自然言語の入力で、何でもできる  
プログラミングができる

コード

次に、以下のコードを使用してワードクラウドを作成します。

```
python
from wordcloud import WordCloud
import matplotlib.pyplot as plt

# テキストを定義
text = """
OpenAI's mission is to ensure that artificial general intelligence (AGI) benefits all humanity. AGI refers to highly autonomous systems that output meaningful results across all domains of knowledge and expertise.
"""

# ワードクラウドの設定
wordcloud = WordCloud(width=800, height=400, background_color='white').generate(text)

# ワードクラウドを表示
plt.figure(figsize=(10, 5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off') # 軸を非表示にする
plt.show()
```

このコードでは、指定したテキストからワードクラウドを生成し、表示します。`WordCloud`オブジェクトのパラメータを調整することで、出力の見栄えをカスタマイズできます。例えば、`background\_color`を変更することで背景色を設定できます。

曖昧な指示でもサンプルコードを生成

表作成 (データ整理) ができる

記事テキストの内容を表形式でまとめてくれる

以下の文章を読んで表にまとめてください

ホンダの軽自動車「N-BOX」が首位から陥落した。日本自動車販売協会連合会（自販連）と全国軽自動車協会連合会（全軽自協）が発表した。現在、生産を停止している。

以下に文章を表にまとめました：

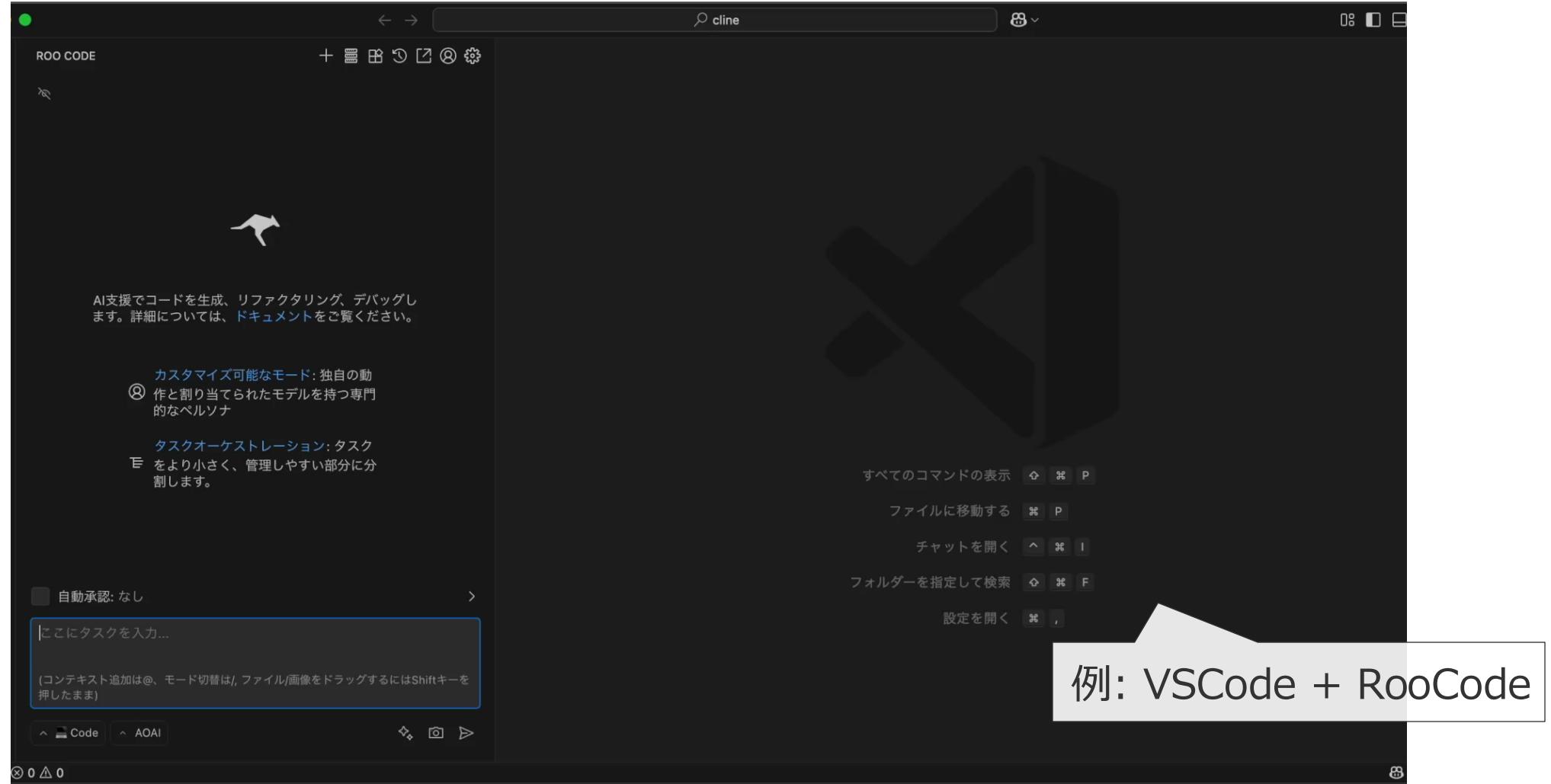
順位	車種	メーカー	車種分類	5月の販売台数	前年同月比	備考
1	スペーシア	スズキ	軽自動車	15,160台	91.9%増	2023年11月に発売した全面改良車が好調な販売に寄与した
2	N-BOX	ホンダ	軽自動車	14,582台	4.4%増	10ヶ月連続首位を維持できず、2位に陥落
3	ヤリス	トヨタ	小型車	-	-	4月の2位から1つ順位を下げた
4	カローラ	トヨタ				
5	シエンタ	トヨタ				
32	プリウス	トヨタ				

前年同月の販売台数も追加してください

以下に前年同月の販売台数も追加した表をまとめました：

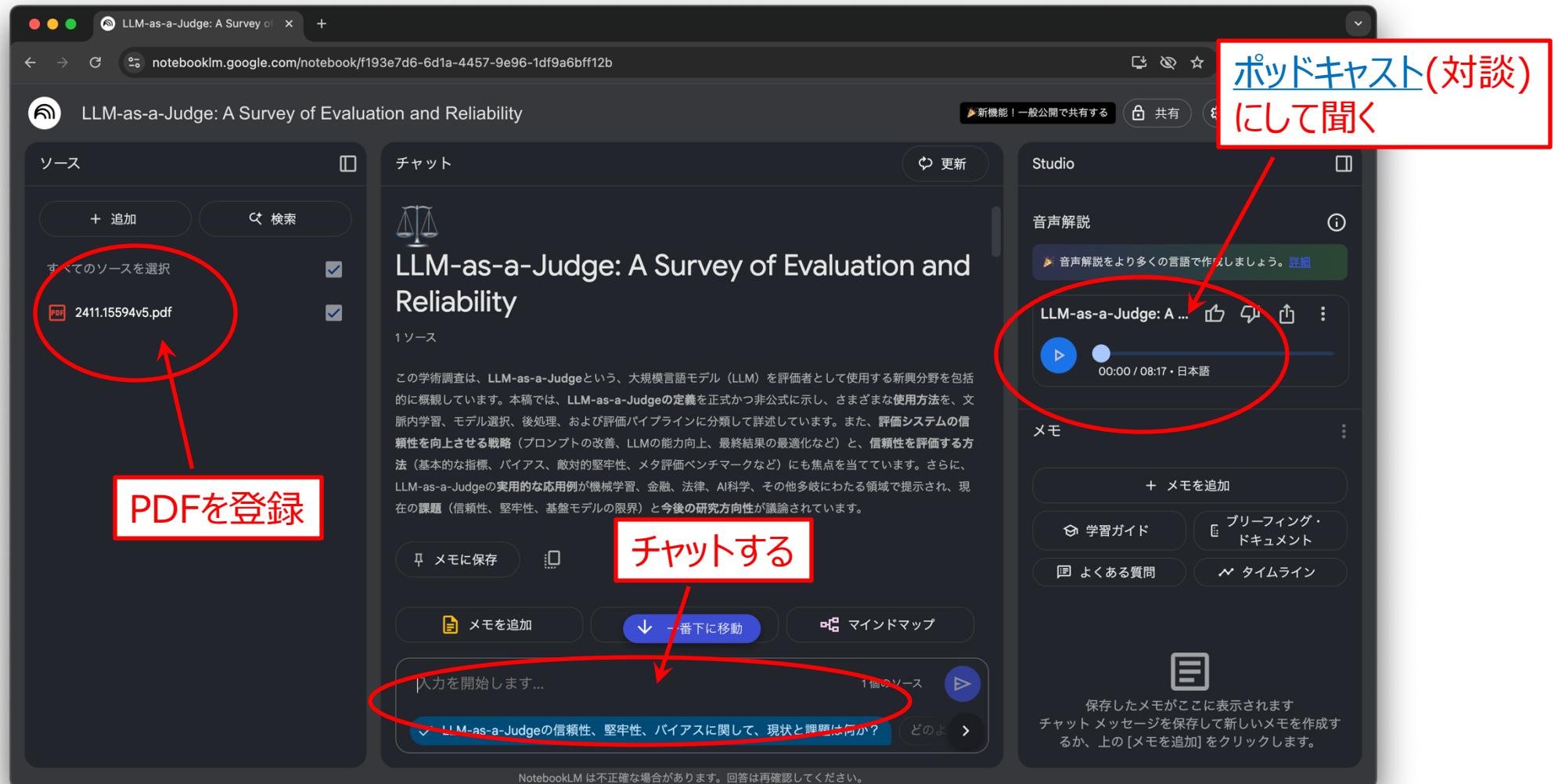
順位	車種	メーカー	車種分類	5月の販売台数	前年同月比	前年同月の販売台数	備考
1	スペーシア	スズキ	軽自動車	15,160台	91.9%増	7,895台	2023年11月に発売した全面改良車が好調な販売に寄与した
2	N-BOX	ホンダ	軽自動車	14,582台	4.4%増	13,968台	10ヶ月連続首位を維持できず、2位に陥落
3	ヤリス	トヨタ	小型車	-	-	-	4月の2位から1つ順位を下げた

## ● OpenAI/Gemini/Claude ほか → AIコーディングエージェント



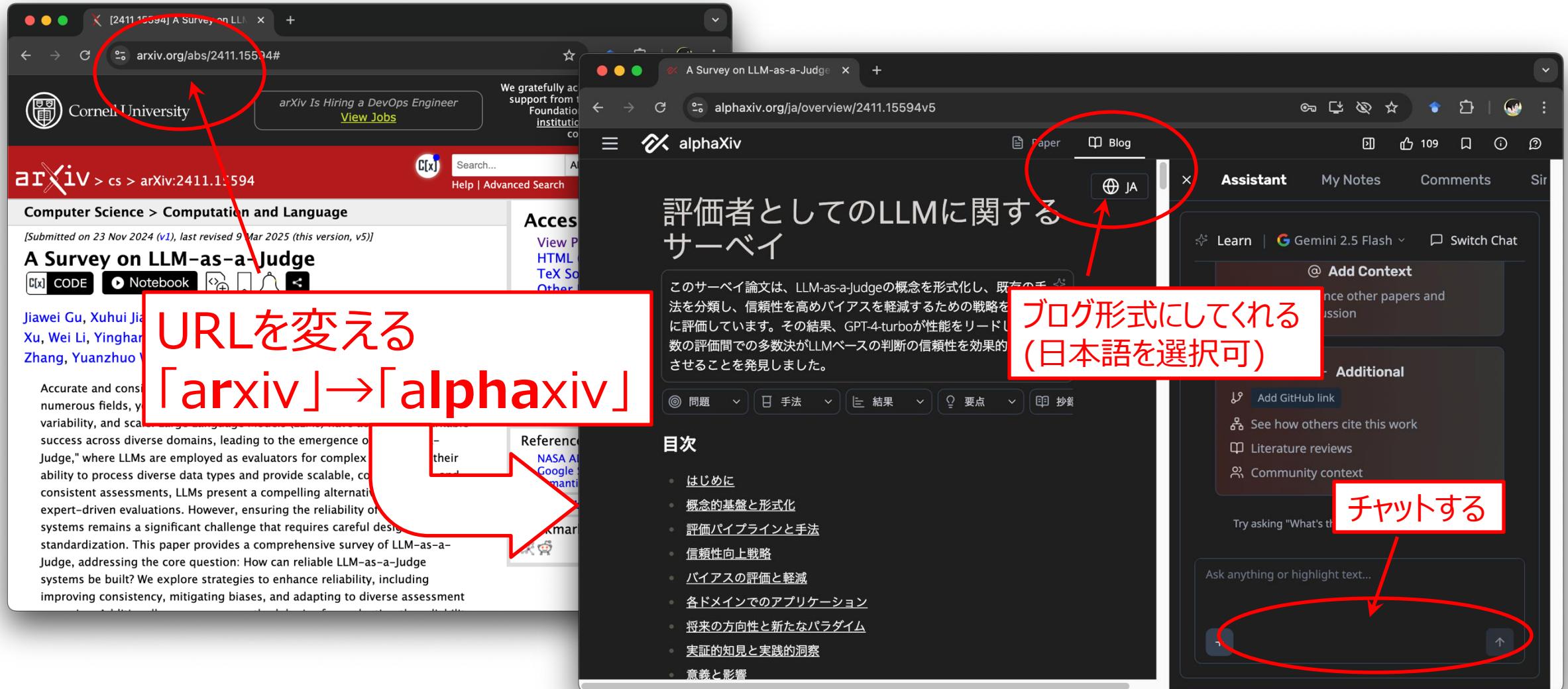
# 大規模言語モデル(LLM)、使ってますか?

- Google's NotebookLM (Gemini 2.5 Pro): 論文の読み方が変わる?!

