

[2024年度]

マテリアルズ・インフォマティクスの基礎と応用

計算物質科学におけるLLMの活用～基礎編

LLMの基礎と環境構築

2024/12/13(金) 10:30-12:00

東京大学 物性研究所 吉見 一慶



東京大学 物性研究所
THE INSTITUTE FOR SOLID STATE PHYSICS
THE UNIVERSITY OF TOKYO

PASUMS

Project for advancement of
software usability in materials science

自己紹介

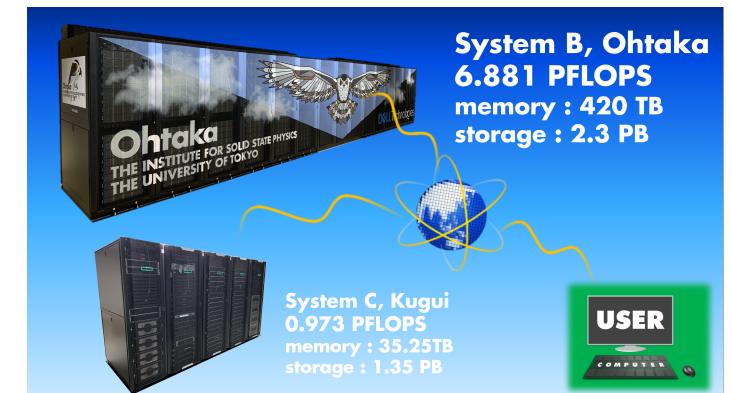
年月	経歴	
2009/03	東京大学理学系研究科 物理学専攻 博士課程修了 (東京大学物性研究所 加藤研究室)	 東京大学 物性研究所 THE INSTITUTE FOR SOLID STATE PHYSICS THE UNIVERSITY OF TOKYO
2009/04- 2010/03	東京医科歯科大学 教養部 特任助教 (越野研究室)	 国立大学法人 東京医科歯科大学 TOKYO MEDICAL AND DENTAL UNIVERSITY
2010/04- 2012/03	東京大学理学系研究科 物理学専攻 特任研究員	
2012/04- 2015/03	株式会社 構造計画研究所 入社	 構造計画研究所 KOZO KEIKAKU ENGINEERING Inc.
2015/04-	東京大学物性研究所 特任研究員 (PI) ソフトウェア開発・高度化プロジェクト 2016/01-2020/03の期間は 計算物質科学人材育成コンソーシアム (PCoMS) 次世代研究員を兼任	 東京大学 物性研究所 THE INSTITUTE FOR SOLID STATE PHYSICS THE UNIVERSITY OF TOKYO

ソフトウェア開発・高度化プロジェクトの紹介 (1)

プロジェクト開始の背景-1

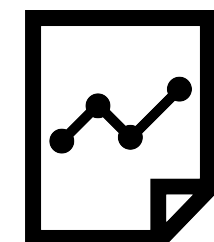
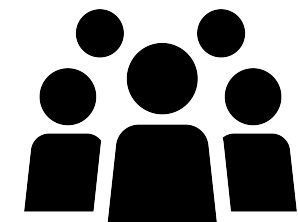
- ・ 研究・プロジェクトで開発されたソフトウェア

- ・ スーパーコンピュータの性能を活用した最先端のソフトウェア
- ・ 最新の機能を実装した高度なソフトウェア



- ・ ソフトウェアを一般公開することの利点

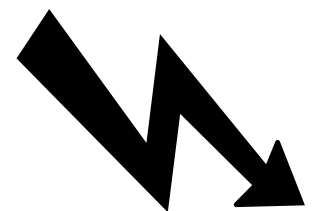
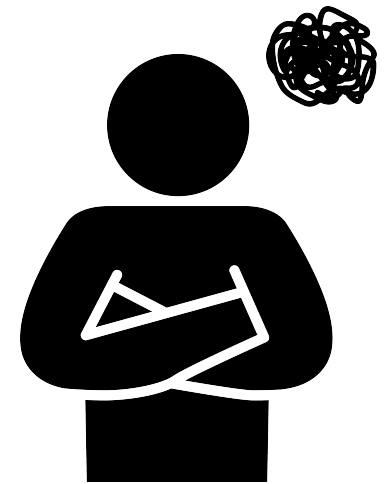
- ・ コミュニティコードとして、分野全体での利用・発展が可能
- ・ 開発者がソフトウェア論文を執筆することで、引用を通じて業績として認識される



ソフトウェア開発・高度化プロジェクトの紹介 (2)

プロジェクト開始の背景-2

- ・ その一方で
 - ・ 教育や研究といった他の業務に多くの時間を費やさざるを得ない
 - 使いやすさの向上、テスト、マニュアル、ウェブページ整備の優先度は低くなる
- ・ 結果として
 - ・ 優れたソフトウェアがあっても普及しにくい
 - ・ 他の研究者が利用・改良する機会が失われる



ソフトウェア開発・高度化プロジェクト

よりユーザーフレンドリーな
スーパーコンピュータの提供を目指して

毎年12月に公募 (2015年～)

- 1,2件の課題を選定
 - OSSとして年度末に公開
- 年間スケジュールイメージ



PASUMS

Project for advancement of
software usability in materials science

2023年度HPCIソフトウェア賞
普及部門賞の最優秀賞受賞！

主要メンバー (2024/10/11現在)



吉見 一慶

(2015/4～)



本山 裕一

(2018/4～)



青山 龍美

(2022/5～)

April



Kick-off meeting

May ~ Feb.



GitHub

Software development



Regular meeting/month

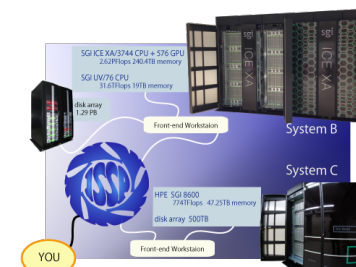
March ~



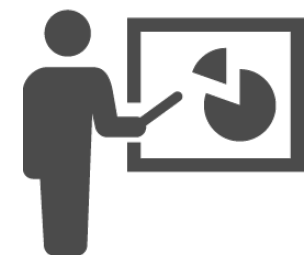
Website

MateriApps

Portal site



Preinstall



Lecture

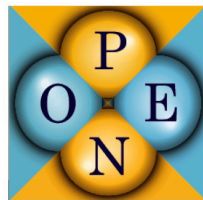
ソフトウェア開発・高度化プロジェクトの紹介 (4)

高度化したソフトウェアの一覧

First principles calculations and related applications

abICS

abICS



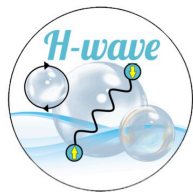
OpenMX



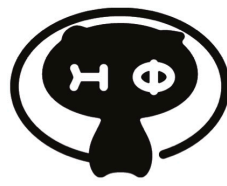
RESPACK

ESM-RISM

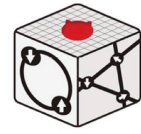
Quantum lattice model solvers



H-wave



HΦ



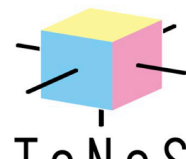
mVMC



DSQSS



DCore



TeNeS

2023年度HPCIソフトウェア賞 (開発部門賞)
HΦ: 優秀賞、mVMC: 最優秀賞

Mathematical libraries and others



PHYSBO



Kw



2dmat



MateriApps

Google Scholarでの引用数 (2024/10/8時点)

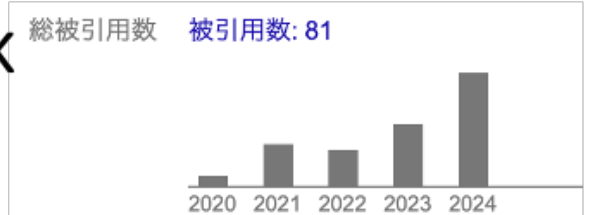
mVMC



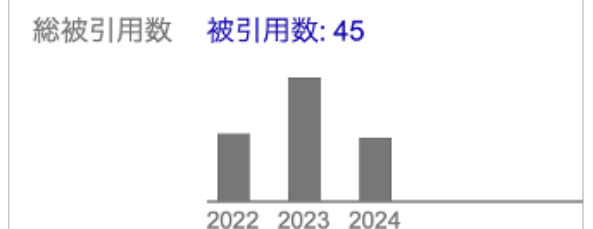
HΦ



RESPACK



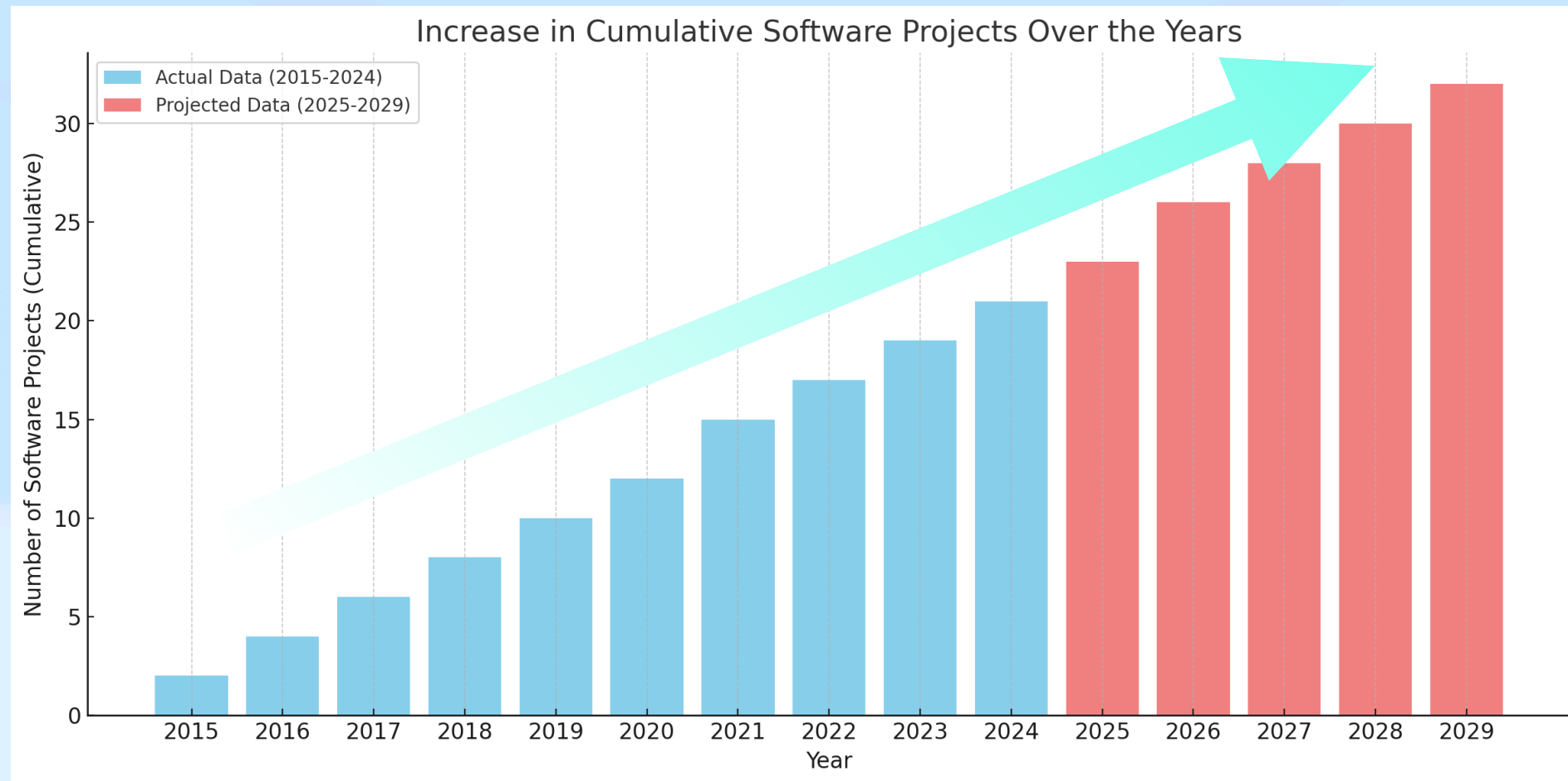
PHYSBO



開発したソフトウェアに関する論文も執筆・出版 URL: <http://www.pasums.issp.u-tokyo.ac.jp>
物性分野の研究者と幅広い連携が可能