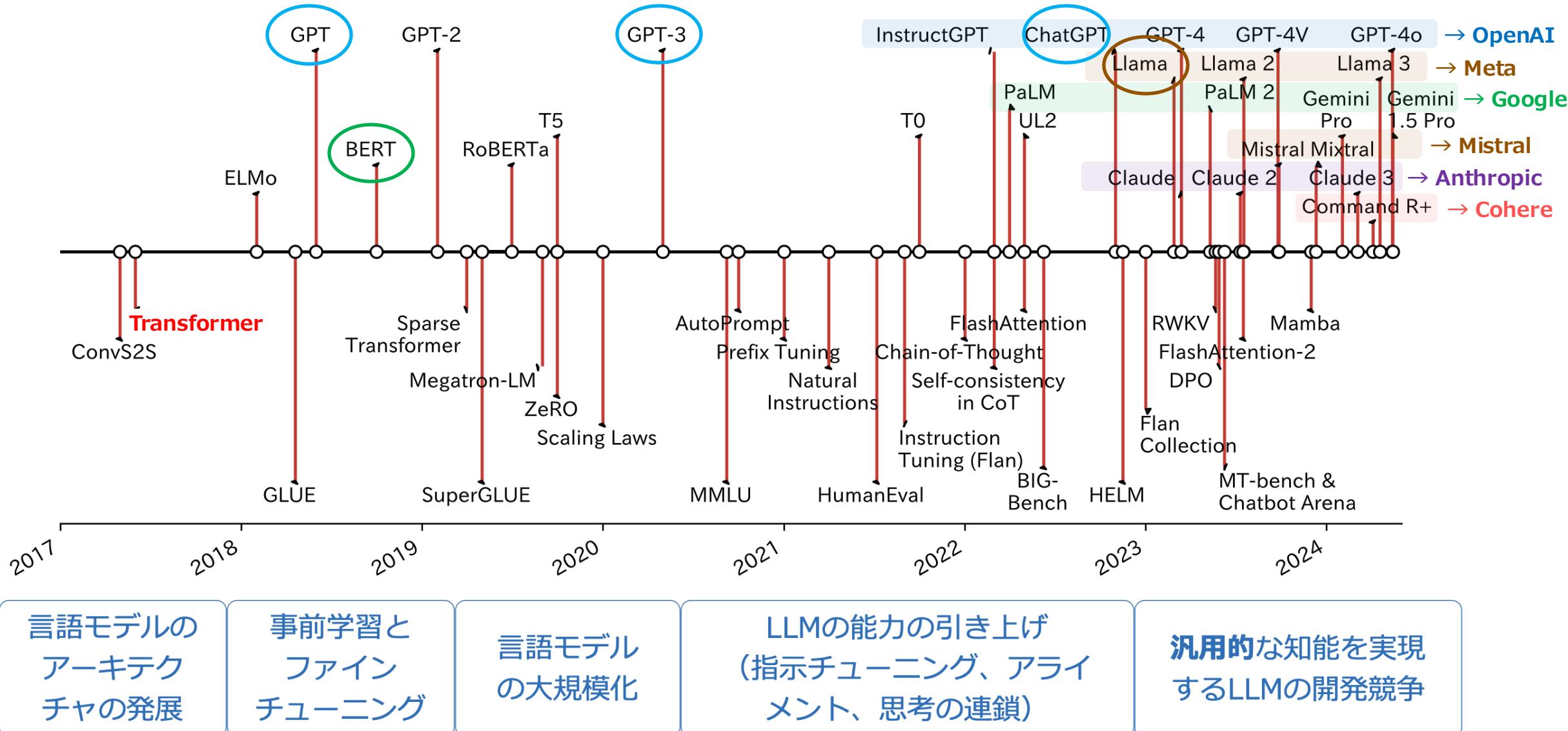


# 大規模言語モデル(LLM)、使ってますか?

## ● [alphxiv](#) (Gemini 2.5 Flush ほか): 論文の読み方が変わる?!



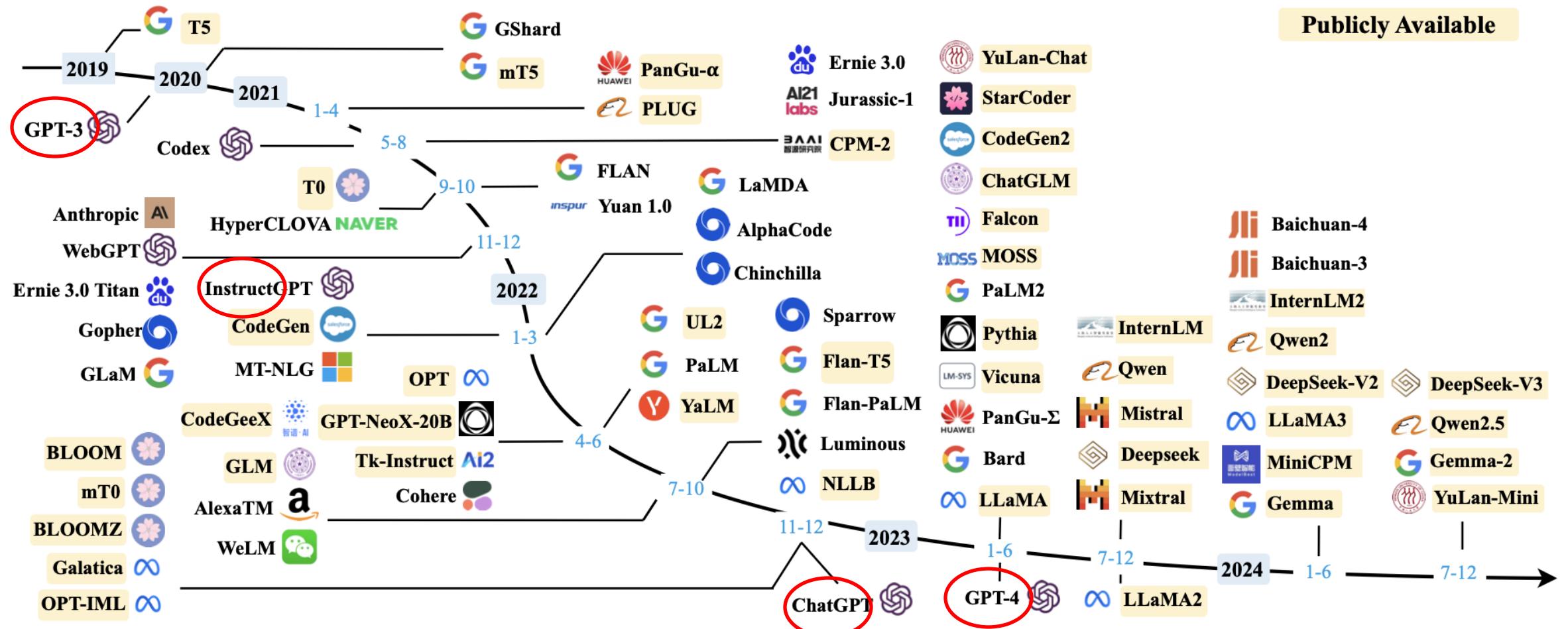
# 大規模言語モデル(LLM)の歴史 (2017-2024年)



出所: [岡崎, 2024] JSAI2024 チュートリアル講演1「大規模言語モデルの開発」

# 大規模言語モデル(LLM)の進展

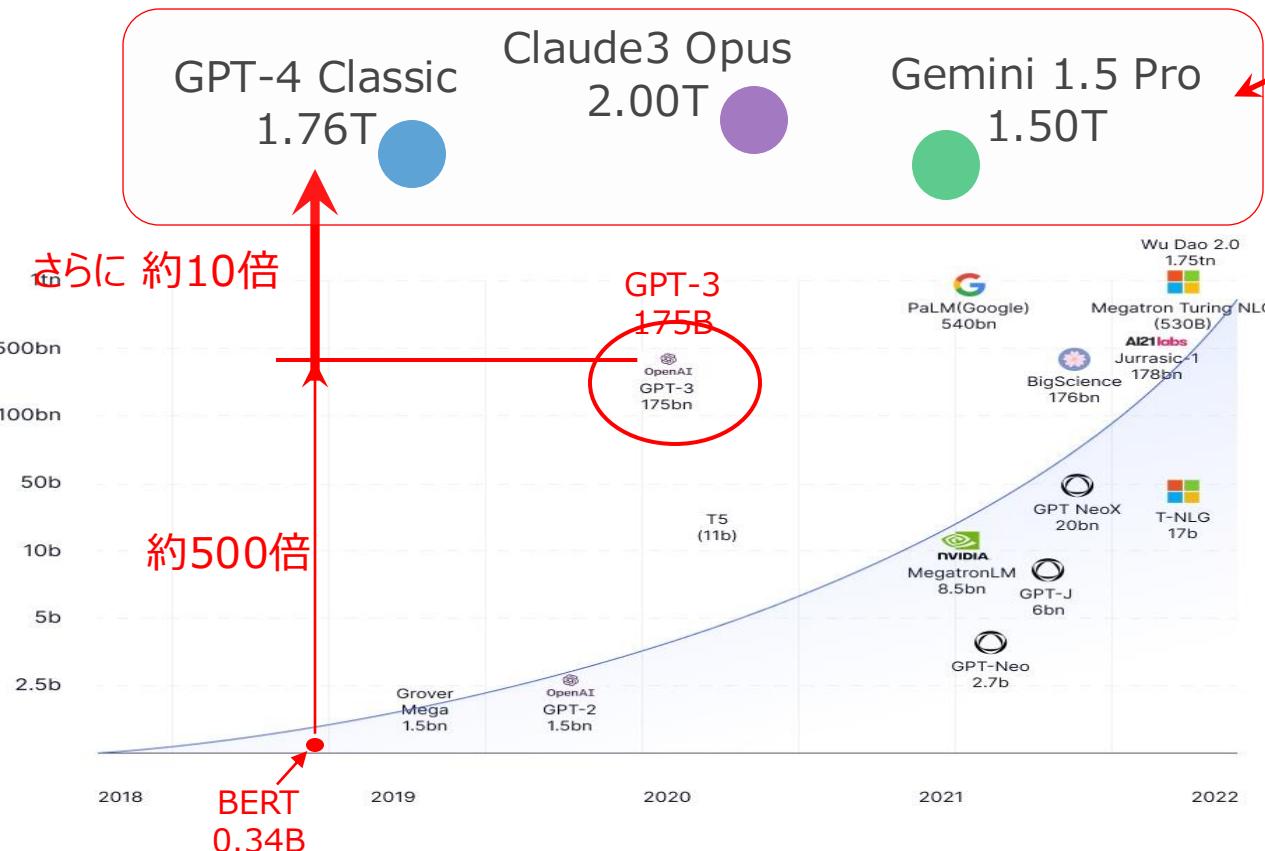
- 2020年のGPT-3登場後、様々な企業から多くのLLMが登場し、技術革新が加速



出典: Wayne Xin Zhao et al. (2023), "[A Survey of Large Language Models](#)", [v16]

# 大規模言語モデル(LLM)は、どのくらい大きいのか

- 大量の計算機資源で、大量データを大きなモデルで学習する  
“Scaling laws” [Kaplan (OpenAI)+, 2020/01]



GPT-4.1 や Gemini2.5 Claude 4 などの最近のモデルは、正確なパラメータ数は公式に明らかにされていないものの、数千億～数兆規模と推測されている

Llama 3 70B (Meta) の学習:

パラメタ数	テキストデータ	計算時間 (GPU時間)	計算環境 (GPU数)
700億 (70B)	約15兆トークン (43TB 程度)	6.4M GPU時間 (731 GPU年)	24K 枚 (H100)



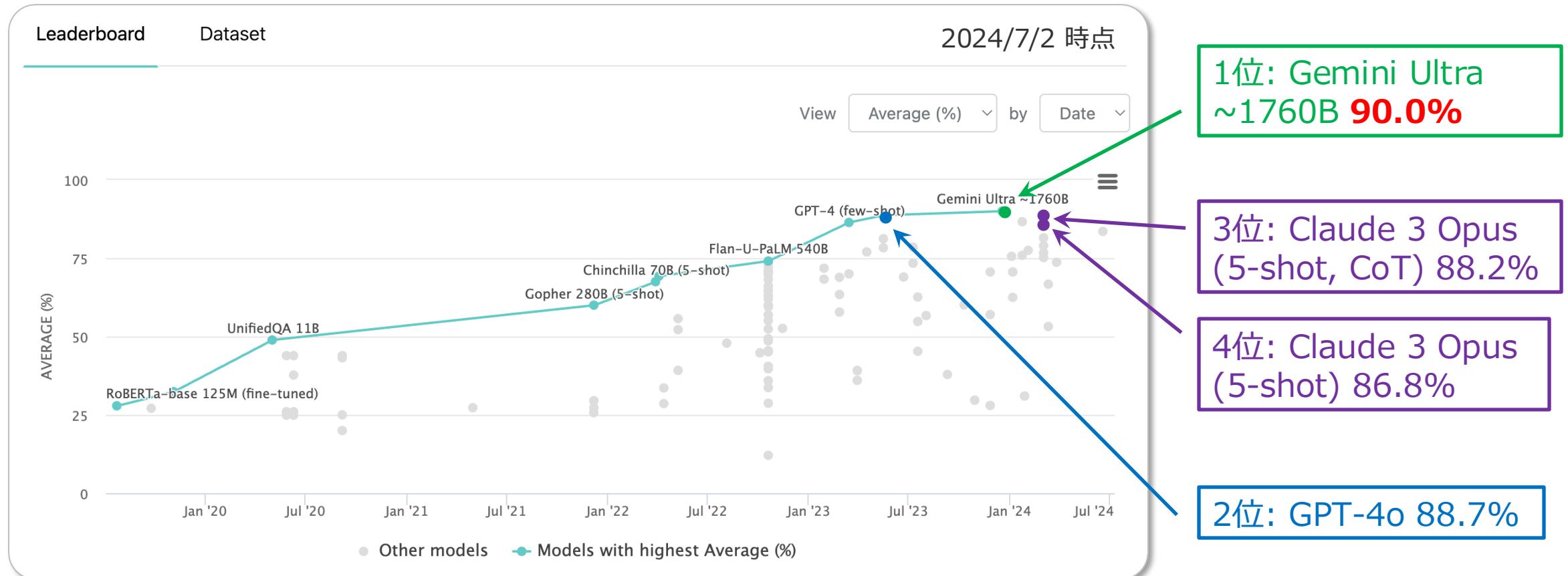
AWS H100 の時間単価で 126億円  
※p5.48xlarge(8GPUs): \$98.32/h

さくら高火力 PHY の月額単価で 34億円  
※高火力PHY(8GPUs): ¥3,046,120/月

出典: [Llama 3 のモデルカード](#)、Andrej Karpathy 氏の X

# 大規模言語モデル(LLM)は、どのくらい賢いか

- 2023年に Google Gemini Ultra が数学、歴史、情報科学、法律などの57の領域の問題で構成される MMLU ベンチマークで、**人間の専門家**によるスコア (89.8%) を超えるスコアを達成

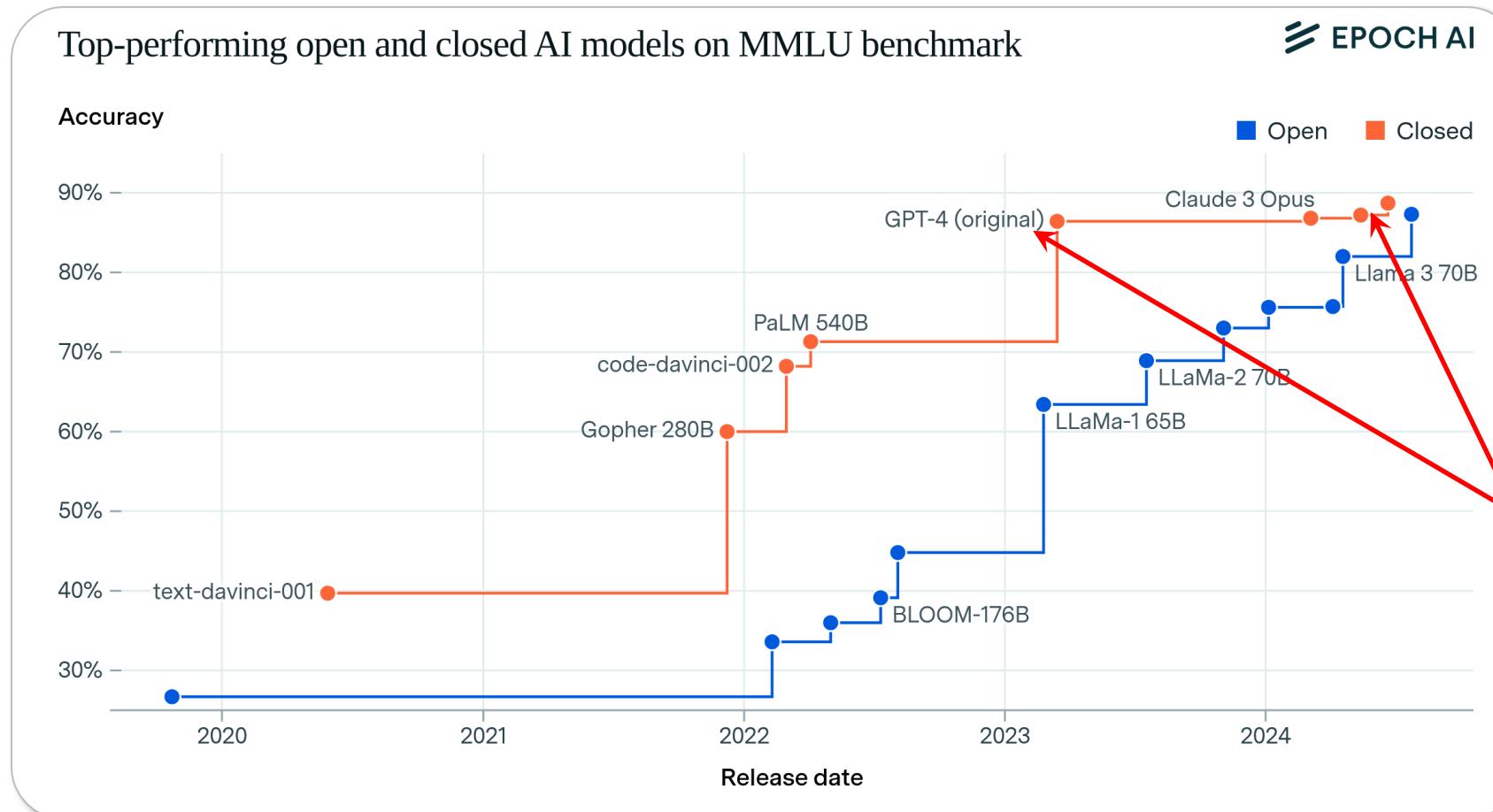


出典: [Paperwithcode の MMLU リーダーボード](#)

# オープンな大規模言語モデル(LLM)

- 2023年はじめ頃から短い期間で性能が大きく改善

- オープン(青)とクローズド(オレンジ)との性能差が狭くなっている → LLMの“民主化”が進む



- ・優れたオープンモデルは、パフォーマンスや学習計算においてクローズドモデルと同等だが、約1年の遅れがある

Llama 3.1 405BのリリースはGPT-4と比較して16か月遅れている

出典: 「[How Far Behind Are Open Models?](#)」

# 日本におけるオープンな大規模言語モデル(LLM) ※ 2024/7月時点

令和6年講義資料

年	月	日	モデル	開発元	規模
2022	11	30	nlp-waseda/gpt2-xl-japanese	早稲田大学	1.5
2023	3	16	okazaki-lab/japanese-gpt2-medium-unidic	東京工業大学	medium
	5	12	retrieva-jp/t5-small-short	Retrieva	S, B, L, XL
	17	cyberagent/open-calm-7b	CyberAgent	1, 3, 7	
	31	rinna/japanese-gpt-neox-3.6b-instruction-ppo	Rinna	XS, S, M, 1, 3.6	
	7	31	rinna/bilingual-gpt-neox-4b	Rinna	4
	8	8	stockmark/gpt-neox-japanese-1.4b	Stockmark	1.4
	10	stabilityai/japanese-stablelm-base-alpha-7b	Stability AI Japan	7	
	18	line-corporation/japanese-large-lm-3.6b-instruction-sft	Line	1.7, 3.6	
	20	matsuo-lab/weblab-10b	東京大学	10	
	29	elyza/ELYZA-japanese- <b>Llama-2</b> -7b	ELYZA	7	
2024	9	28	pfnet/plamo-13b	Preferred Networks	13
	20	llm-jp/llm-jp-13b-v1.0	LLM-jp	13	
	25	stabilityai/japanese-stablelm-base-gamma-7b	Stability AI Japan	7	
	27	stockmark/stockmark-13b	Stockmark	13	
2025	2	cyberagent/calm2-7b	CyberAgent	7	
	2	stabilityai/japanese-stablelm-base-beta-70b	Stability AI Japan	7, 70	
2026	6	moneyforward/houou-instruction-7b-v1	Money Forward Lab	7	
	19	tokyotech-llm/Swallow-70b-hf	東工大, 産総研	7, 13, 70	
	21	rinna/nekomata-14b	rinna	7, 14	
	27	elyza/ELYZA-japanese- <b>Llama-2</b> -13b	ELYZA	13	

年	月	日	モデル	開発元	規模
2024	1	15	lightblue/qarasu-14B-chat-plus-unleashed	Lightblue	7, 14
		31	karakuri-ai/karakuri-lm-70b-v0.1	KARAKURI	70
	2	9	llm-jp/llm-jp-13b-dpo-lora-hh_rhf_ja-v1.1	LLM-jp	13
		13	kotoba-tech/kotomamba-2.8B-v1.0	Kotoba Technologies	2.8
		26	sambanovasystems/SambaLingo-Japanese-Base	SambaNova Systems	7
	3	11	tokyotech-llm/Swallow-MS-7b-v0.1	東工大, 産総研	7
	11	tokyotech-llm/Swallow-MX-8x7b-NVE-v0.1	東工大, 産総研	46.7 (8x7)	
		21	Rakuten/RakutenAI-7B	楽天	7
	4	20	abeja/Mixtral-8x7B-v0.1-japanese	ABEJA	46.7 (8x7)
	24	leia-llm/Leia-Swallow-13b	Studio Ousia	7, 13	
2025		30	llm-jp/llm-jp-13b-v2.0	LLM-jp	13
	5	7	rinna/ <b>Llama-3</b> -youko-8b	rinna	8
		10	Fugaku-LLM/Fugaku-LLM-13B	東工大, 東北大, 富士通, 理研, サイバーエージェント, Kotoba Technologies	13
		16	stockmark/stockmark-100b	Stockmark	100
2026	6	25	elyza/ <b>Llama-3</b> -ELYZA-JP-70B	ELYZA	8, 70
	7	2	tokyotech-llm/ <b>Llama-3</b> -Swallow-70B-v0.1	東工大, 産総研	8, 70

: フルスクラッチで学習したモデル

: 繙続事前学習したモデル (Llama2や3, Mistralなどの高精度な学習済みモデルに追加学習する場合が多い)

出所: [岡崎, 2024] JSAI2024 チュートリアル講演1「大規模言語モデルの開発」に加筆して修正

# オープンな大規模言語モデル(LLM) – 代表的なオープンウェイトモデル

## ● 海外のオープンウェイトモデル

開発元	モデル名 (海外)	ライセンス
Meta	Llama 2 (7B, 13B, 70B)	Llama 2
	Llama 3 (8B, 70B)	Llama 3
	Llama 3.1 (8B, 70B, 405B)	Llama 3.1
	Llama 3.2 (1B, 3B, 11B, 90B)	Llama 3.2
Mistral AI	Mistral 7B	Apache 2.0
	Mixtral (8x7B, 8x22B)	Apache 2.0
	Mistral Nemo (12B)	Apache 2.0
Microsoft	Phi-2 (2.7B)	MIT
	Phi-3 (mini, small, medium)	MIT
	Phi-3.5 (mini, MoE, Vision)	MIT
	Phi-4 (4B, 15B)	MIT
Google	Gemma (2B 7B)	Gemma
	Gemma 2 (9B 27B)	Gemma
	Gemma 3 (1B, 4B, 12B, 27B)	Gemma
	Gemma 3n (E4B, E2B)	Gemma

## ● 国内のオープンウェイトモデル

開発元	モデル名 (国内)	ライセンス
東科大・産総研	Swallow (7B, 13B, 70B)	Llama 2
	Llama-3-Swallow (8B, 70B)	Llama 3
	Llama-3.1-Swallow (8B)	Llama 3.1
	Llama-3.3-Swallow (70B)	Llama 3.3
NII	LLM-jp-13B	Apache 2.0
	LLM-jp-3.1 (1.8B, 13B, 8x13B)	Apache 2.0
ELYZA	ELYZA-japanese-Llama-2 (7B, 13B, 70B)	Llama 2
Cybernetic	モバイルデバイスやPCで動く高性能なモデルも登場	Llama 3
	-8B	Apache 2.0
	CALM3-22B	Apache 2.0

Apache 2.0, MIT, Gemmaは、商用利用・改変・再配布が広く認められているが原著作者の表示が必要。Llamaライセンスは、基本的に商用利用が認められているが、月間アクティブライター数が7億人を超えるとMeta社から別途ライセンスの取得が必要になる。

出典: 「[LLM Supervised Fine-tuningの理論と実践](#)」に加筆して修正

# 自然言語処理はどうなる?

- 人間が書いたり話したりする言葉をコンピュータで処理する技術・研究分野

- 応用技術: 自然言語処理を応用したアプリケーション

テキスト検索

テキスト分類

テキスト要約

情報抽出

機械翻訳

質問応答

対話

など

**ChatGPT で  
いいんじゃない?**

- 基盤技術: 言語を応用タスクで利用しやすい形式に変換する

言語モデル

形態素解析

固有表現抽出

構文解析

意味解析

文脈解析

談話解析

など

## 大規模言語モデル (LLM) の成り立ち

# (再掲) 自然言語処理とは

- 人間が書いたり話したりする言葉をコンピュータで処理する技術・研究分野
- 応用技術: 自然言語処理を応用したアプリケーション

テキスト検索 テキスト分類 テキスト要約 情報抽出  
機械翻訳 質問応答 対話 など

- 基盤技術: 自然言語を応用タスクで利用しやすい形式に変換する

言語モデル 形態素解析 固有表現抽出 構文解析  
意味解析 文脈解析 談話解析 など

# 言語モデル (Language Model: LM) とは

- 単語の並びの生成確率をモデル化したもの（確率的言語モデル）

単語の並び  $\Rightarrow y_1, y_2, \dots, y_T$  生成確率  $\Rightarrow P(y_1, y_2, \dots, y_T)$

- 特定の単語の後に来る単語を予測できる

$$y^* = \operatorname{argmax}_{y \in V} P(\text{日本}, \text{の}, \text{首都}, \text{は}, y)$$

$P(\text{日本}, \text{の}, \text{首都}, \text{は}, \text{ロンドン})$	= 0.00000043
$P(\text{日本}, \text{の}, \text{首都}, \text{は}, \text{パリ})$	= 0.00000082
$P(\text{日本}, \text{の}, \text{首都}, \text{は}, \text{東京})$	= <b>0.00000103</b>
$P(\text{日本}, \text{の}, \text{首都}, \text{は}, \dots)$	= ...

計算された確率が最大値を  
取る単語を選択する  
↓  
**東京**

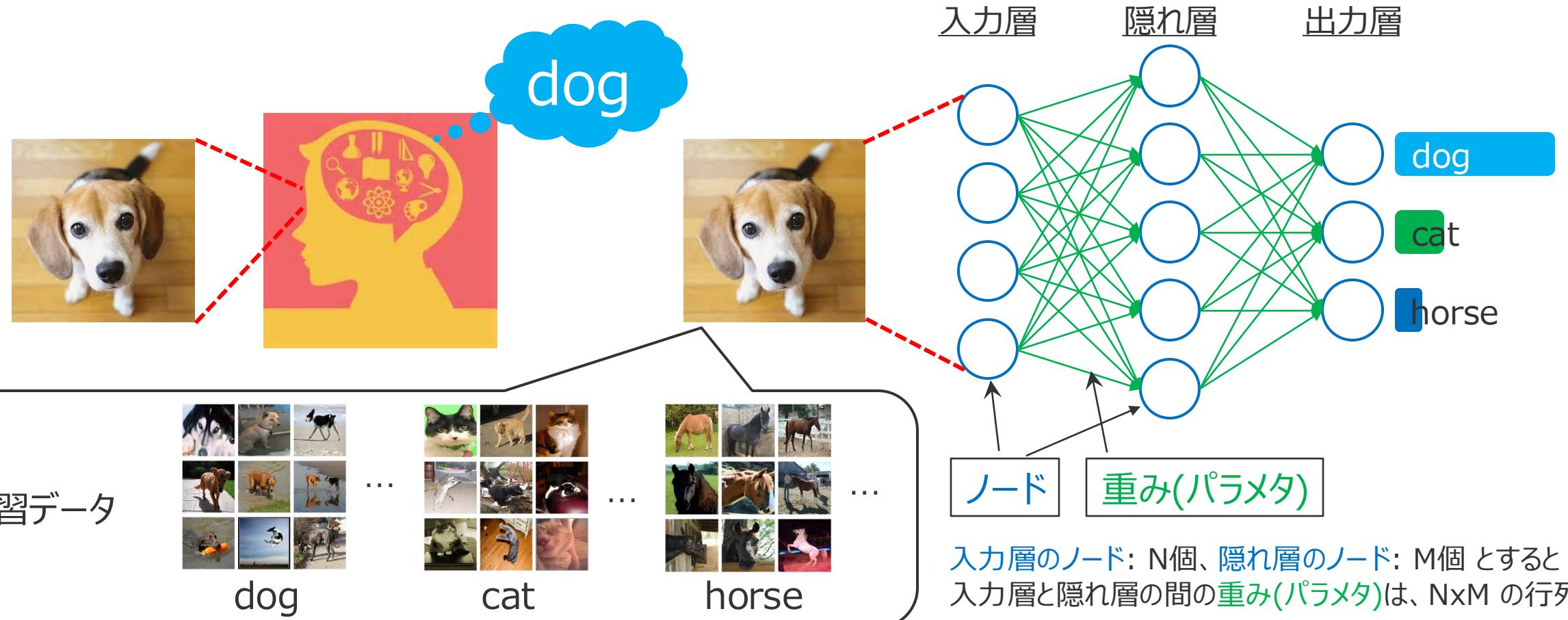
- 単語の並びに従ってモデル化することで文脈も考慮  $\rightarrow$  N-gram言語モデル

$$y^* = \operatorname{argmax}_{y \in V} P(\text{日本}, \text{の}, \text{首都}, \text{は}, y) = \operatorname{argmax}_{y \in V} P(y | \text{日本}, \text{の}, \text{首都}, \text{は})$$

yの前方N-1個の単語(トークン)

# ニューラルネットワークとは

- 脳の神経細胞(ニューロン)の働きを模した機械学習の手法
- ちなみに、機械学習とはデータを学習しパラメータを獲得すること



- 分布仮説 [Harris+, 1954]

- 単語の意味はその周囲の単語から形成されるという仮説  
→ 似た文脈で出現する単語は意味が似ている

文1: 昨日、りんごを食べた。りんごジュースを飲んだ。りんごの皮をむいた。

文2: 昨日、りんごを食べた。ぶどうジュースを買った。ぶどうの皮をむいた。

文3: 昨日、自転車に乗った。自転車を修理した。自転車を買った。

共起による  
ベクトル表現  
[Lin, 2002]

	…	食べる	…	飲む	…	修理	…
りんご	0	1	0	1	0	0	0
ぶどう	0	1	0	1	0	0	0
自転車	0	0	0	0	0	1	0
:							

← 語彙数(数万～数十万)分の疎なベクトルになる →

似てる  
似てない

- 分布仮説 [Harris+, 1954]

- 単語の意味はその周囲の単語から形成されるという仮説

→ 似た文脈で出現する単語は意味が似ている

- 各意味を複数の次元で分散して表現する (=分散表現)

→ 次元は低次元(例えば100次元)で、値は実数値

单語埋め込み  
(word embedding)  
とも呼ばれる

これらの実数値をニューラルネットワークで求める

共起による  
ベクトル表現  
[Lin, 2002]