生成AIを用いたソフトウェア開発の取り組み背景

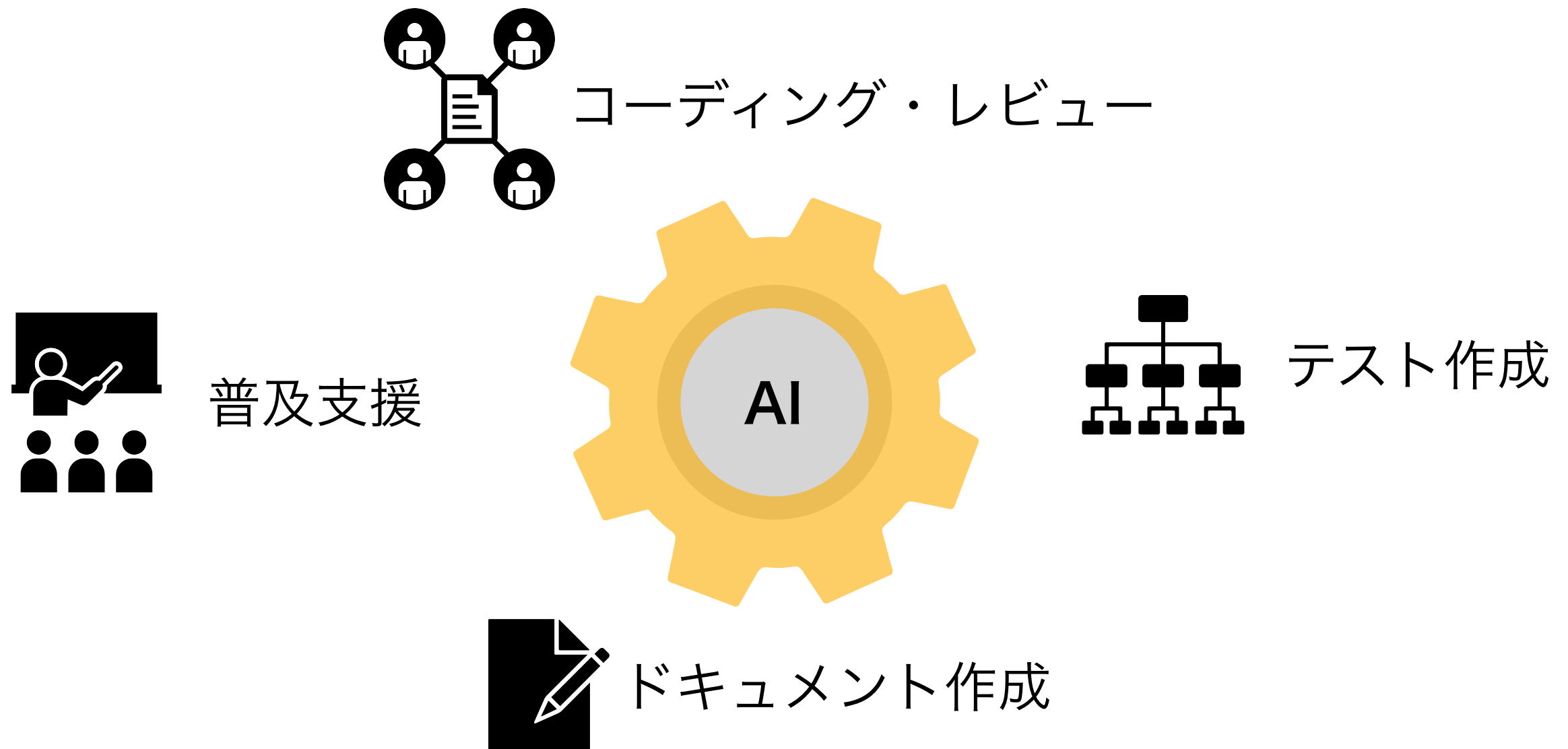
保守するソフトウェア・機能の増加



- ・ コミュニティコードとして管理を目指すが…
 - ・ 個々のソフトウェアのメンテナンスと質の管理が重要
 - ・ 開発プロセスの効率化が求められる

生成AIを用いたソフトウェア開発の取り組み背景

生成AIによる業務改善の可能性



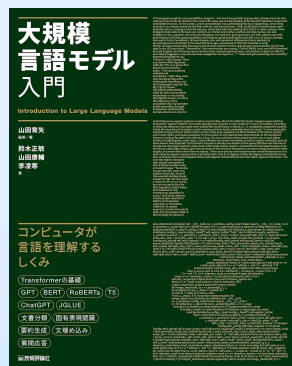
- ・ 開発業務の自動化・普及活動支援による時間短縮
- ・ ソフトウェアの品質向上

本日本話しする内容

- ・ LLMの基礎知識 ～ 言語モデルの仕組みとその進化
- ・ LLMツール(ChatGPTとGitHub Copilot)の概要
- ・ LLMツール(ChatGPTとGitHub Copilot)の環境構築
- ・ まとめ

本日本話しする内容

- LLMの基礎知識 ～ 言語モデルの仕組みとその進化
- LLMツール(ChatGPTとGitHub Copilot)の概要
- LLMツール(ChatGPTとGitHub Copilot)の環境構築
- まとめ



参考文献) 「大規模言語モデル入門」 1～3章、
山田 育矢 (監修), 鈴木 正敏 (著), 山田 康輔 (著), 李 凌寒 (著)、
技術評論社(2023)



参考文献) 「ソフトウェア開発にChatGPTは使えるのか？」 1章、
小野 哲、技術評論社(2023)

コンピュータによる言語理解

1. コンピュータの言語処理の基本

コンピュータは、数値や明確な指示を処理するのが得意だが、人間の言語は曖昧さや多義性を含むため、理解が難しい。



2. 自然言語処理 (NLP) の課題:

1. 曖昧な表現の解釈: 同じ言葉でも文脈によって意味が変わる。
2. 多義語の処理: 例えば、「はし」という言葉は「橋」「箸」「端」など複数の意味を持つ。
3. 文脈の理解: 前後の文脈や背景知識を考慮しないと、誤った解釈をする可能性。
4. 非構造化データの扱い: 人間の言語は非構造化データであり、コンピュータが処理しにくい。

言語モデルの基本概念 (1)

言語モデルと確率的言語モデル

1. 言語モデルとは

定義: 言語モデルは、テキストデータから言語の統計的性質を学習し、文脈に応じた単語の予測や生成を行うモデル。

目的: 自然言語処理 (NLP) タスクにおいて、文の生成や解釈を支援。

2. 確率的言語モデルの概要

確率分布の学習: 言語モデルは、単語や文の出現確率を学習し、次に来る単語の予測する。



応用例: 機械翻訳、音声認識、テキスト生成など、多岐にわたるNLPタスクで活用。

言語モデルの基本概念 (2)

n-gramモデル

定義: n個の連続した単語の組み合わせを指し、nの値によりユニグラム (n=1)、バイグラム (n=2)、トリグラム (n=3) などと分類。

仕組み: 直前のn-1個の単語に基づいて、次の単語の出現確率を計算

$$P(w_n \mid w_{n-1}, w_{n-2}, \dots, w_{n-(N-1)}) = \frac{\text{Count}(w_{n-(N-1)}, \dots, w_{n-1}, w_n)}{\text{Count}(w_{n-(N-1)}, \dots, w_{n-1})}$$

具体例) “私は/プログラミングが/好きで/プログラミングが/得意です。”

✓ バイグラム (2-gram) : (私は, プログラミングが), (プログラミングが, 好きで), (好きで, プログラミングが), (プログラミングが, 得意です)

$P(\text{好きで} \mid \text{プログラミングが}) = \text{Count}(\text{プログラミングが}, \text{好きで}) / \text{Count}(\text{プログラミングが}) = 1/2$

(注) 文章が長くなるとCount数が増えて、より確率ぽくなる。

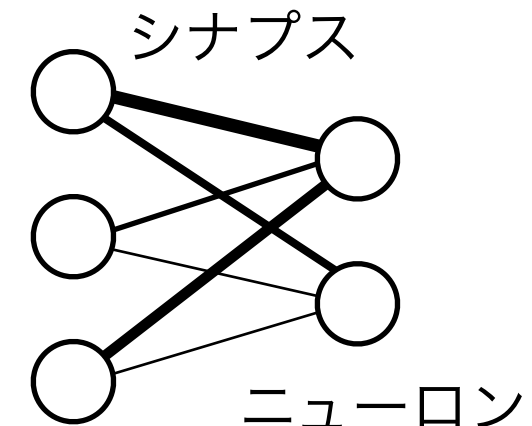
利点と限界: 簡便で計算効率が高い一方で、長距離の文脈依存性を捉えるのが難しい。

ニューラルネットワークによる言語モデル

深層学習による言語処理の進化

・ニューラルネットワーク(NN)の概要

- ・人間の脳を模倣したモデル: NNは、多数の「ニューロン」(ノード)とそれらを結ぶ「シナプス」(重み)で構成され、人間の脳の情報処理を模倣。



NNのイメージ図

- ・複雑なパターンや特徴を自動学習: 画像認識や音声認識など、多様なデータから自動的にパターンや特徴を学習。

・言語モデルとの関係

- ・言語モデルへの適用: テキストデータから言語の統計的特徴や文脈情報を学習。
- ・確率的言語モデルとの比較: n-gramモデルでは直前のn-1単語のみを考慮しているのに対して、ニューラル言語モデルでは広い文脈を考慮可能

Word2Vecと埋め込み空間

単語の意味を数値ベクトルで捉える手法

- **単語のベクトル化:**
 - 単語を数値ベクトルに変換し、意味的な類似性を表現する。
- **埋め込み空間 (Embedding Space)**
 - **意味的な類似性の可視化**
 - ベクトル空間上で単語同士の距離が近いと、意味的にも関連性が高いことを示す。
 - **類似単語の発見**
 - ベクトルの近さから類似した単語を見つける。

Word2Vecと埋め込み空間

単語の意味を数値ベクトルで捉える手法

・ 具体例

単語	ベクトル
山	(0.8, 0.6, 0.1, 0.9)
岩	(0.7, 0.2, 0.1, 0.8)
木	(0.6, 0.1, 0.9, 0.7)
森	(0.7, 0.5, 0.8, 0.8)

類似度



値が大きいものほど類似性 高

「山」 vs 「岩」 = 0.9621

「山」 vs 「森」 = 0.7227

$$\text{Cosine Similarity}(\mathbf{A}, \mathbf{B}) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \cdot \|\mathbf{B}\|}$$

・ アナロジー

$$\text{「山」} - \text{「岩」} + \text{「木」} = (0.7, 0.5, 0.9, 0.8)$$



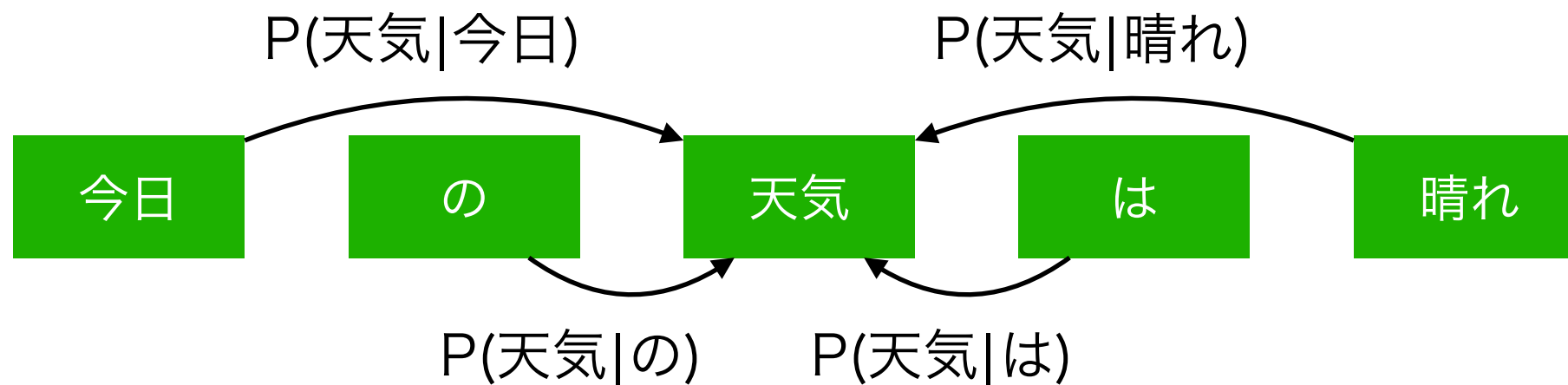
$$\text{「森」} = (0.7, 0.5, 0.8, 0.8) \text{ と非常に近い}$$