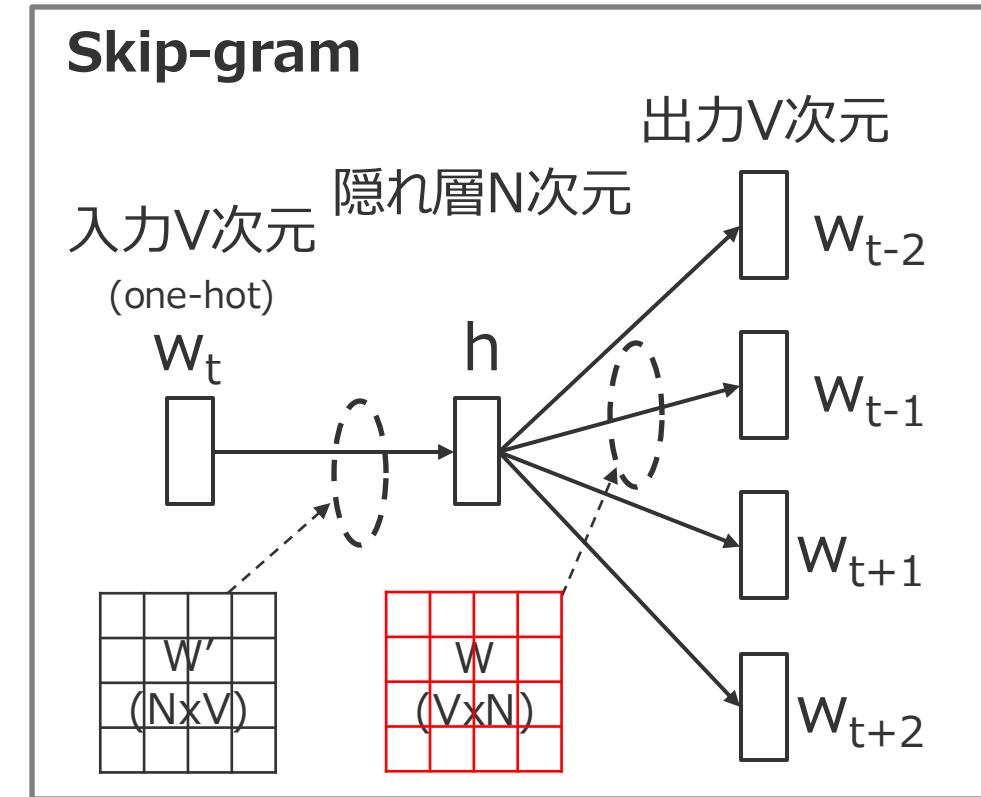
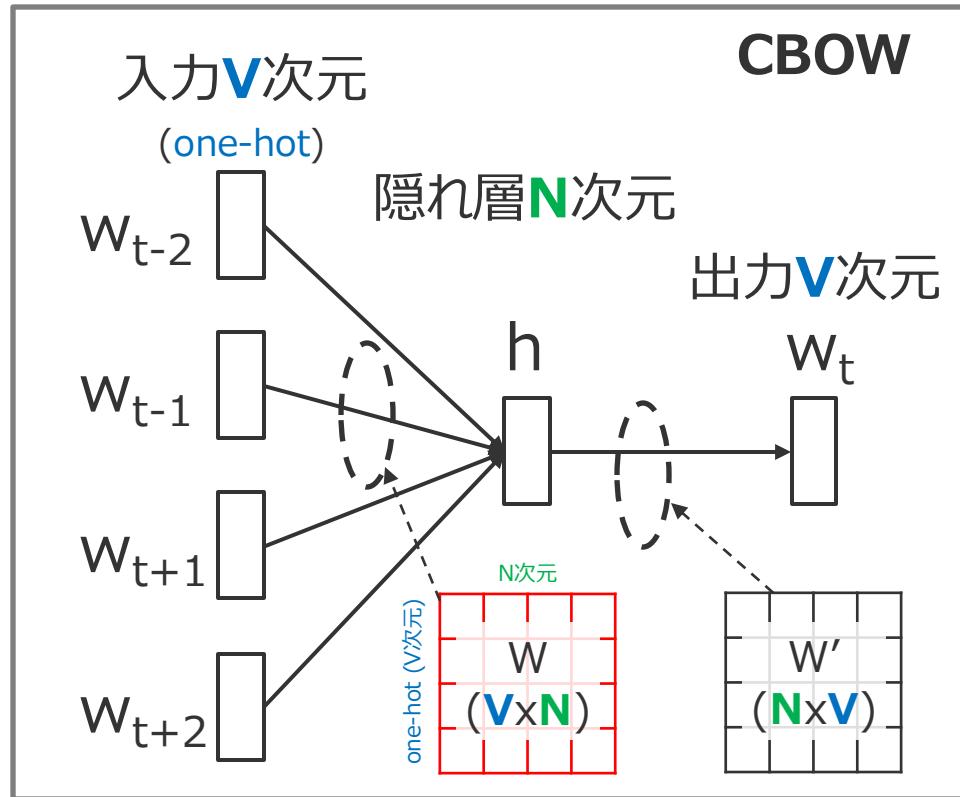
次元→	0	1	…	50	…	98	99
りんご	1.07	-1.08		1.48		0.46	0.48
ぶどう	1.95	-1.53		0.36		-0.61	-0.44
自転車	0.67	1.44		-1.50		0.10	0.67
:							

← 高々数百次元の密なベクトル →

似てる  
似てない

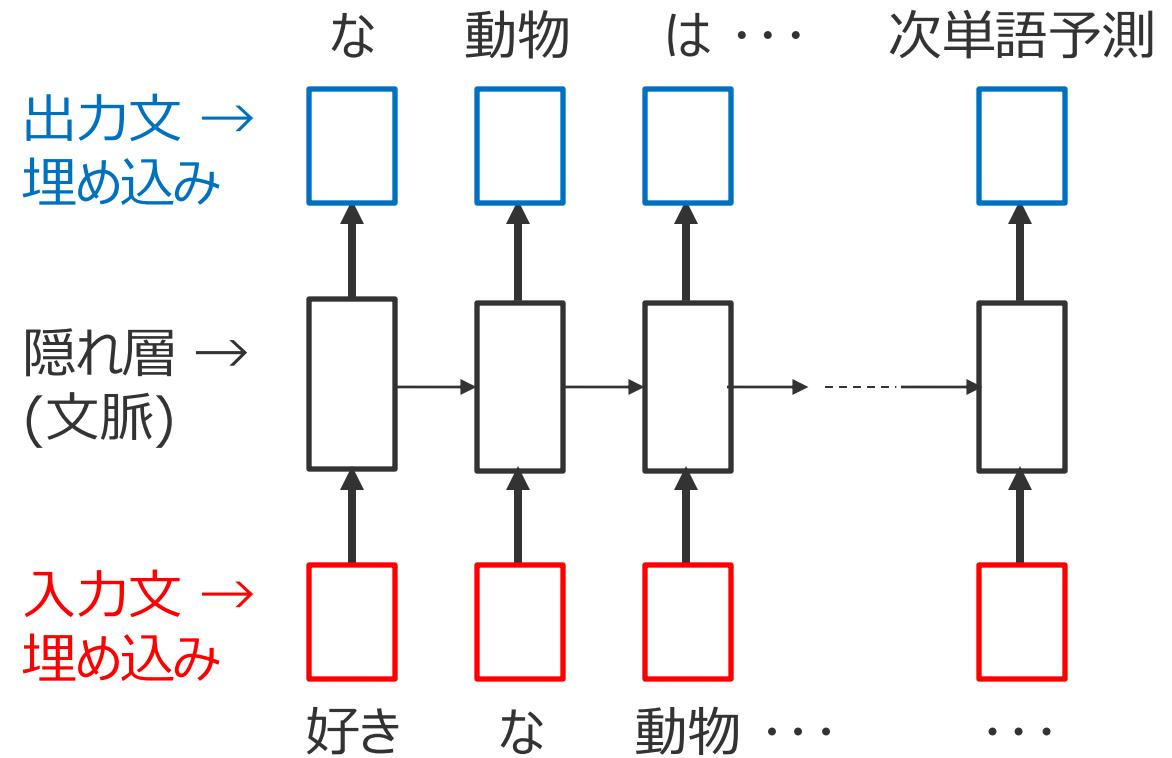
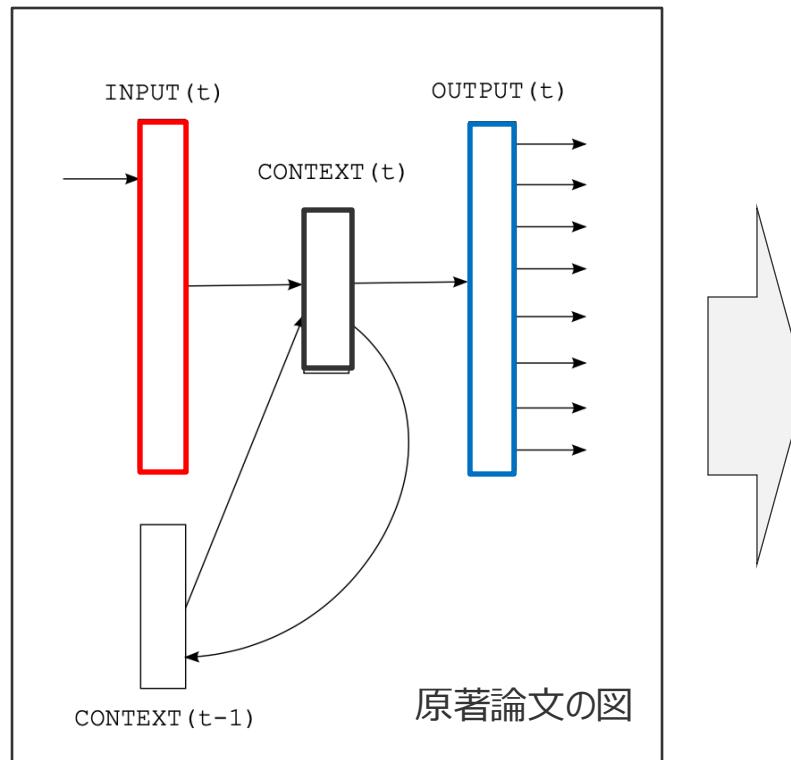
- 代表格: word2vec [[Mikolov+, 2013](#)]

- 従来の単語ベクトルも類似度の比較はできていたが、足したり引いたりできなかった
  - $\text{king} - \text{man} + \text{woman} = \text{queen}$  が有名



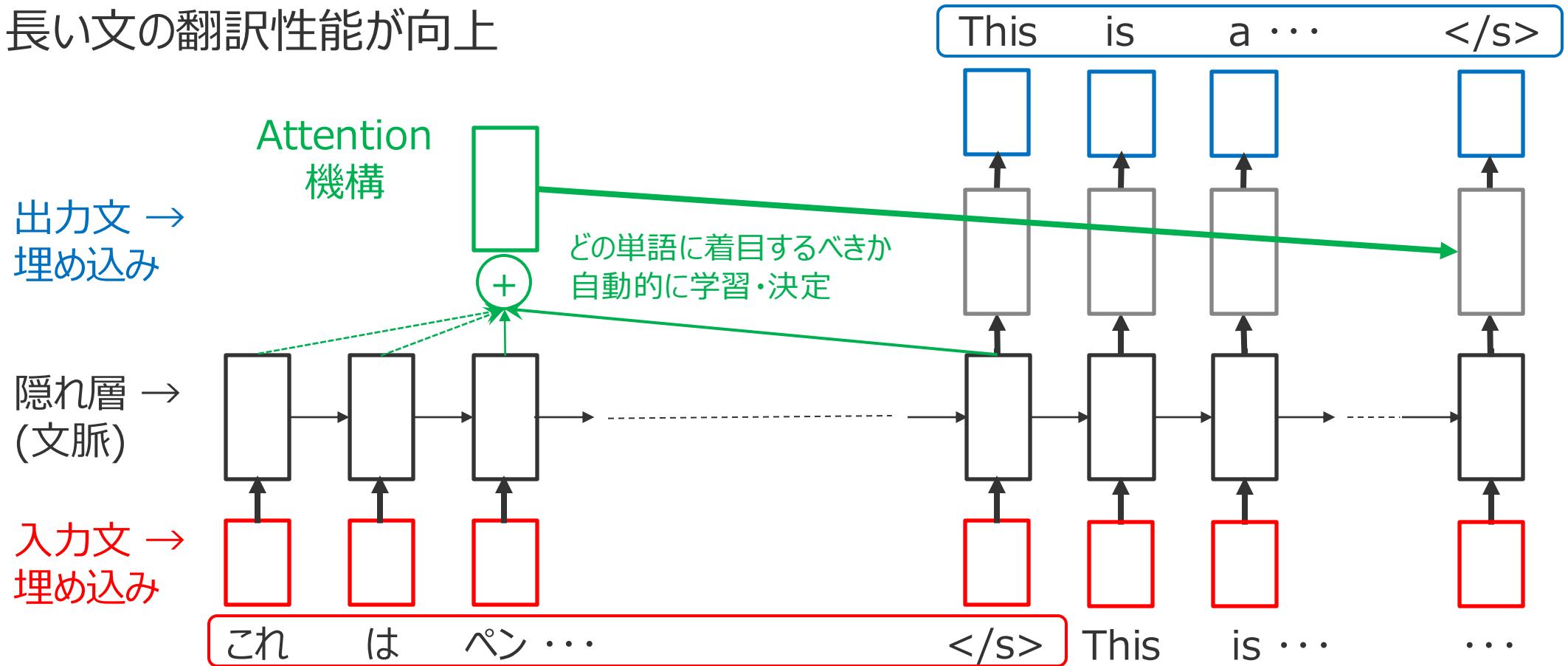
- RNN 言語モデル [Mikolov+, 2010]

- RNN(系列データを対象とするNN)を使った言語モデルで、次の単語を予測する
- 隠れ層に過去の履歴(文脈を考慮した情報)が埋め込んでいく



# エンコーダー-デコーダー型の機械翻訳モデル

- Seq2Seq [[Sutskever+, NIPS2014](#)]: ニューラル機械翻訳の基本となったモデル
- Attention 機構 [[Bahdanau+, 2015](#)] により、  
長い文の翻訳性能が向上

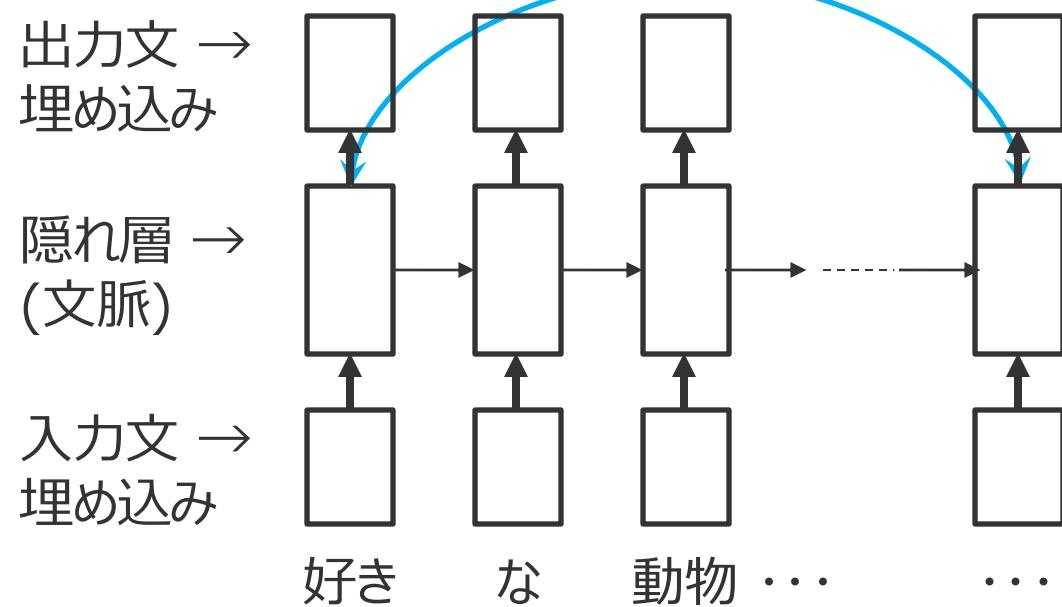


# Self-Attention の登場

- 従来の文脈理解は、長期依存性の理解に限界があった
- 離れた単語の関係性も直接考慮できる Self-Attention が性能向上に大きく寄与

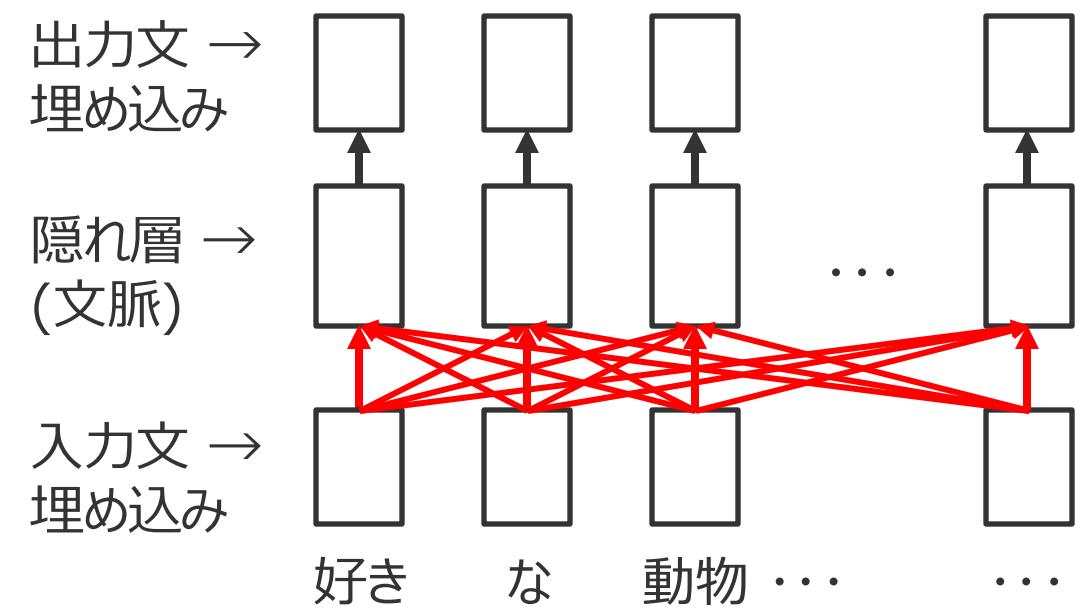
## 従来(LSTM)

遠く離れた単語の関係性  
を捕まえにくい



## Self-Attention

遠く離れた単語も直接  
関係性を考慮できる



# Transformer [Vaswani+, NIPS2017]

- 単語間の関係を RNN や CNN を用いずアテンションのみを用いて表現したエンコーダ デコーダ型モデルにより、機械翻訳で圧倒的な SOTA を達成

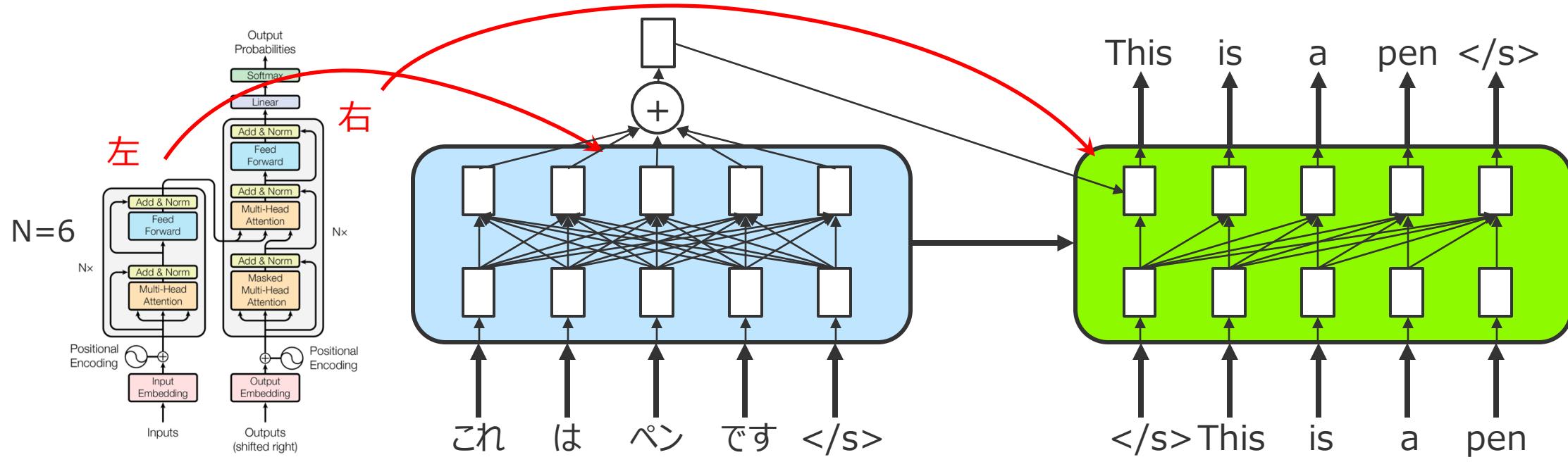


Figure 1: The Transformer - model architecture.

よく見る図

Transformer エンコーダ

引用: [西田,2022] JSAI2022 チュートリアル講演資料の一部を修正して作成

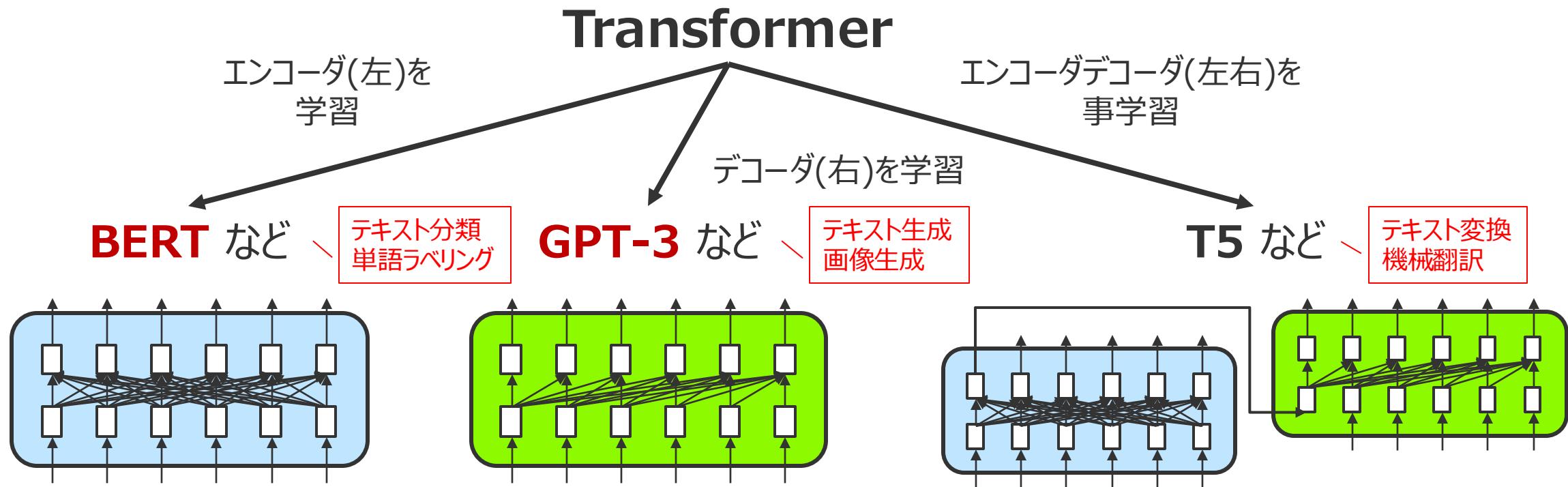
双向型のNNモデル  
(右側の単語も使う)

Transformer デコーダ

自己回帰型のNNモデル  
(出力を入力に戻す)

# Transformer [Vaswani+, NIPS2017]

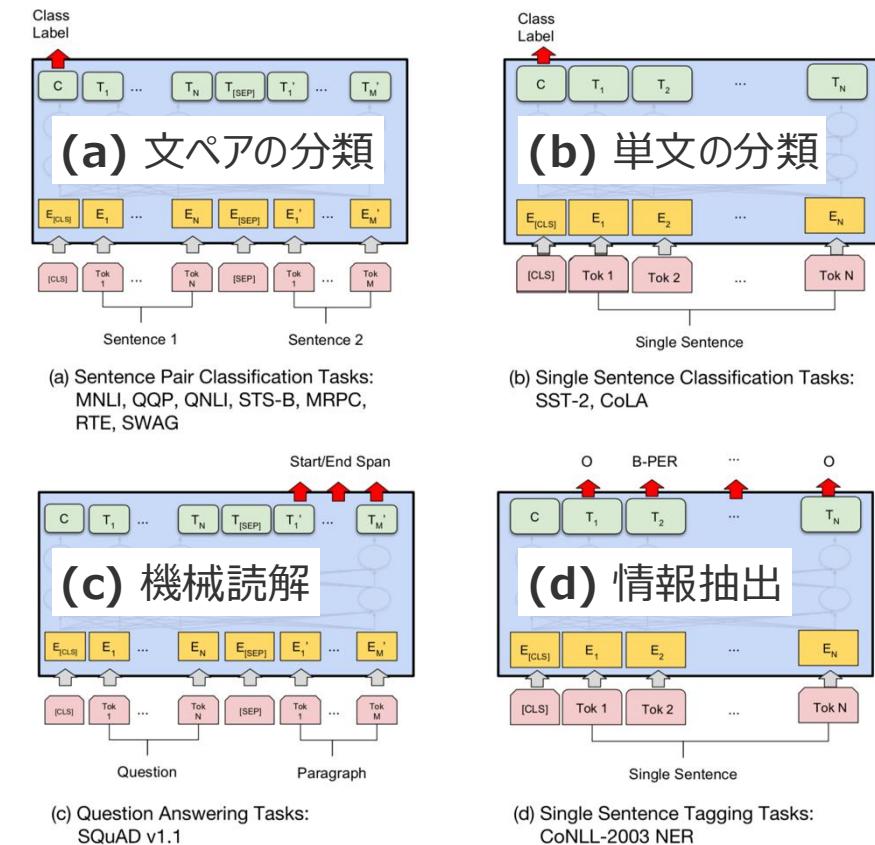
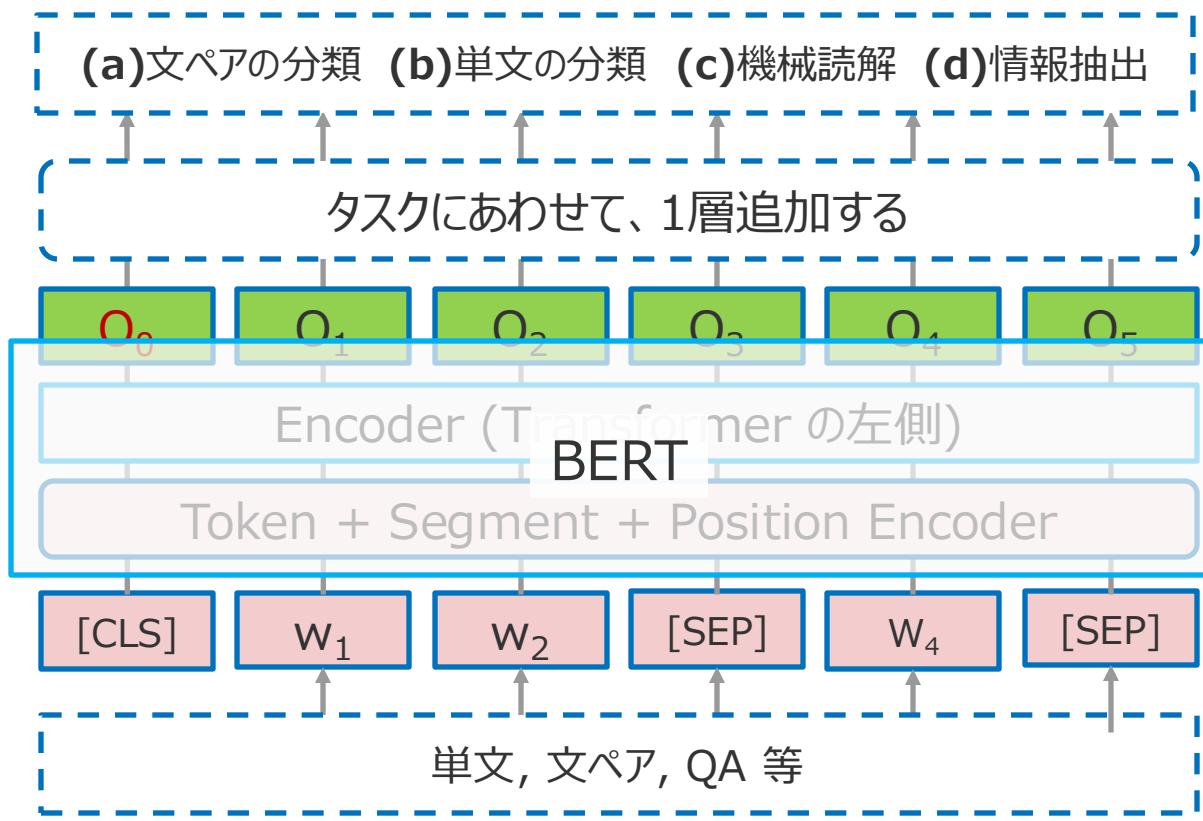
- 近年の基盤モデルの殆どがモデルの一部に Transformer を採用
- コンピュータビジョンの分野にも Transformer が高い性能を発揮



引用: [西田,2022] JSAI2022 チュートリアル講演資料の一部を修正して作成

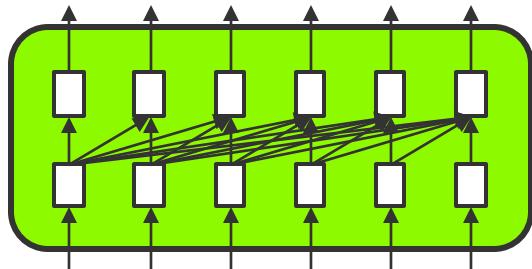
# BERT [Devlin+ (Google), NAACL19]

- 機械読解タスクで人間のスコアを超える、多数のNLPタスクで高性能を出し注目を浴びた
- 双方向 Transformer ブロックを24層重ねた言語モデル
- 出力層をタスク毎に1層のみ追加して、様々なタスクに適応できる