Word2Vecと埋め込み空間

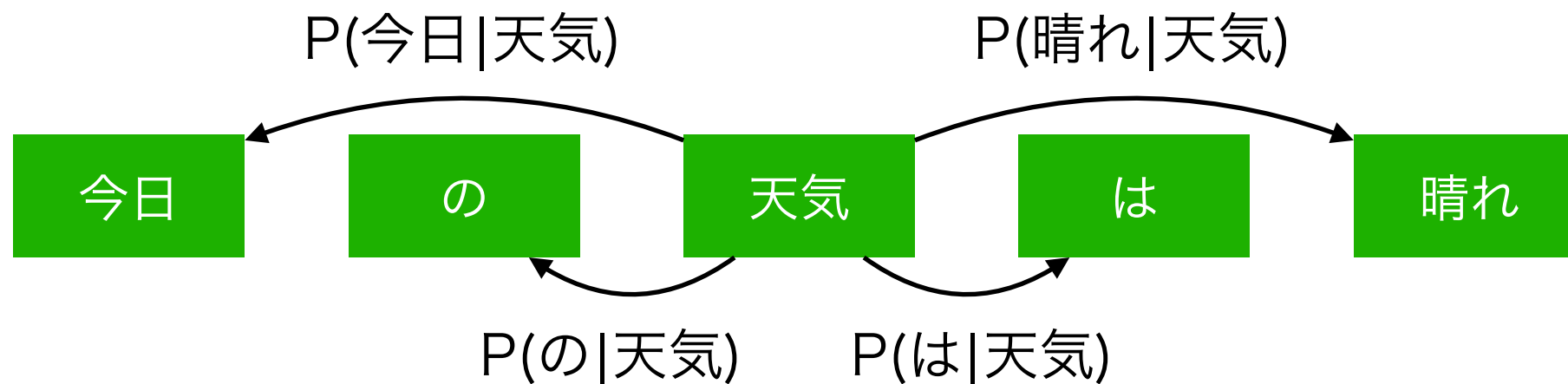
単語の意味を数値ベクトルで捉える手法

- モデルの種類:

- CBOWモデル:** 周囲の単語から中心の単語を予測。



- Skip-gramモデル:** 中心の単語から周囲の単語を予測。



Word2Vecと埋め込み空間

単語の意味を数値ベクトルで捉える手法

- 単語の予測確率

- ・ ソフトマックス関数を用いて計算。

$$\sigma(z_i) = \frac{\exp(z_i)}{\sum_{j=1}^n \exp(z_j)} \quad \text{for } i = 1, 2, \dots, n$$

- 最適化する対象

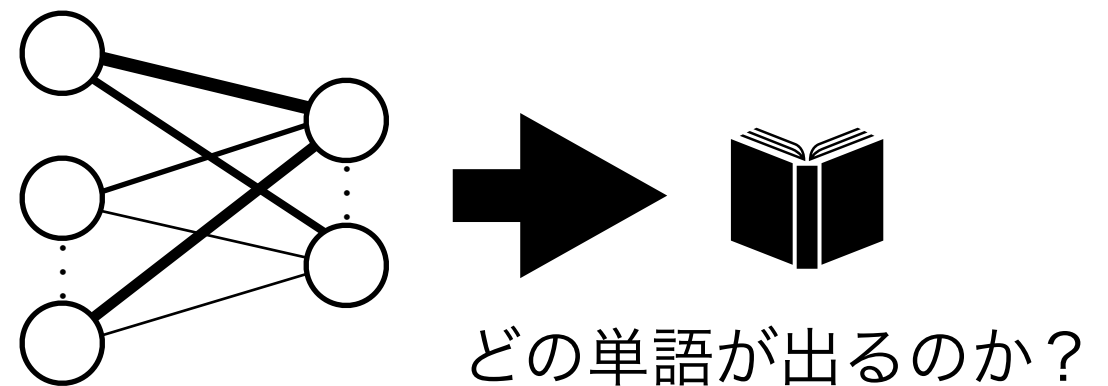
- ・ 単語の埋め込みベクトルと重み行列

- 損失関数

- ・ クロスエントロピー損失 (正解ラベルの確率を上げるようにする)

- メリットと限界

- ・ メリット: 単語間の関係性を数値的に捉えられ、分類、クラスタリングなどの性能向上に寄与。
- ・ 限界: 文全体の意味や文脈を十分に捉えきれない場合がある。

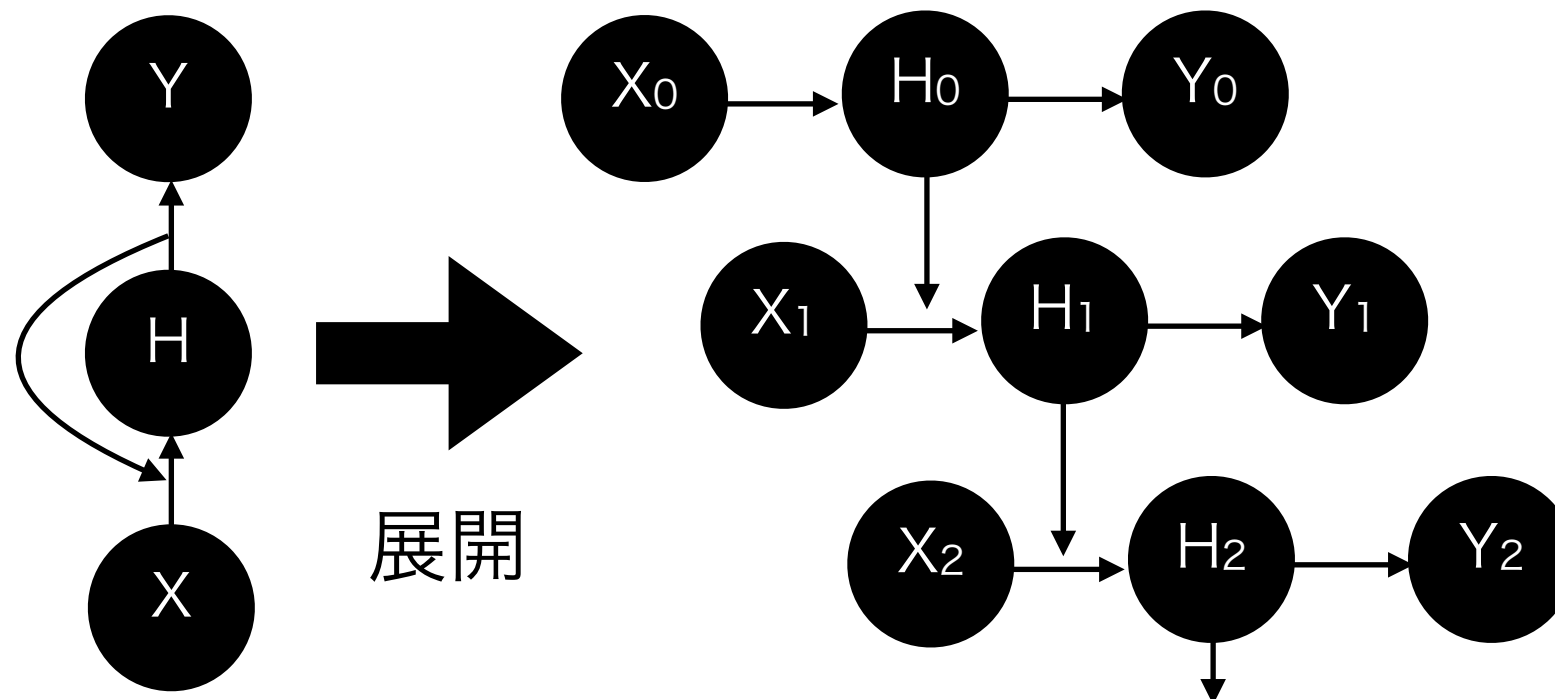


リカレントニューラルネットワーク (RNN)

時系列データ処理と言語モデルへの応用

- RNNの概要

- **時系列データに強い**: シーケンシャルなデータを扱うために設計されたニューラルネットワーク。
- **再帰的構造を持つ**: 隠れ層の出力を次の時間ステップの入力として使用することで、過去の情報を保持する。



リカレントニューラルネットワーク (RNN)

時系列データ処理と言語モデルへの応用

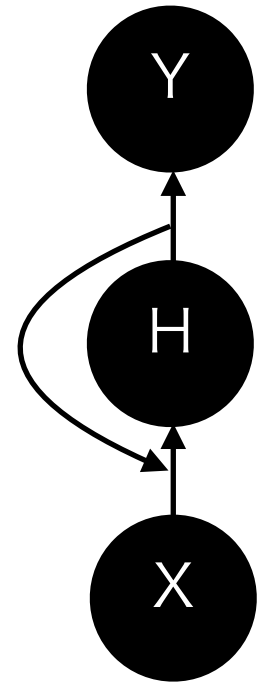
1. 隠れ状態の計算

$$h_t = f(W_{xh}x_t + W_{hh}h_{t-1} + b_h)$$

2. 出力の計算

$$y_t = \text{softmax}(W_{yh}h_t + b_y)$$

- 入力: x_t (時刻 t の入力)
- 隠れ状態: h_t (時刻 t の隠れ状態、過去の情報の要約)
- 出力: y_t (時刻 t の出力)



損失関数は、**全時刻の損失の和**または**平均**を取る

リカレントニューラルネットワーク (RNN)

時系列データ処理と言語モデルへの応用

- RNNと言語モデル

- **文脈情報の活用:** 前の単語や文脈を考慮して次の単語を予測できる。
- **長距離依存関係の学習:** 長い文章でも前後の関係を捉えられる。

- RNNの課題と改良

- **時系列が長い場合は、後ろの情報の影響が薄まる**ため、長い依存関係のあるデータには不向き。
- **勾配消失・勾配爆発問題:** 長いシーケンスでは学習が困難になることがある。

トランスフォーマーの導入

自己注意機構による革新的なアーキテクチャ

- トランスフォーマーの背景
 - RNNの限界
 - 長距離依存関係の学習が難しい
 - 並列処理が困難で学習に時間がかかる
- トランスフォーマーの解決策
 - 自己注意機構（Self-Attention）の導入
 - 並列処理が可能になり、学習効率が向上

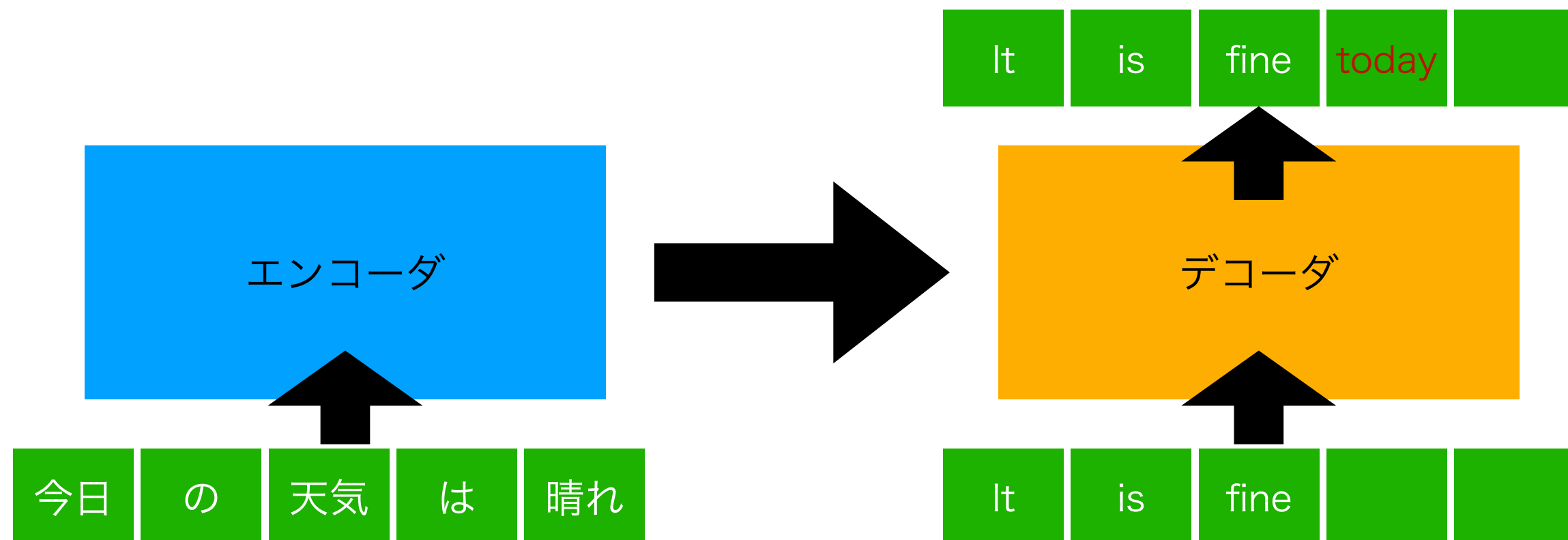
トランスフォーマーの全体構造

モデルの主要コンポーネントの概要

- トランスフォーマーの基本構造

- エンコーダとデコーダ

- エンコーダ：入力シーケンスを高次元の表現に変換
- デコーダ：エンコーダの出力を用いて出力シーケンスを生成



トランスフォーマーの構造と特徴

位置エンコーディング (1)