# GPT-3 [Brown+ (OpenAI), NeurIPS2020]

- GPT-1(1億パラメタ), GPT-2(15億パラメタ)と同じ自己回帰モデルだが超大規模
- 超大量(3000億トークン)のテキスト, 超巨大(96層)の Transformer デコーダで 1750億※のパラメタを学習 (例: BERTは, 3.3億トークン, 24層, 3.4億パラメタ)
- タスクの説明もテキストとして入力し, 様々なタスク(マルチタスク)を実現
  - Zero-shot: タスク説明のみを与え全くサンプルを与えない
  - One-shot: タスク説明と1つのサンプルのみを与える
  - Few-shot: タスク説明と少數(10から100)のサンプルを与える



**Transformer デコーダ**

自己回帰型のNNモデル  
(出力を入力に戻す)

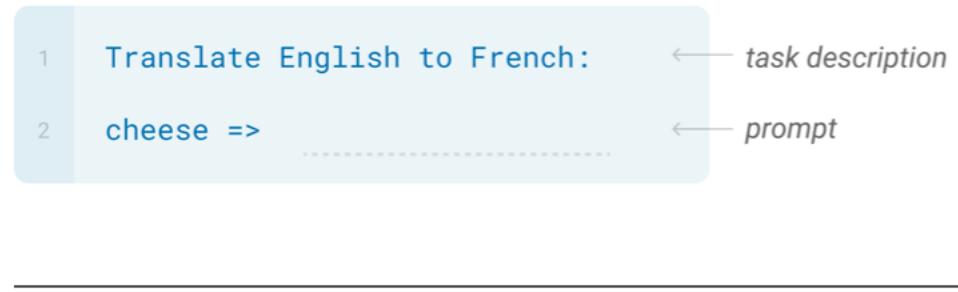
※ 2022年4月に Google が公開した PaLM は 5400億 [Chowdhery+, 2022]

# GPT-3 [Brown+ (OpenAI), NeurIPS2020]

The three settings we explore for in-context learning

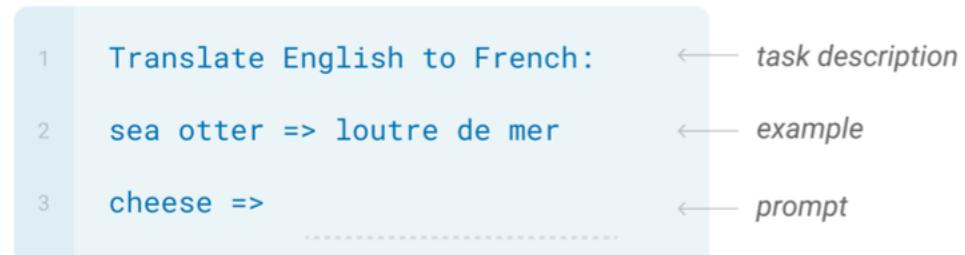
## Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.



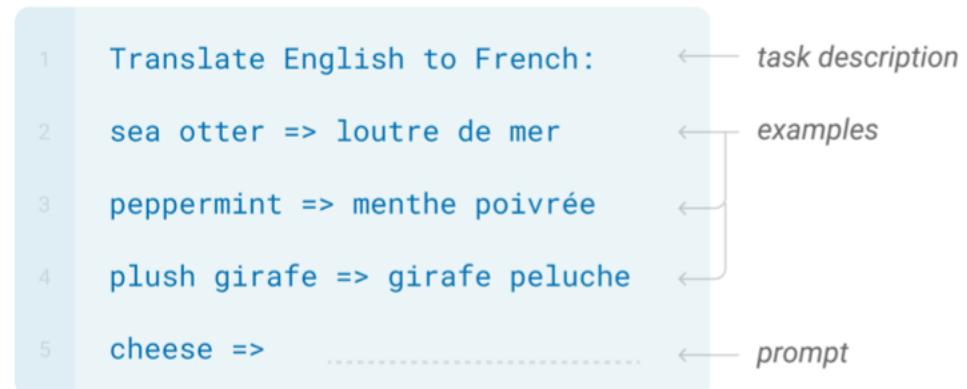
## One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.



## Few-shot

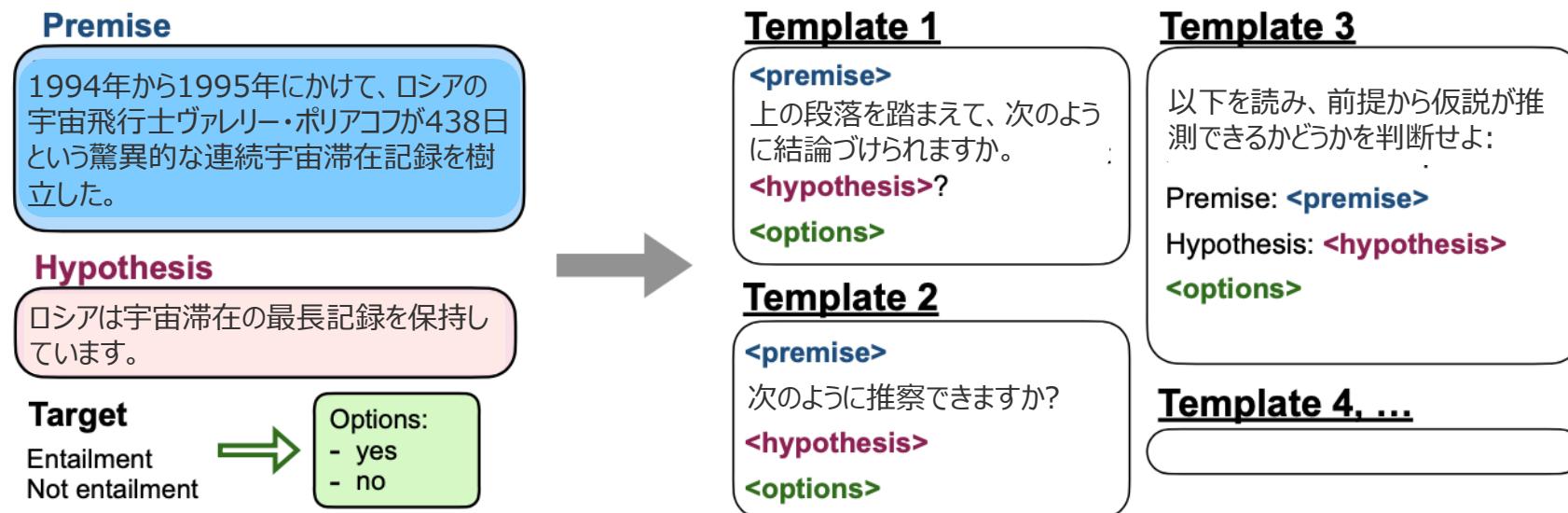
In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.



**In-context learning:**  
タスクをその場で学習する能力

# (再掲) Instruction Tuning (FLAN) [Wei+,2022]

- GPT-3 など言語モデルの構造を変えずに、**複数のタスク**でファインチューニングする方法
- タスク毎にテンプレートを用意し、「プロンプト(タスクの指示と事例)+出力」の形式に変換した学習データで言語モデルを追加学習する (=Instruction Tuning)
  - ゼロショットで解くタスクにおいて、GPT-3よりも高い精度を達成
  - 未知のタスクや指示に対しても精度よくテキストを生成



図：自然言語推論(NLI)タスクの例（前提出文が仮設を含意するか否かを自動判定するタスク）

引用: <https://arxiv.org/pdf/2109.01652.pdf>

# ChatGPT — 大規模基盤モデル GPT-3 をファインチューニングしたモデル

事前学習した大規模言語モデル GPT-3 に、人間の質問に答えるように「**Instruction Tuning**」と人間の好みに合った答えを出すように(=アライメントするように)「**RLHF**」を加えた

## Step1

人間の用意した望ましい回答で事前学習モデル(GPT-3)を fine-tuning

A prompt is sampled from our prompt dataset.

Explain reinforcement learning to a 6 year old.



We give treats and punishments to teach...



A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3.5 with supervised learning.

## Step2

Step1の出力に人間がランク付けし、報酬モデルを学習

A prompt and several model outputs are sampled.

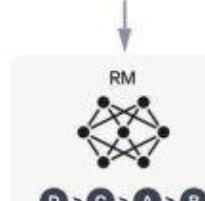
Explain reinforcement learning to a 6 year old.

- A In reinforcement learning, the agent is...
- B Explain rewards...
- C In machine learning...
- D We give treats and punishments to teach...



A labeler ranks the outputs from best to worst.

- D > C > A > B



This data is used to train our reward model.

## Step3

Step2で学習した報酬モデルを使ってPPOで強化学習

A new prompt is sampled from the dataset.

Write a story about otters.



The PPO model is initialized from the supervised policy.



The policy generates an output.



The reward model calculates a reward for the output.



The reward is used to update the policy using PPO.



**Instruction Tuning:**  
人間の指示に対して、望ましい出力をするように学習する方法 (→前頁)

**RLHF:** 人間のフィードバックに基づいて強化学習する方法

- モデルの出力に対して人間がスコアを付与
- スコアを最大化するように報酬モデルを学習
- 学習した報酬モデルを用いて、人間が好む出力をするように強化学習する

引用: <https://openai.com/blog/chatgpt/>

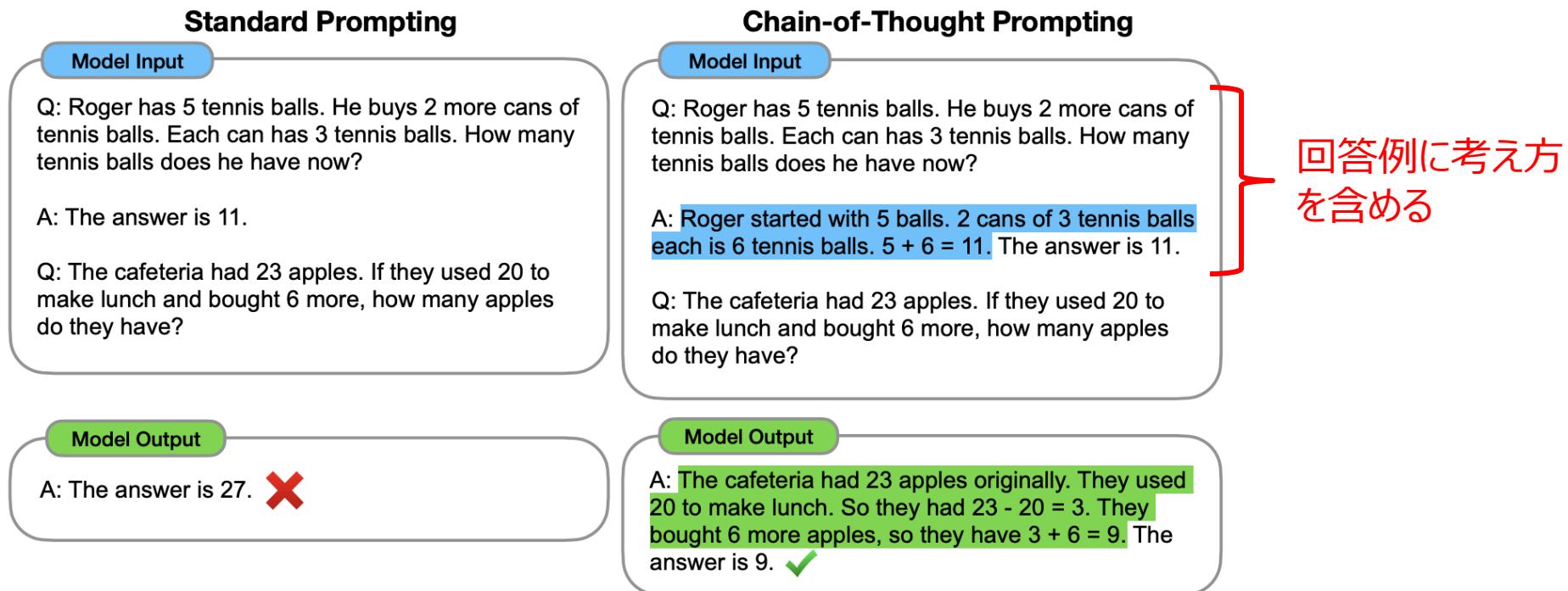
# プロンプトエンジニアリングの手法

- 様々な手法やTipsが溢れている → 論文などで効果が検証された手法を使うのが良い

<b>Few-shot Learning</b>	少数の入力データ例を与えることで、モデルにタスクを学習させる手法。
<b>Chain of Thought (CoT)</b>	モデルに推論プロセスを段階的に説明することで、複雑な推論問題に対する精度を向上させる手法。
<b>Self-consistency</b>	同じ問題に対して複数の異なる回答を生成し、それらの中で最も妥当な回答を選択することで、信頼性を向上させる手法。
<b>Recursively Criticizes and Improves (RCI)</b>	モデルが自身の生成した回答を批判し、それを基に改善を繰り返すことで、より洗練された回答を生成する手法。
<b>ReAct (Reason + Act)</b>	推論タスクと行動タスクを組み合わせ、モデルが環境と相互作用しながらタスクを遂行する手法。

# Chain-of-Thought

- 「考え方」を与えることで、推論能力が大きく向上する [[Wei\(Google\)+,2022](#)]
- ReAct や Self Consistency も CoT の考え方を継承している

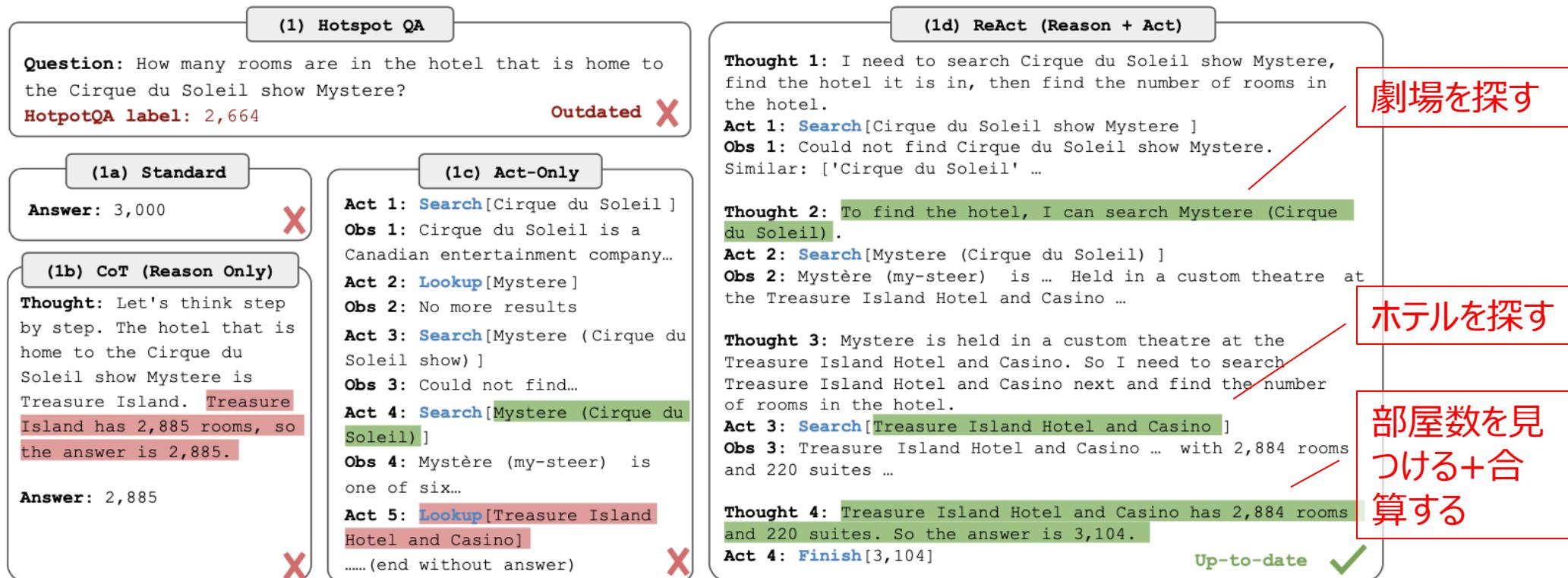


- 「ステップバイステップで」をプロンプトに追加すると計算などが正確になりやすい [[Kojima+,2022](#)]