- 例) The cat sat on the cat

トークンID (ID毎にEmbedding vectorが定義)

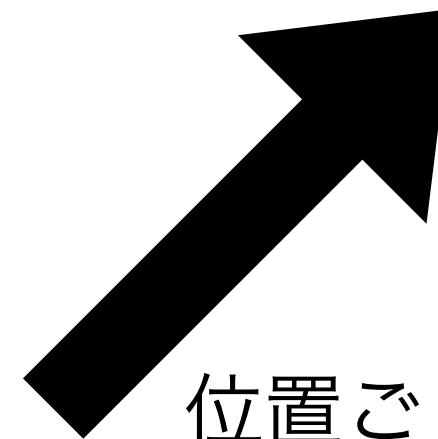
単語 (トークン)	トークンID
The	101
cat	201
sat	301
on	401
the	101
mat	501

- The cat sat on the cat

→ 101, 201, 301, 401, 101, 201

位置インデックス

位置インデックス	トークン
0	The
1	cat
2	sat
3	on
4	the
5	cat



位置ごとに変わるベクトルを
足し込むことで区別！

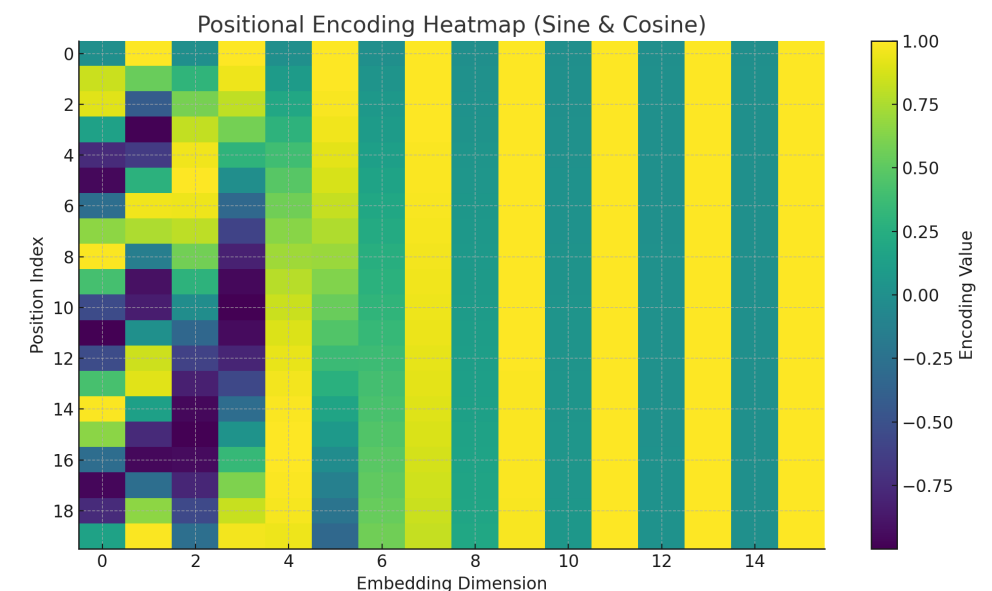
トランスフォーマーの構造と特徴

位置エンコーディング (2)

- 位置エンコーディング
 - 単語の位置情報をモデルに提供 (入力時に単語ベクトルに足し込む)
 - 順序情報がないと文の意味を正しく理解できない
 - 計算式 (サイン波・コサイン波によるエンコーディング) :

$$\text{PE}_{(pos, 2i)} = \sin\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right)$$
$$\text{PE}_{(pos, 2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right)$$

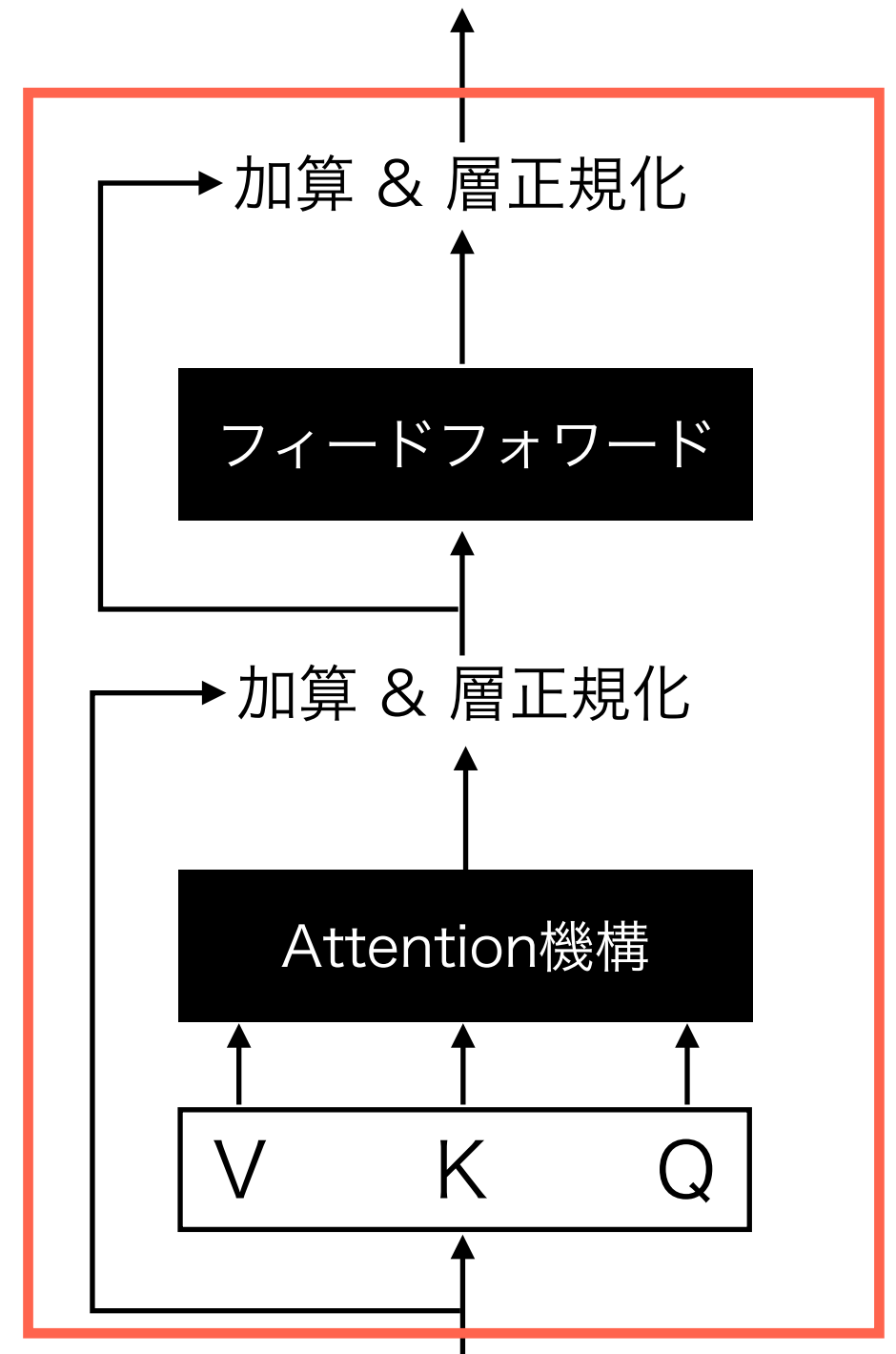
- pos : 単語の位置
- i : 次元数のインデックス
- d_{model} : モデルの次元数



トランスフォーマーの全体構造

モデルの主要コンポーネントの概要

- ・ エンコーダ、デコーダともに、基本的には以下のブロックの積み重ね
- ・ ブロックの構成要素
 - ・ バリュー(V)、キー(K)、クエリ(Q)
 - ・ 自己注意機構 (Self-Attention)
 - ・ フィードフォワードネットワーク (Feed-Forward Network)
 - ・ 残差接続と層正規化



自己注意機構 (Self-Attention) の仕組み

クエリ、キー、バリューとは？

- ・ 自己注意機構の基本概念

- ・ クエリ (Query) 、キー (Key) 、バリュー (Value) とは何か？

- ・ クエリ (Q) : 注目する単語が「何を知りたいか」を表す。

- ・ キー (K) : 他の単語が「どんな情報を持っているか」を表す。

- ・ バリュー (V) : 他の単語が持つ実際の情報を表すベクトル

- ・ 大まかなイメージ

- ・ QとKの内積を取ることで関連性(類似度)を算出

- ・ 確率分布密度をsoftmaxで求めて前のVにかけて、単語ベクトル(V)を更新する。

自己注意機構 (Self-Attention) の仕組み

実際の計算のステップ

1. 入力ベクトル (H)に重み行列(W)を乗算して、 d_k 次元のキー (K)、クエリ (Q)、バリュー (V)の生成する。

$$K=W_k H, \quad Q=W_Q H, \quad V=W_V H$$

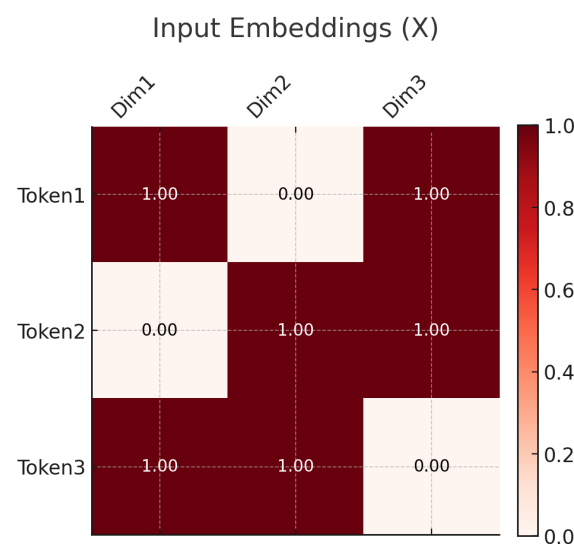
2. 注目する単語のクエリと他の単語のキーとの内積 (QK^T)を計算し、関連度 (注意スコア) を計算
3. ソフトマックス関数で正規化
4. 各バリューに対応する重みを掛けて合計する

$$\text{Attention}(Q, K, V) = \text{softmax} \left(\frac{QK^T}{\sqrt{d_k}} \right) V$$

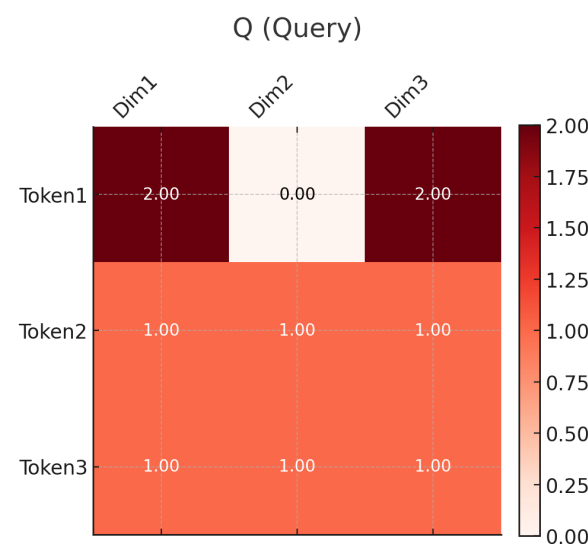
自己注意機構 (Self-Attention) の仕組み

計算のイメージ

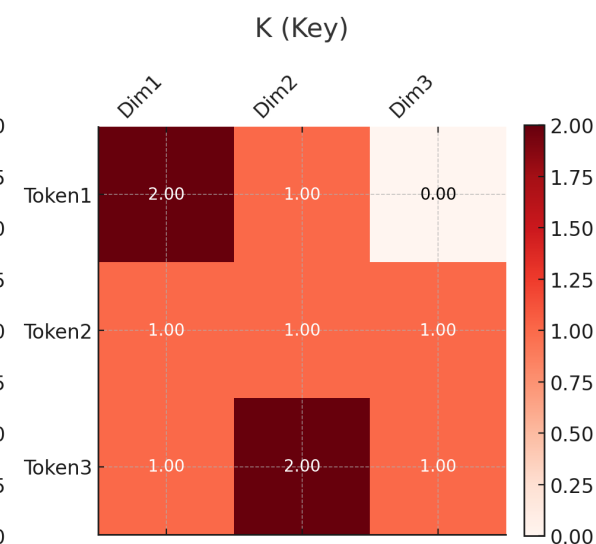
初期の埋め込みベクトル



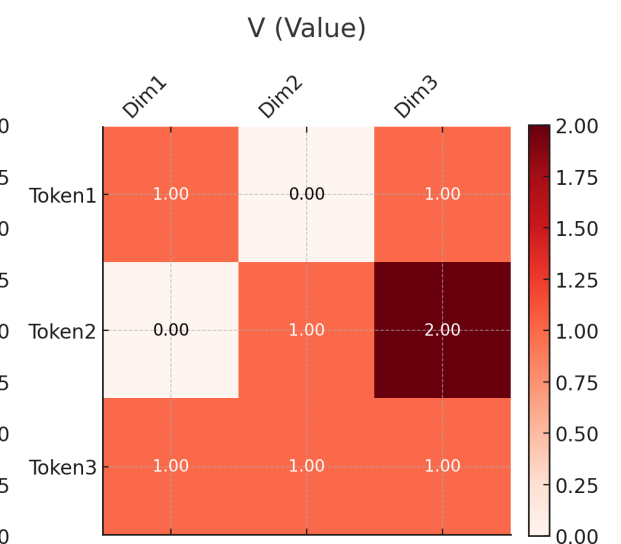
どんな情報が欲しいか？



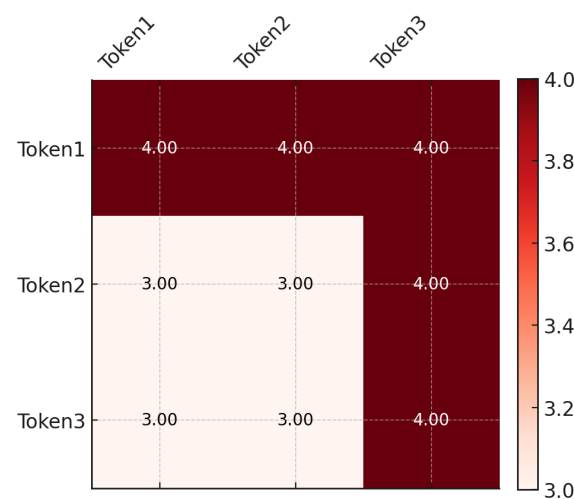
どんな情報を持っているか？



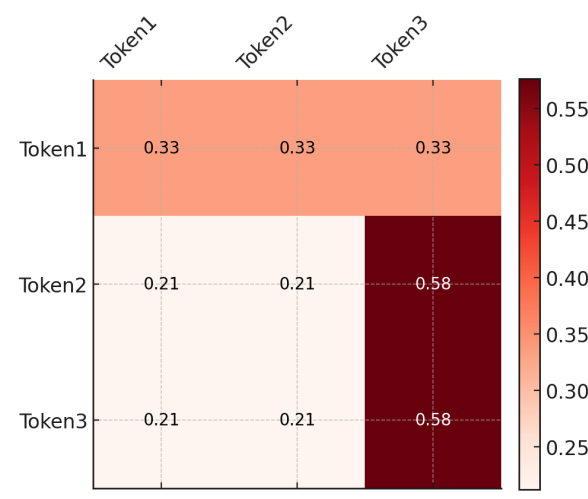
現在持っている情報



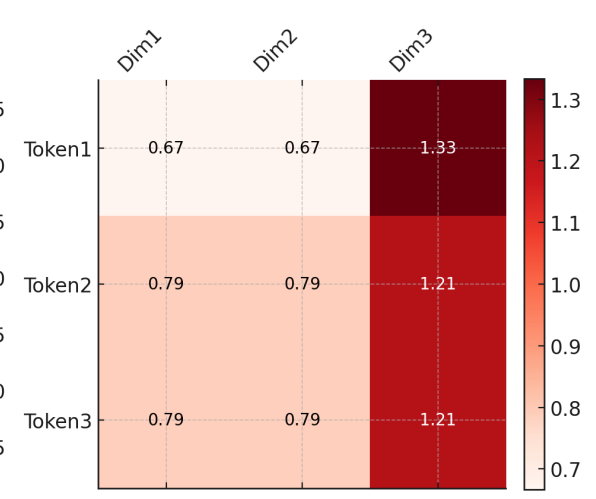
Attention Scores ($Q * K^T$)



Attention Weights (Softmax)



Output (Weighted Sum of V)



情報の類似度 (スコア)

スコアの正規化

出力結果

トランスフォーマーの構造と特徴

マルチヘッドアテンション

- ・マルチヘッドアテンション
 - ・複数の自己注意機構を並列に実行
 - ・異なる位置や特徴に同時に焦点を当てる
 - ・ 計算式:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)W^O$$

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

- ・ W_i^Q, W_i^K, W_i^V : 各ヘッドの重み行列
 - ・ W^O : 出力の重み行列
- ・ メリット
 - ・ 多様な関係性を一度に学習可能
 - ・ モデルの表現力が向上

フィードフォワードネットワーク (FFN)

モデルの表現力を高める非線形変換

- ・ フィードフォワードネットワークの構造
 - ・ 位置ごとに適用される全結合ネットワーク：各単語の表現を個別に処理
 - ・ 具体的な構造：2つの線形変換と活性化関数（通常はReLU）から構成

- ・ 数式：

$$\text{FFN}(x) = \text{ReLU}(xW_1 + b_1)W_2 + b_2$$

- ・ x ：入力ベクトル
 - ・ W_1, W_2 ：重み行列
 - ・ b_1, b_2 ：バイアス項

- ・ 役割と効果
 - ・ 非線形性の導入：モデルがより複雑な関数を学習可能
 - ・ 位置ごとの独立性：並列計算が可能で高速な処理が実現

残差接続とレイヤー正規化

深層学習における学習の安定化と効率化

- 残差接続 (Residual Connections)

- 各サブレイヤーの入力と出力を直接結合し、深いネットワークにおける勾配消失問題を緩和する。

$$\text{出力} = x + \text{サブレイヤー}(x)$$

- x : サブレイヤーへの入力

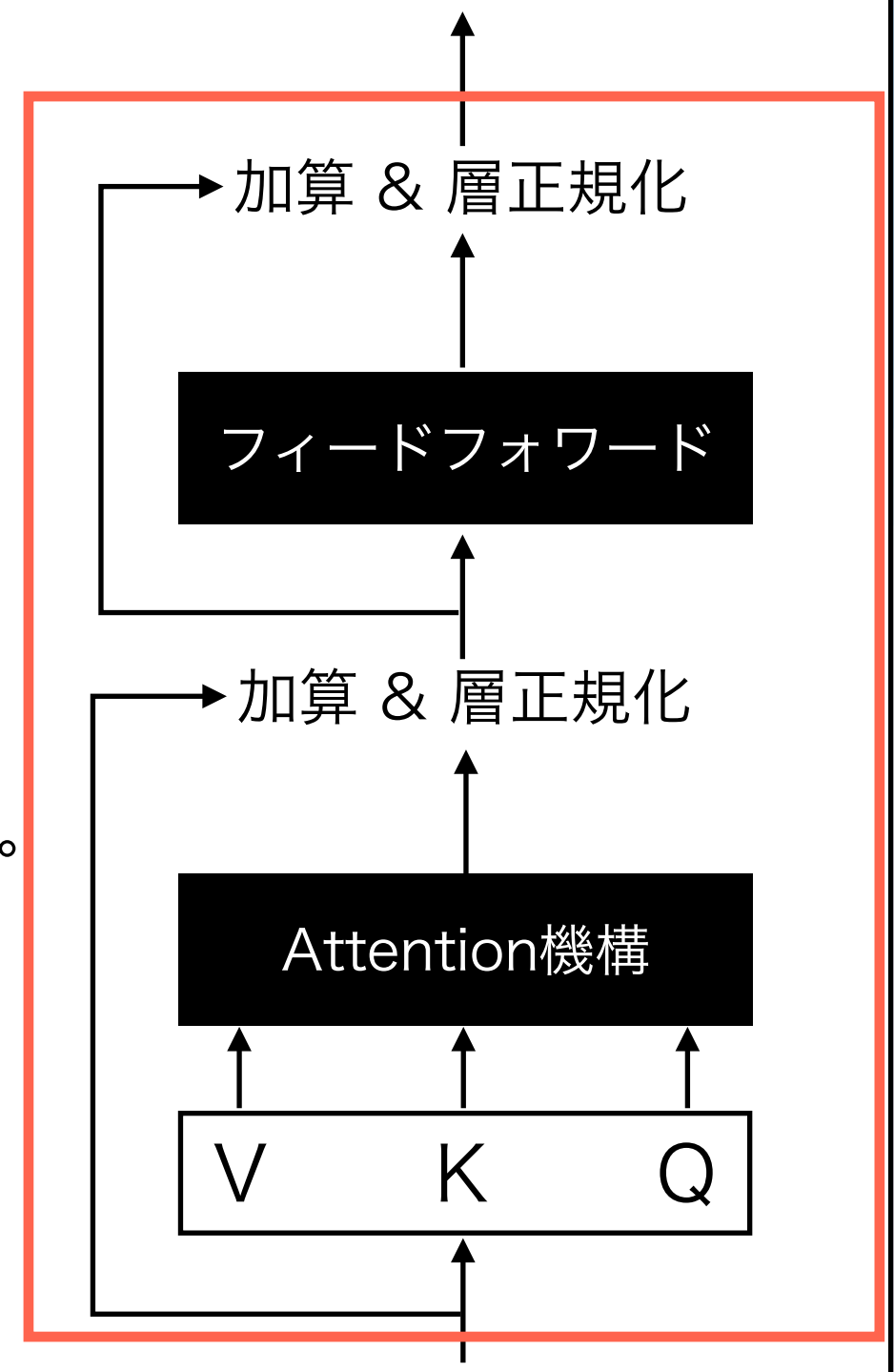
- レイヤー正規化 (Layer Normalization)

- 各層の入力を正規化し、データ分布の変動を抑制する。

$$\text{LayerNorm}(x) = \frac{x - \mu}{\sigma} \gamma + \beta$$

- μ, σ : 入力 x の平均と標準偏差

- γ, β : 学習可能なパラメータ



トランスフォーマーの応用例

トランスフォーマーがもたらす実世界での成果

- 機械翻訳への応用：高精度な翻訳を実現

- エンコーダ・デコーダモデルの数式表現

- エンコーダの出力: $H = \text{Encoder}(X)$

- デコーダの出力: $Y = \text{Decoder}(H, Y_{\text{shifted}})$

- 文章生成：自然で一貫性のある文章を生成(対話システム)

- 次の単語の予測確率計算:

$$P(y_t \mid y_{<t}, X) = \text{softmax}(\text{Linear}(\text{Decoder}(H, y_{<t})))$$

- y_t : 時刻 t の出力単語

- $y_{<t}$: 時刻 t 未満の生成済み単語

(注) ChatGPTの場合はデコーダのみで構成されるため、エンコーダの入力 X と出力 H は必要ない。

トランスフォーマーの発展と最新モデル

GPTとBERTによる自然言語処理の革新

- GPT (Generative Pre-trained Transformer)
 - トランスフォーマーデコーダを使用した自己回帰型モデル。
 - 大規模なテキストデータで事前学習。
- BERT (Bidirectional Encoder Representations from Transformers)
 - 双方向のトランスフォーマーエンコーダを使用
 - マスク付き言語モデル (Masked Language Model) として事前学習。
 - 入力文の一部をマスクして、そのマスクされた単語を予測することでモデルを訓練する。これにより、文脈の両側からの情報を活用。