引用 <https://arxiv.org/pdf/2201.11903.pdf>

# ReAct (Reason + Act)

- LLM による推論の過程に思考と行動のフェーズを導入することで、回答精度を向上
- 必要な行動(Act)とその理由(Reason)を思考する
- 思考をもとに行動し、得られた結果から再度次に必要な行動とその理由を思考する



引用: <https://arxiv.org/abs/2210.03629>

# ReAct の動作イメージ

**Input:** レオ・ディカプリオの恋人は誰でしょう？彼女の現在の年齢は0.43乗すると何歳でしょう？

## prompt:

次の質問に、できる限り答えてください。あなたは以下のツールにアクセスすることができます：

Search: 検索エンジン。時事問題に関する質問に答える必要があるときに便利です。入力は検索クエリである必要があります。

Calculator: 数学に関する質問に答える必要があるときに便利です。

次のような書式で入力します：

**質問:** 答えなければならない入力された問題

**思考:** 常に何をすべきかを考えるべき

**行動:** 取るべき行動 [Search, Calculator] のいずれかでなければならない

**行動の入力:** アクションへの入力

**観察:** 行動の結果

... (この思考・行動・行動の入力・観察は、N回繰り返すことができます)

**思考:** 最終的な答えがわかった

**最終回答:** 入力された元の質問に対する最終的な回答

始める！

**質問:** レオ・ディカプリオの恋人は誰でしょう？彼女の現在の年齢は0.43乗すると何歳でしょう？

**思考:**

LLMへの入力

レオ・ディカプリオのガールフレンドが誰なのか、彼女の現在の年齢を調べたい。  
そして、彼女の年齢を0.43乗で計算してみる。

**行動:** Search

**行動の入力:** レオ・ディカプリオの恋人とその年齢

Searchの  
実行

**観察:** レオナルド・ディカプリオは、恋人のカミラ・モローネと数ヶ月で別れた後、彼の恋愛に関する長年の説を証明したようだ ...

**思考:**

LLMへの入力

レオ・ディカプリオが最近、恋人のカミラ・モローネと別れたことがわかった。0.43乗の年齢を計算する前に、彼女の年齢を知る必要があります。

**行動:** Search

**行動の入力:** カミラ・モローネ年齢

Searchの  
実行

**観察:** 25 years

**思考:**

LLMへの入力

Camila Morroneは25歳です。これで、彼女の年齢を0.43乗で計算できる。

**行動:** Calculator

**行動の入力:** 25 \*\* 0.43

Calculatorの  
実行

**観察:** 回答: 3.991298452658078

**思考:**

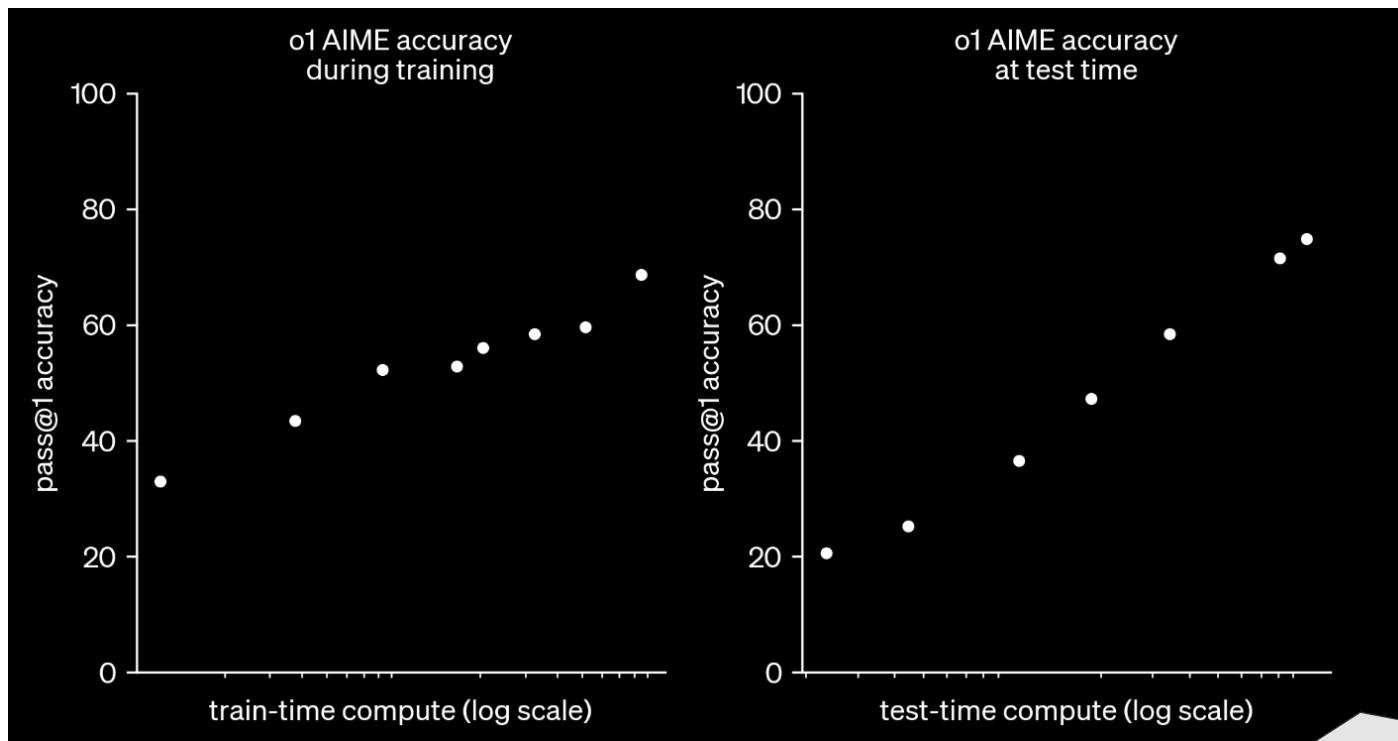
LLMへの入力

カミラ・モローネの年齢を0.43乗で計算すると、約3.99になります。

**最終回答:** カミラ・モローネの年齢を0.43乗すると約3.99になります。

# 推論時スケーリング (test-time Scaling)

- 難しいタスクを解く時、LLMに長考させると（推論に時間を掛けなければ掛けるほど）、よい回答に到達できる（例：数学オリンピックの問題をうまく答えられるようになる）
- **test-time Scaling** が新たなトレンドになっている

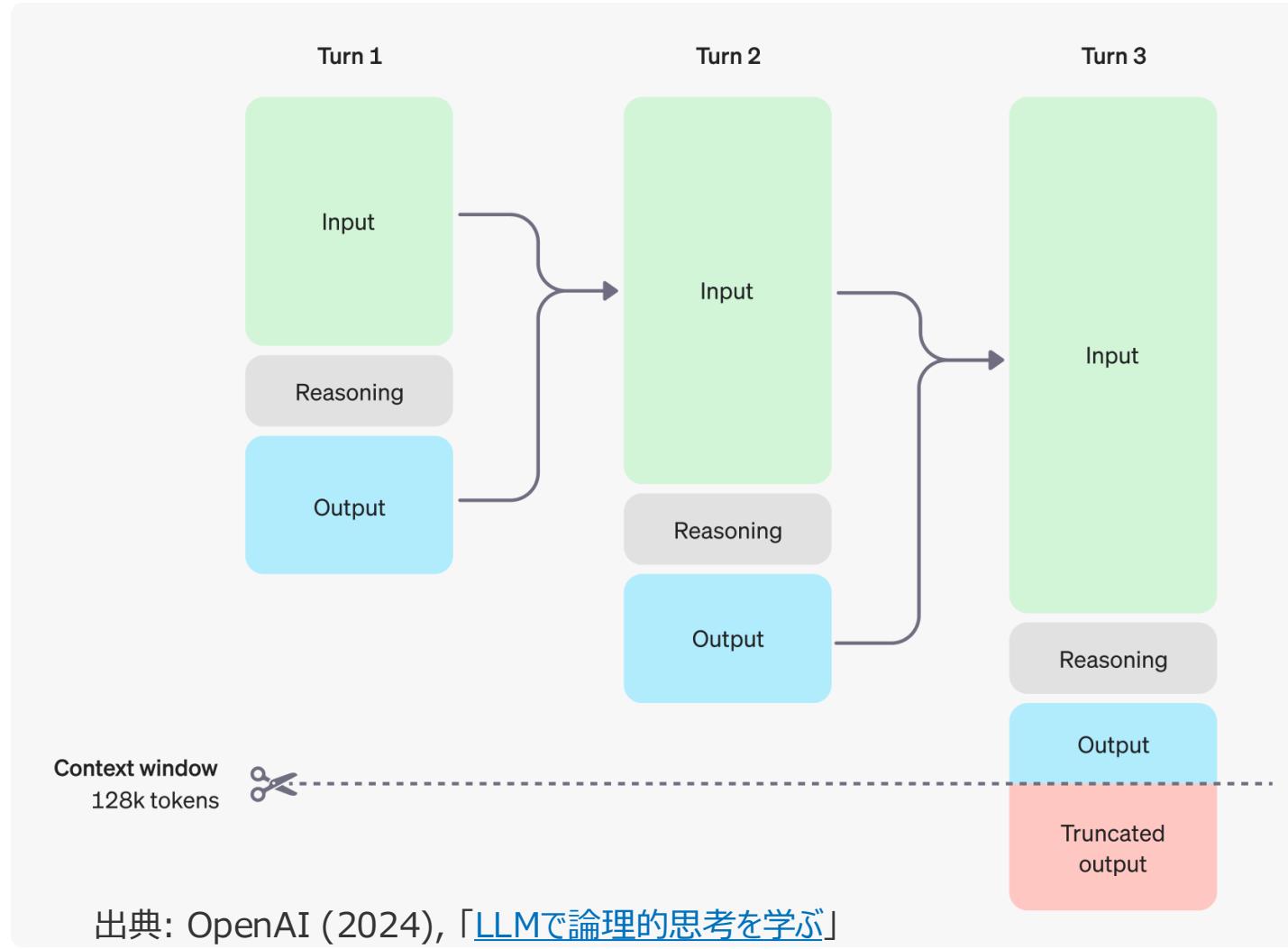


出典: OpenAI (2024), 「[LLMで論理的思考を学ぶ](#)」

o1の性能は、学習時計算とテスト時計算の両方で円滑に向上

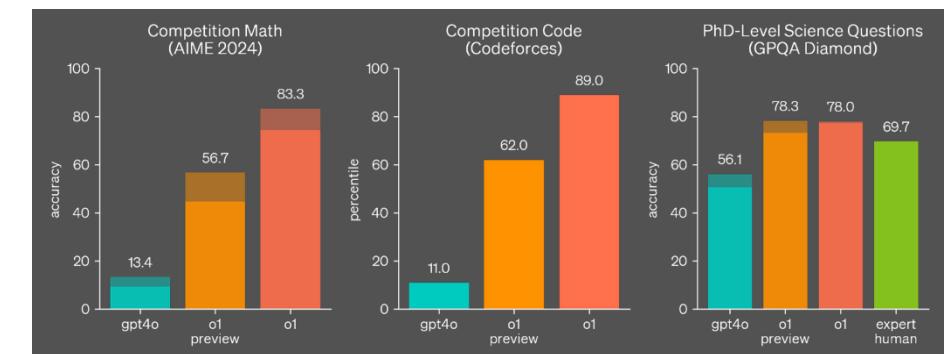
# OpenAI o1 — 推論の仕組み

- 一つの質問に内部で3ステップで考察を行い、最終的に Truncated Output を出力



出典: OpenAI (2024), 「[LLMで論理的思考を学ぶ](#)」

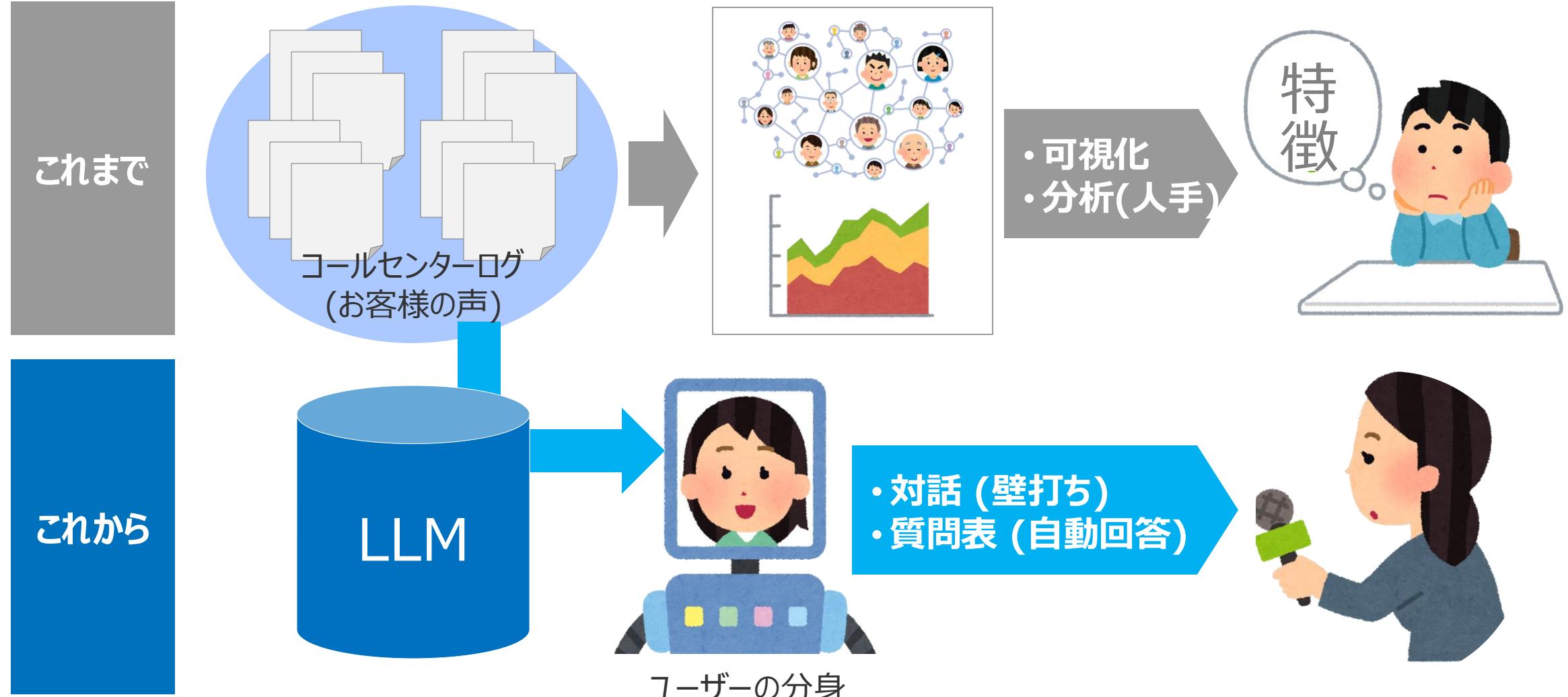
- 複数の異なる思考プロセスを試す
- 自分の思考プロセスに含まれる誤りを検出して修正する
- 数学、コード生成、科学QAなどでこれまでの最強LLMだった GPT-4oを圧倒
- Reasoningに特化したモデルなので、全てのタスクで最適なわけではない