本日本話しする内容

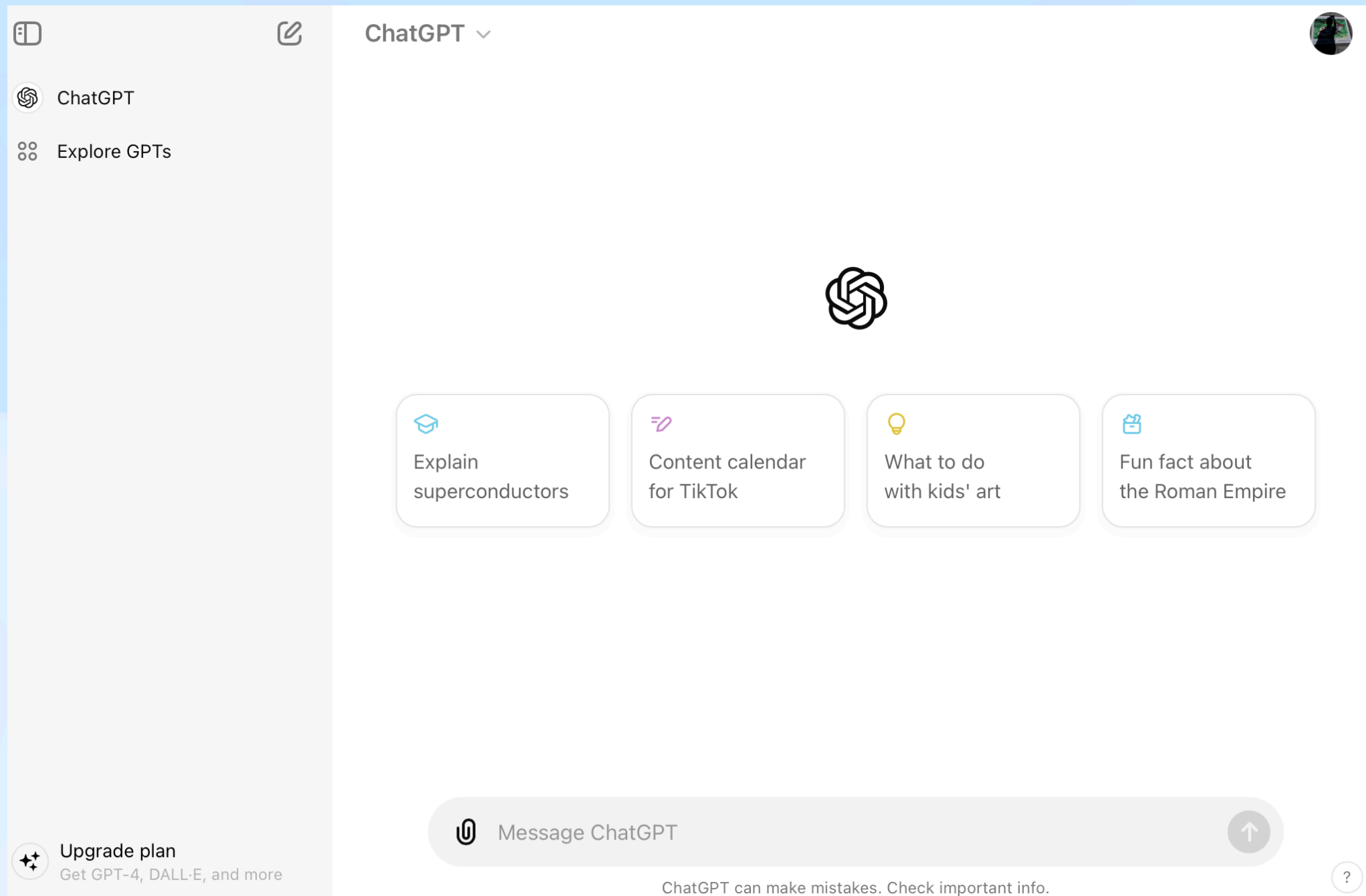
- ・ LLMの基礎知識 ～ 言語モデルの仕組みとその進化
- ・ LLMツール(ChatGPTとGitHub Copilot)の概要
- ・ LLMツール(ChatGPTとGitHub Copilot)の環境構築
- ・ まとめ

LLMツール(ChatGPTとGitHub Copilot)の概要

各ツールの特徴と活用例を理解する

- ・ このセクションで学ぶこと
 - ・ ChatGPTとGitHub Copilotの基本的な違いを理解する。
 - ・ 各ツールがどのように計算物質科学で役立つかを知る。
 - ・ 実際の使用例から活用方法のイメージを得る。

ChatGPTとは？



ChatGPTとは？

- ChatGPTとは？



<https://chatgpt.com/>

- 対話型のLarge Language Model (LLM)
- OpenAIによって開発されたAI言語モデル。
- 膨大なテキストデータを用いてトレーニングされている。
 - ✓ 学習データ量：GPT-4 10兆トークン、GPT-3.5 約3410億トークン
 - ✓ パラメータ数：GPT-4 約100兆個、GPT-3.5 約1750億個)

(注) GPT-4は正式な数値は発表していないので巷で予想されている値

- ChatGPTができること

- 自然言語を理解し、質問応答や文章生成が可能。
- 人間のようなテキスト会話の実行、質問への回答、説明の提供など
(最近では、画像・ファイル・音声入力などにも対応)。

ChatGPTで選択できるモデル

モデル名	主な特徴	主な用途
GPT-4	- 高度な自然言語処理能力。 - 複雑なタスクへの対応。 - 多言語対応。 - 大規模なコンテキストウィンドウ。	- 高度な文章生成。 - 複雑な質問応答。 - 専門的な内容の要約。
GPT-4o	- GPT-4と同等の知能レベル。 - 応答速度がGPT-4 Turboの2倍。 - コスト効率がGPT-4 Turboの半分。 - テキスト、ビジョン、オーディオの統合処理。	- リアルタイムの対話。 - マルチモーダルデータの処理。 - コスト効率を重視したアプリケーション。
GPT-4o mini	- GPT-4oの軽量版。 - 高速な応答。 - コスト効率が高い。 - 小規模なタスクに最適化。	- 簡易なテキスト処理。 - リアルタイムのカスタマーサポート。 - 軽量なアプリケーション。
o1-preview	- 人間のような推論能力を持つ。 - 複雑な問題解決に優れる。 - 数学やプログラミングタスクで高い性能を発揮。 - 内部的な「思考」プロセスを持ち、回答の一貫性と精度が向上。	- 高度な問題解決。 - 数学的推論。 - 複雑なプログラミングタスク。

ChatGPTの価格プラン

プラン名	価格 (米ドル)	主な特徴
無料版	\$0/月	- 基本的な機能を利用可能。 - GPT-4o miniへのアクセス。
Plus	\$20/月	- GPT-4、GPT-4o、GPT-4o miniへのアクセス。 - データ分析、ファイルアップロード、ビジョン、Webブラウジングへのアクセス。 - 高度な音声モードへのアクセス。 - DALL-E画像生成。 - カスタムGPTの作成と利用。 - 新機能への早期アクセス。
Team	\$25/ユーザー/月 (年額課金) \$30/ユーザー/月 (月額課金)	- Plusの全機能。 - GPT-4o miniへの無制限アクセス。 - ワークスペースでのGPTの作成と共有。 - ワークスペース管理用の管理コンソール。 - チームデータはデフォルトで学習から除外。
Enterprise	要問い合わせ	- Teamの全機能。 - GPT-4、GPT-4o、GPT-4o miniへの無制限の高速アクセス。 - 拡張されたコンテキストウィンドウ。 - カスタムのデータ保持ウィンドウ。 - 管理者制御、ドメイン認証、アナリティクス。 - 充実したサポートと継続的なアカウント管理。

(注) 月額\$200のProプランが2024/12/5にリリース (GPT4-o1が無制限に利用可能)

https://openai.com/ja-JP/chatgpt/pricing/?utm_source=chatgpt.com

ChatGPTの計算物質科学への使用例(1)

専門用語の解説と翻訳



- ・ 問題
 - ・ 専門文献を参照する際、専門用語の理解や翻訳に時間がかかる。
- ・ ChatGPTの使い方
 - ・ 専門語の解説

「以下の専門用語の意味を簡潔に説明してください：第一原理計算」
 - ・ 英語論文の日本語翻訳

「以下の英文を日本語に翻訳してください： 'The band structure calculations reveal that the material exhibits a direct band gap at the Γ point, indicating potential applications in optoelectronic devices.'」
 - ・ 日本語文献の英語翻訳

「以下の日本語文を英語に翻訳してください： 'この材料は高い熱伝導率を持ち、電子デバイスの冷却用途に適しています。'」

ChatGPTの計算物質科学への使用例(2)

論文からの情報抽出



- ・ 問題
 - ・ 学術論文から必要な情報を効率的に抽出する必要がある。
- ・ ChatGPTの使い方
 - ・ 論文要約

「以下の学術論文の要点を500文字以内で要約してください。研究の背景、目的、方法、結果、結論を含めてください。」 + PDFをアップロード
 - ・ 研究トレンドの把握

「以下の複数の論文要約を基に、現在の研究トレンドを分析し、主要なテーマや注目されている課題を箇条書きで挙げてください。」
 - ・ 未解決課題を把握

「以下の論文の結論部分を基に、この研究分野における未解決の課題や今後の研究方向性を箇条書きで挙げてください。」

ChatGPTの計算物質科学への使用例(3)

研究アイデアのブレインストーミング



- ・ 問題
 - ・ 新たな研究テーマやアプローチを考案する際にアイデアが欲しい。
- ・ ChatGPTの使い方
 - ・ 計算物質科学における新規研究テーマの提案

「あなたは、計算物質科学の専門家です。
以下の制約条件に従って、 三次元画像から特徴量を抽出することを活用した、
新しい研究テーマのアイデアを考えてください。

制約条件

 - * 先行研究と比べて、十分新規性がある研究テーマを考えること
 - * 出力形式に従うこと
 - * 簡潔でわかりやすい文章で出力すること

研究分野

 - * 有機分子」

ChatGPTの計算物質科学への使用例(4)

プレゼンテーション資料の作成支援



- ・ 問題
 - ・ 研究成果を効果的に伝えるプレゼン資料の作成に時間がかかる。
- ・ ChatGPTの使い方
 - ・ 研究内容の要約作成

「以下の研究内容を簡潔に要約し、プレゼンテーションのイントロダクション用のスライドに適した文章を作成してください。」
 - ・ 複雑な概念の視覚化アイデアの提案

「以下の計算物質科学における複雑な概念を視覚的にわかりやすく説明するためのアイデアを提案してください。」
 - ・ プレゼン資料の構成案の作成

「計算物質科学に関する以下の研究テーマについて、効果的なプレゼンテーション資料の構成案を提案してください。」

ChatGPTの計算物質科学への使用例(5)

データ解析手法の提案



- ・ 問題

- ・ 複雑なデータセットに対して適切な解析手法を選定する必要がある。

- ・ ChatGPTの使い方

- ・ 第一原理計算データの解析手法の提案

「以下のデータセットは、第一原理計算によって得られた材料の電子バンド構造情報を含んでいます。各データポイントはエネルギー値と対応する波数ベクトルで構成されています。このデータからバンドギャップを正確に抽出し、材料の半導体特性を評価したいと考えています。適切な解析手法やアルゴリズムを提案してください。」

- ・ 材料特性データの機械学習モデル構築の提案

「複数の材料について、結晶構造、原子種、格子定数、弾性定数、熱伝導率などの特性データを収集しました。これらのデータを用いて、未知の材料の熱伝導率を予測するモデルを構築したいと考えています。適切な機械学習アルゴリズムと前処理方法を提案してください。」

ChatGPTの計算物質科学への使用例(6)

コード生成とデバッグ支援



- ・ 問題
 - ・ シミュレーションやデータ解析のためのコード作成に多くの時間がかかる。
- ・ ChatGPTの使い方
 - ・ データ解析用のPythonコードの自動生成

「Pythonを使用して、CSVファイルからデータを読み込み、特定の列の平均値と標準偏差を計算するコードを作成してください。」
 - ・ シミュレーションコードのバグ修正

「以下のPythonコードは、数値積分を用いて微分方程式を解くことを目的としています。しかし、実行時にエラーが発生します。コードのバグを特定し、修正してください。」
 - ・ シミュレーションコードの最適化

「以下のPythonコードは、Matplotlibを使用してデータをプロットしています。コードの効率性と可読性を向上させるための最適化を提案してください。」

ChatGPTの計算物質科学への使用例(7)

研究データの可視化提案



- ・ 問題
 - ・ データの可視化方法を検討する際、最適なグラフや図の選定に悩む。
- ・ ChatGPTの使い方
 - ・ 材料の電子バンド構造データの可視化

「以下の電子バンド構造データを効果的に可視化するための最適なグラフ形式を提案してください。データは波数ベクトルと対応するエネルギー値で構成されています。視覚的にバンドギャップやディスパージョン関係を明確に示したいと考えています。」
 - ・ 材料特性の相関関係の可視化

「複数の材料特性（例：熱伝導率、電気伝導率、弾性率）の相関関係を可視化したいと考えています。これらの特性間の関係性やトレンドを明確に示すための最適なグラフ形式を提案してください。」

GitHub Copilotとは？



- ✓ GitHubとOpenAIが共同開発したAIコード補完ツール
- ✓ コードの予測・提案を行い、開発者を支援

<https://github.com/features/copilot>

主な機能

1. AIベースのコード補完

→ 高度な機械学習モデルを使用してコードを予測し、自動補完

例: ループ構造や関数呼び出しの提案など

2. リアルタイムのコード提案

→ 入力中に即座にコードのサジェストを表示し、効率的なコーディングを支援

例: 入力途中の変数名や関数名の候補を表示

3. マルチ言語対応

→ Python、JavaScript、Javaなど多言語に対応し、多様なプロジェクトで利用可能

例: 異なる言語間で一貫した提案

GitHub Copilotの提案生成方法

1. オープンソースから学習したパターンを活用

- **学習元:** 大量のオープンソースコードを解析
- **利点:** 一般的なコーディングパターンやベストプラクティスを提案
- **結果:** 効率的で読みやすいコードを自動生成



2. 現在編集中のファイルの内容を解析

- **解析対象:** コード本体とコメントを含むファイルの内容
- **文脈に基づく提案:** 直前のコードやコメントに即したコードを提案



3. 関連ファイルとプロジェクト全体の設定を考慮

- **解析範囲:** 他の開いているタブ、関連ファイル、プロジェクトフォルダの設定ファイル
- **使う情報:** コード内容、設定ファイル（例: settings.json など）
- **結果:** 他ファイルのコード、設定情報、フォルダ構造を基に、プロジェクト全体に即した提案を行う。

GitHub Copilot Pricing Plans

Subscription Plans

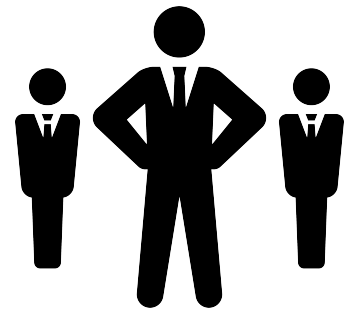
プラン名	対象ユーザー	価格（米ドル）	支払いサイクル	主な機能と特典
Copilot Individual	個人ユーザー	月額 \$10 または 年額 \$100	月単位/年単位	- コード補完 - チャット機能 - コマンドラインでの質問 - プルリクエストのサマリー生成 - 30日間の無料トライアル
Copilot Business	組織	月額 \$19/ユーザー	月単位	- Copilot Individualの全機能 - ユーザー管理 - ポリシー管理 - 監査ログ - IP補償
Copilot Enterprise	企業	月額 \$39/ユーザー	月単位	- Copilot Businessの全機能 - カスタム大規模言語モデルの微調整



学生、教育者、オープンソースのメンテナンス担当者は、条件を満たすことで Copilot Individualを無料で利用できます。

伝統的?なソフトウェア開発の課題

- ・ 開発者のスキルや経験による生産性・レビューの差
- ・ 手動で行うテストとレビューに時間がかかる
- ・ マニュアルやサンプル事例の作成に多大な工数が必要
- ・ テスト漏れによるバグ発生リスク
- ・ セキュリティ上の脆弱性を見逃しやすい etc...

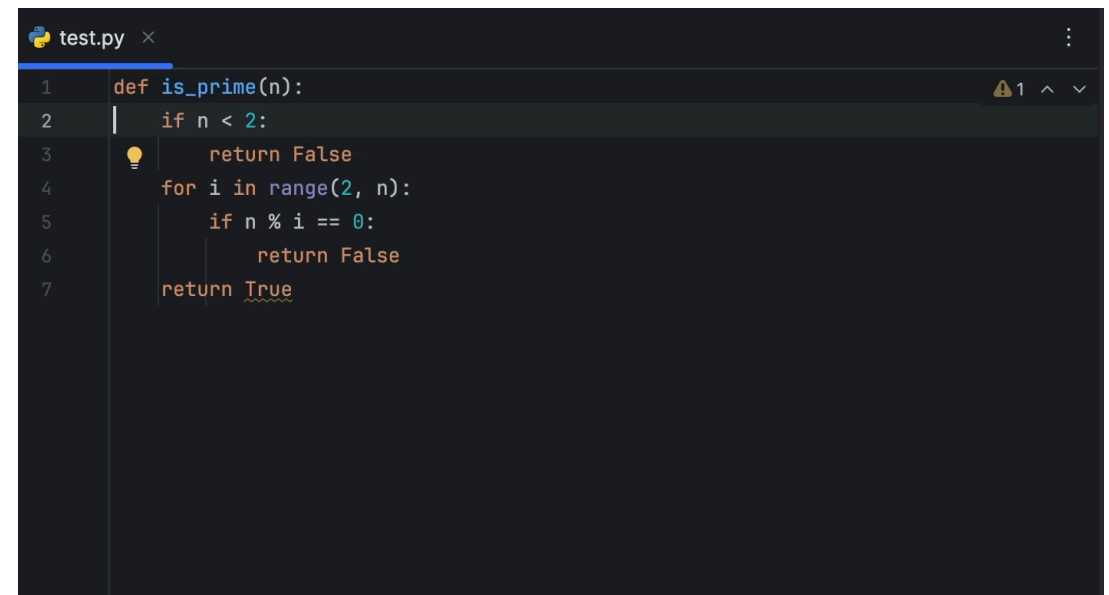
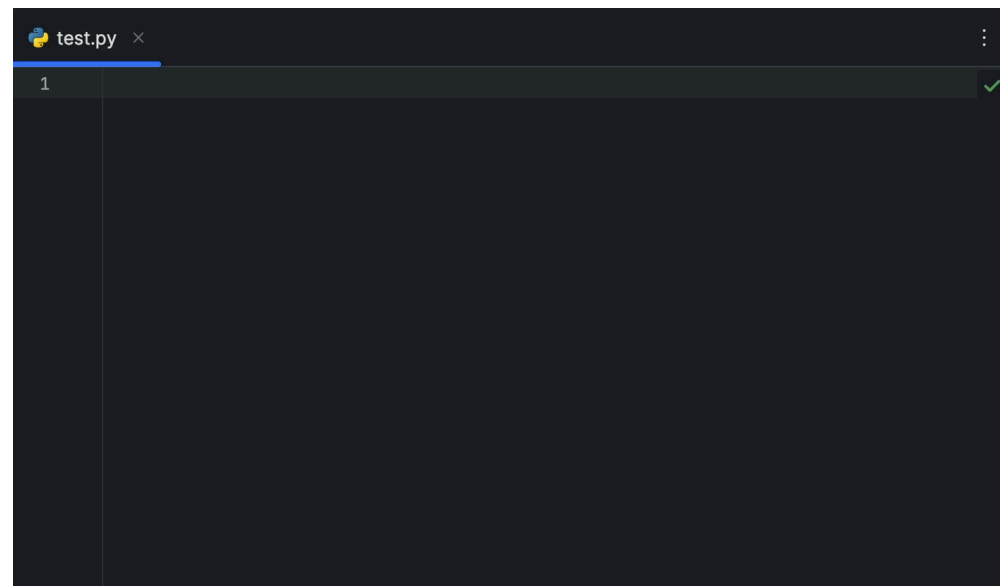


AIを活用したコーディング支援の利点

- 自動コード補完による開発スピードの向上
(例: GitHub Copilot)
- リアルタイムのエラー検出と修正提案
- ドキュメント自動生成やテストコードの生成

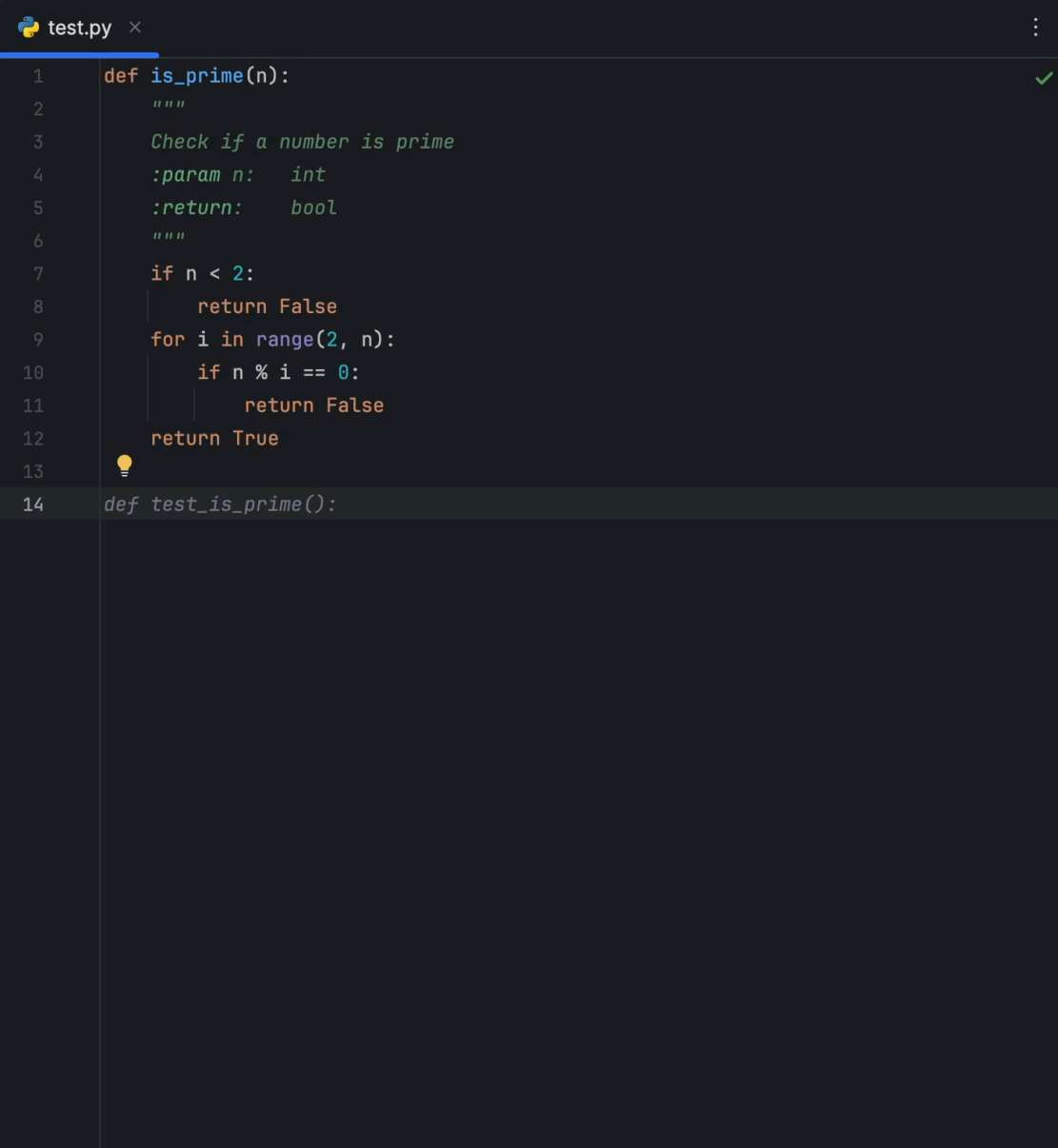


例) GitHub Copilotを用いた自動コード補完



AIがもたらすソフトウェア品質の向上

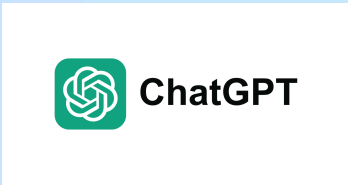
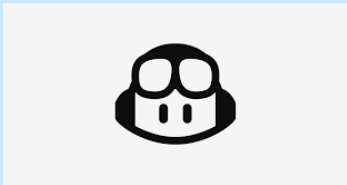
- テスト生成の導入によるバグの早期発見
- コードレビューの自動化と品質の均一化
- セキュリティ脆弱性の事前検出



```
test.py ×
1 def is_prime(n):
2     """
3     Check if a number is prime
4     :param n: int
5     :return: bool
6     """
7     if n < 2:
8         return False
9     for i in range(2, n):
10         if n % i == 0:
11             return False
12     return True
13
14 def test_is_prime():
```

例) GitHub Copilotを用いたテスト生成

GitHub CopilotとChatGPTのコード開発に対する違い



Feature	GitHub Copilot	ChatGPT
Integration	IDEに直接統合	チャットを通じて使用
リアルタイム提案	あり	なし
自然言語理解	コード補完を主眼	高い
コンテキスト認識	高い(現在編集集中のファイル内)	入力内容に応じて変化
使用例	コード補完、リファクタリングなど	コード生成、トラブルシューティングなど

GitHub Copilotは、IDE内でのコード補完や自動入力をスピーディに行いたい場合に強み。
ChatGPTは、深い理解を要するタスクや、詳細なコード生成、問題解決に強み。

本日本話しする内容

- ・ LLMの基礎知識 ～ 言語モデルの仕組みとその進化
- ・ LLMツール(ChatGPTとGitHub Copilot)の概要
- ・ **LLMツール(ChatGPTとGitHub Copilot)の環境構築**
- ・ まとめ