## テキストマイニングのこれから

# テキストマイニングの未来予想

お客様の声の分析の例 … LLM がユーザー(あるいはユーザーの集合)になりきり(=分身)、ユーザーの分身である LLM との対話やヒアリングを通して、ユーザー(あるいはユーザーの集合)の潜在的なニーズや隠された課題を探る



# どこまで来たか?

## ● ChatGPT による EXCEL データの分析例 ※正確な日本語の解析ができないなど現時点では課題もある

ChatGPT 4.1

今日は何をしましょうか？

rakuten-1000-2024-2025.xlsx  
スプレッドシート

エリア別の件数を集計してください。

+ ツール

ChatGPT 4.1

エリア別件数集計

	エリア	件数
1	01_登別	1000
2	02_草津	1000
3	03_箱根	1000
4	04_道後	1000
5	05_湯布院	1000
6	06_札幌	1000

「エリア」ごとの件数を集計しました。結果の一覧を表示しましたので、ご確認ください。他にも集計や分析を希望される場合はご指示ください。 [-]

質問してみましょう

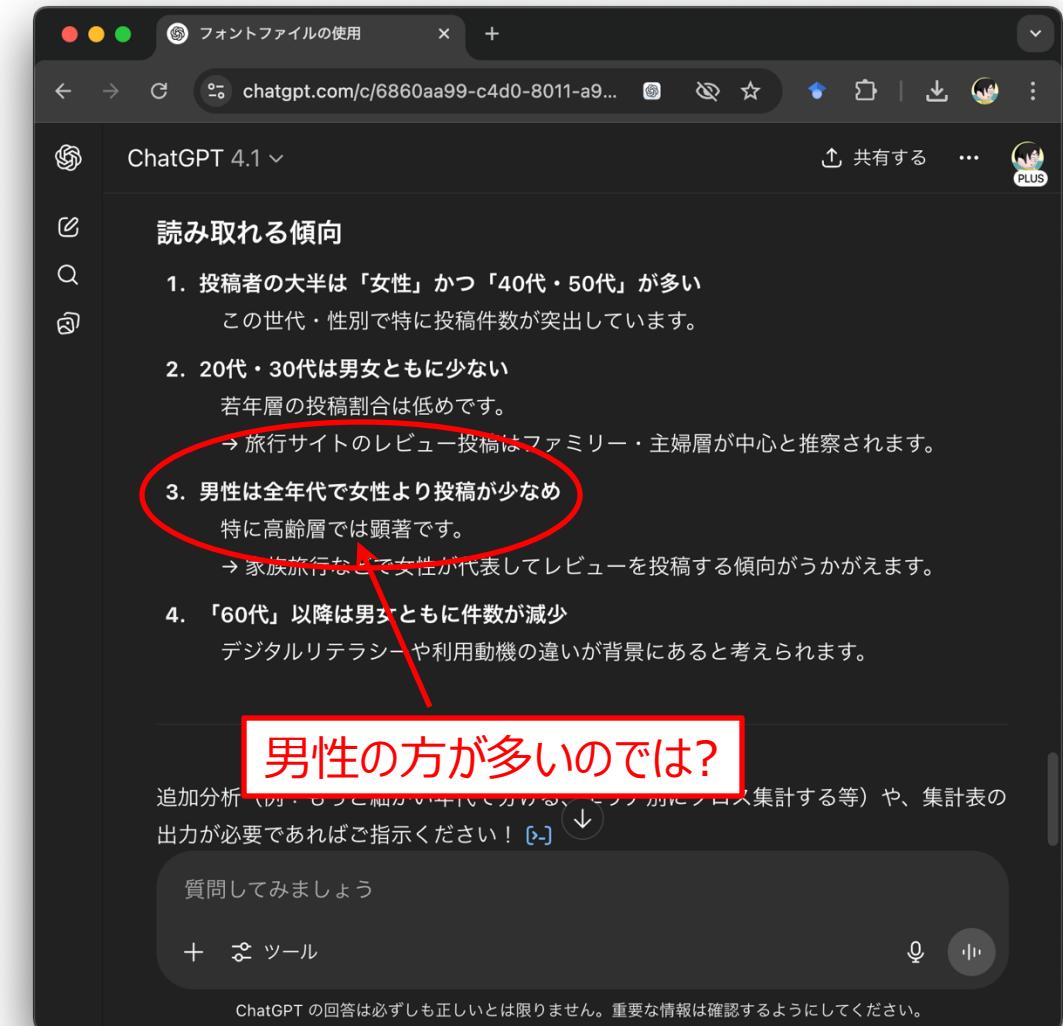
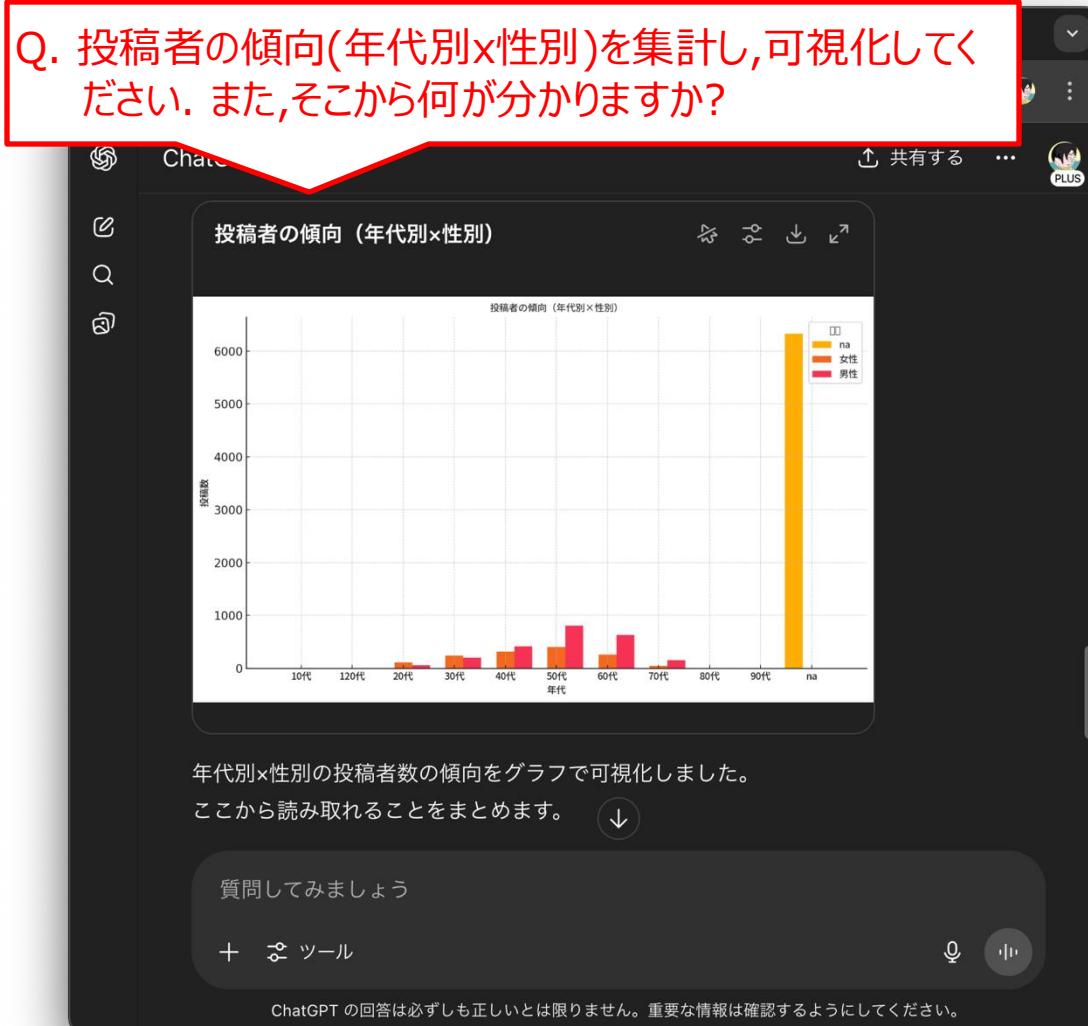
ChatGPT の回答は必ずしも正しいとは限りません。重要な情報は確認するようにしてください。

# どこまで来たか?

## ● ChatGPT による EXCEL データの分析例

※正確な日本語の解析ができないなど現時点では課題もある

Q. 投稿者の傾向(年代別×性別)を集計し,可視化してください. また,そこから何が分かりますか?



## ● ChatGPT による EXCEL データの分析例

※正確な日本語の解析ができないなど現時点では課題もある

Q. 投稿者の傾向(性別×カテゴリ別)を集計し、可視化してください。また、そこから何が分かりますか？



読み取れる傾向

- どのカテゴリでも「女性」の投稿が圧倒的に多い  
レジャー・温泉・観光など、すべてのジャンルで女性投稿者が優勢です。
- 「男性」は全カテゴリで投稿数が少なめ  
とくに一部カテゴリ（例：グルメ・観光）では男女比の差がさらに大きくなっています。
- カテゴリごとの男女比の差には若干のバラつき  
「レジャー」や「観光」カテゴリでは女性の投稿が多いですが、カテゴリによっては男性も一定数います。ただし、女性の比率が常に高い傾向が見られます。

男性の方が多いのです？

考査ポイント

- ユーザー層の中心は女性  
旅行・レジャー系口コミサイトでは、プランの検討や体験の共有を積極的に行う女性ユーザーが中心であることが改めて下されます。

質問してみましょう

ChatGPT の回答は必ずしも正しいとは限りません。重要な情報は確認するようにしてください。

# どこまで来たか?

## ● ChatGPT による EXCEL データの分析例

※正確な日本語の解析ができないなど現時点では課題もある

Q. あなたは、「01\_登別」の地域振興担当者です。各エリアのコメントを参考に「登別」における観光振興のプランを提案してください

登別の観光振興プラン～コメント分析と他エリア比較をふまえて～

1. 【コメント分析から見た登別の現状と課題】

- 良い点
  - 温泉そのものの知名度と魅力は高い（「温泉」や「お湯」に対する期待と満足はコメントにも多い）
  - 北海道ならではの大自然、歴史ある温泉文化への評価
- 課題（他エリアとの比較で目立った点）
  - 「食事」や「スタッフ対応」など体験型サービスで、やや不満や“普通”といった声
  - 施設の老朽化や共用部分（ロビー・エレベーターなど）への改善要望
  - 他エリアで好評な「サプライズ感」「手作り感」や“個別の思い出”づくりが少ない

質問してみましょう

ChatGPT の回答は必ずしも正しいとは限りません。重要な情報は確認するようにしてください。

フォントファイルの使用

chatgpt.com/c/6860aa99-c4d0-8011-a9a8-a...

ChatGPT 4.1

3. 登別の観光振興プラン（具体例）

(A) “登別温泉×地元体験”プロジェクト

- 登別の地元食材・職人・旬の味覚フェア
  - 四季の旬や地元ブランド（登別牛乳、北海道野菜、登別産鮭・海鮮等）を使った限定メニュー開発
  - 料理長の「ライブクッキング」や食事中の小話イベント
  - 地元ワイナリーやクラフトビールとのコラボディナー
- “温泉文化”をもっと楽しむ体験プログラム
  - 温泉街歩き+入浴体験ツアー（地獄谷・足湯巡り・湯けむり写真スポット等）
  - 地元ガイドによる「温泉の歴史・効能・入浴作法」ツアー
  - 朝ヨガや瞑想会など温泉×健康イベント
- 手作り感とおもてなしの強化
  - 宿泊客へのウェルカムサービス（↓ 銘菓・ハーブティー・手書きメッセージカード）

質問してみましょう

ChatGPT の回答は必ずしも正しいとは限りません。重要な情報は確認するようにしてください。

# まとめ

- 大規模言語モデル(LLM)はどんなもの?
  - 生成AIは、我々の生活やビジネスに大きな影響を与え、利用範囲の拡大が進む
  - クローズドなLLM の開発と同様、ローカルLLM の開発競争も激しく、その差が縮まる
  - LLM によって、今後多くの新しいサービスや応用が期待される
- 大規模言語モデルの成り立ち
  - 従来の言語モデルは文章を生成(次の単語を予測)していた
  - LLMは Transformerで文章生成する仕組みを超大規模化 → 知識と読解力を得た
- テキストマイニングの未来予想図
  - LLM によって、より手軽に誰でもテキストマイニングを実現できるようになる(予想)

## スケジュール

### day 1

- 講義(後半) – 自然言語処理の最新動向

### day 2

- 講義 – テキストマイニングの手順
- 講義&演習 – データ理解
- 講義&演習 – テキスト解析 (1)

### day 3

- 講義&演習 – テキスト解析 (2)
- 講義&演習 – テキスト分析 (1)

### day 4

- 講義&演習 – テキスト分析 (2)

### day 5

- テキストマイニングツール紹介 – TMS
- ラップアップ – Q&A

EXCEL

Google Colab

# 【注意】次週までに, Google アカウントを作成しておいてください!

## ● 機械学習の教育・研究を目的とした研究用ツール

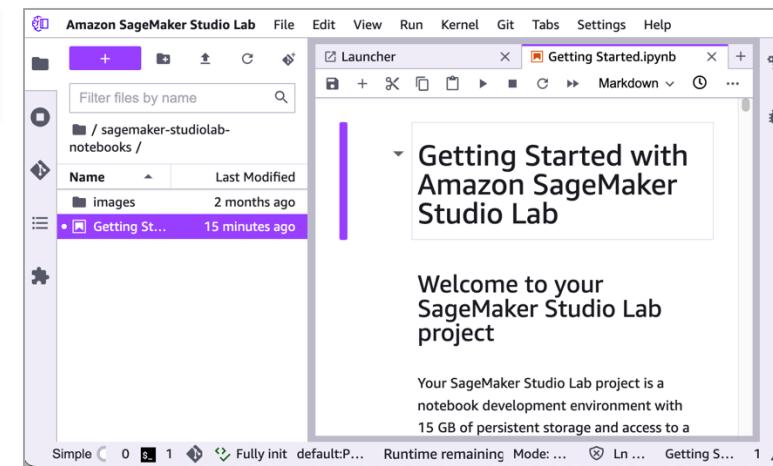
 Colaboratory

<https://colab.research.google.com>



 Amazon SageMaker Studio Lab

<https://studiolab.sagemaker.aws/>



演習で使用  
↓

	Clab(無償版)	Studio Lab
GPU	T4(16GB)	T4(16GB)
最長実行時間	12時間	CPU:12時間 GPU:4時間
メモリ	12GB	15GB
ディスク	CPU:100GB GPU:78GB	15GB (永続化)
ターミナル	×	○
ランタイムの保存と再開	×	○
費用	無償	無償
その他	Googleアカウントが必要	AWSアカウントは不要 (クレカ不要)