LLMツール(ChatGPTとGitHub Copilot)の環境構築

各ツールの導入手順と設定方法を理解する

- ・ このセクションで学ぶこと
 - ・ 各ツールのインストール手順を習得し、研究・開発環境に適切に組み込む方法を学ぶ。
 - ・ 効果的な設定とカスタマイズを行い、開発効率を最大化する。

ChatGPTへの登録手順

1. 公式サイトへのアクセス:

- [ChatGPTの公式ページ](#)にアクセス。

2. アカウントの作成:

- ページ右上の「Sign Up」ボタンをクリック。
- メールアドレスを入力し、パスワードを設定。
 - GoogleやMicrosoftのアカウントを使用して登録することも可能。

3. メールアドレスの確認:

- 登録したメールアドレス宛に確認メールが送信される。
- メール内のリンクをクリックして、メールアドレスを確認。

4. 個人情報の入力:

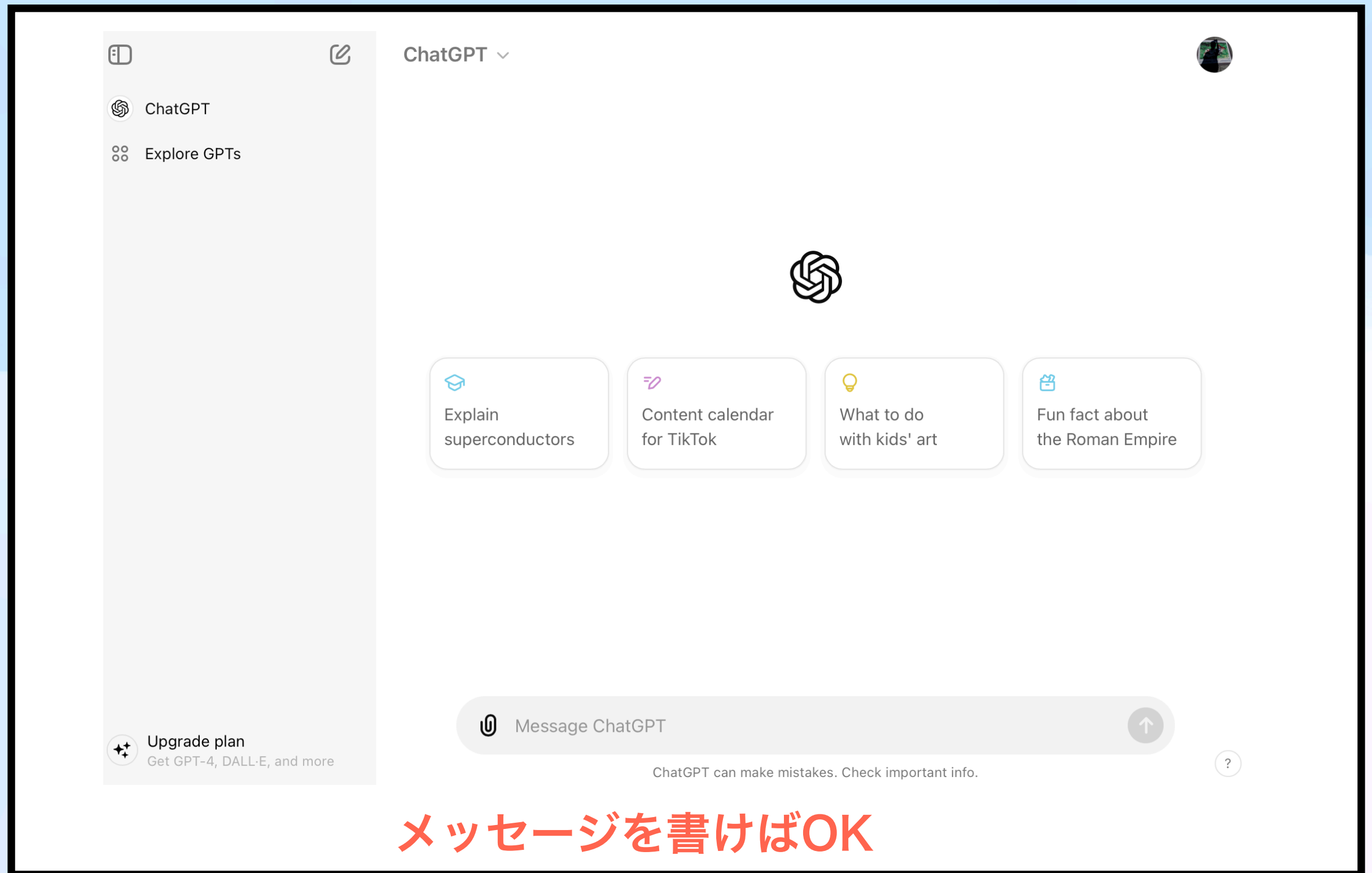
- 氏名と誕生日を入力。
 - 誕生日の入力は、年齢確認のために必要。

5. 電話番号の確認:

- 電話番号を入力し、SMSで送信されるコードを使用して確認。

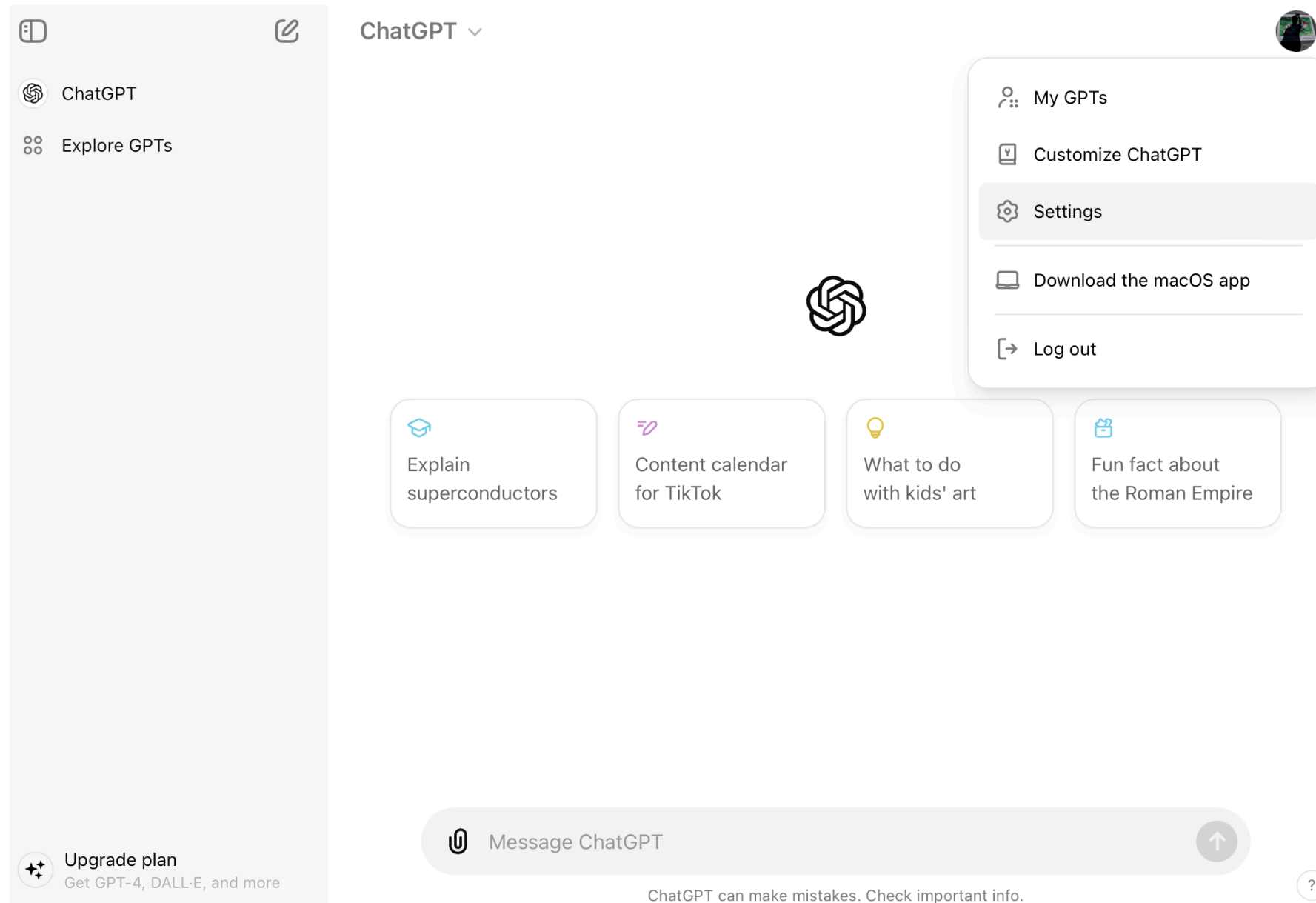
ChatGPTの使い方

基本的な使用方法



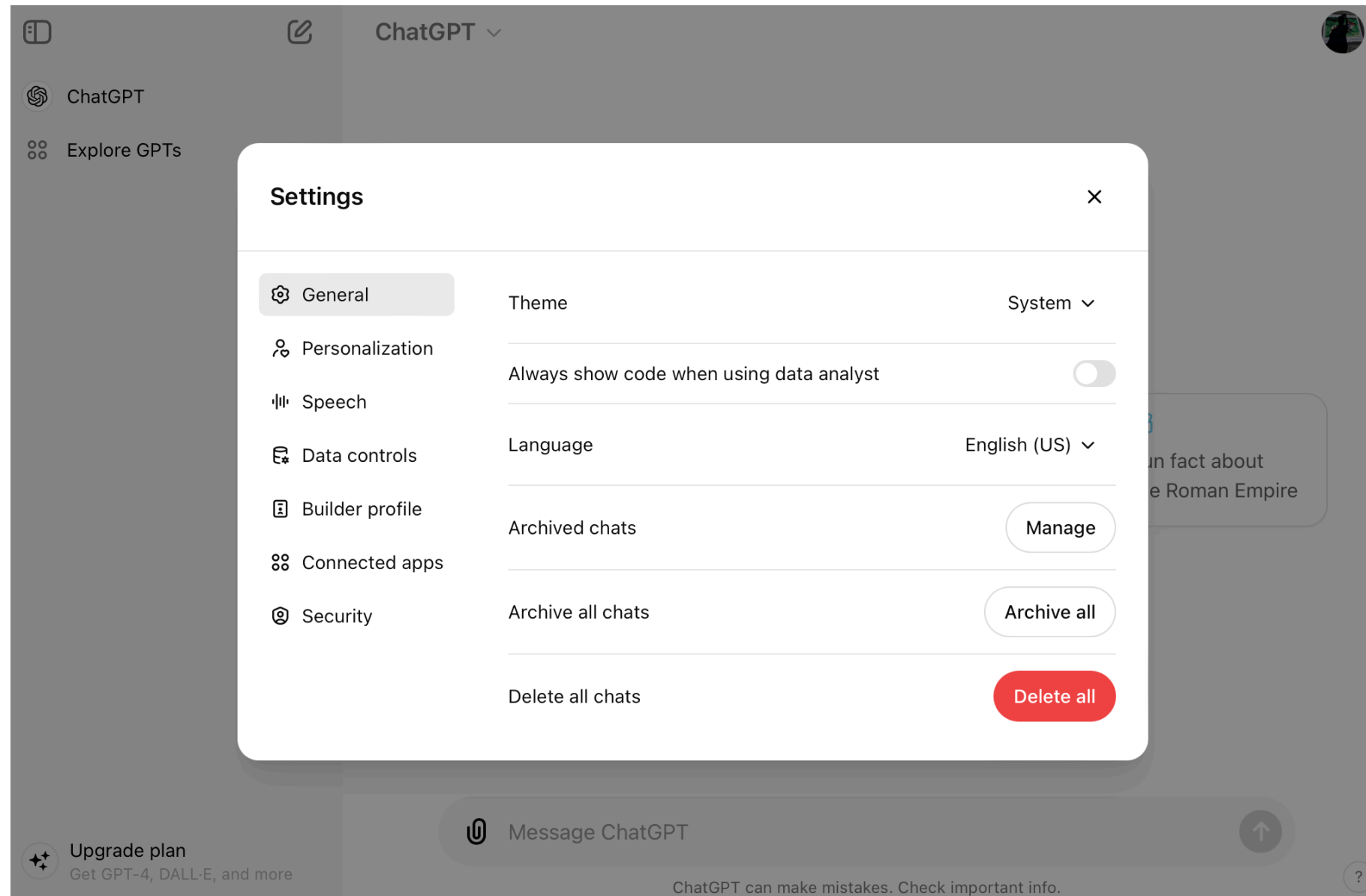
メッセージを書けばOK

ChatGPTの環境設定(1)



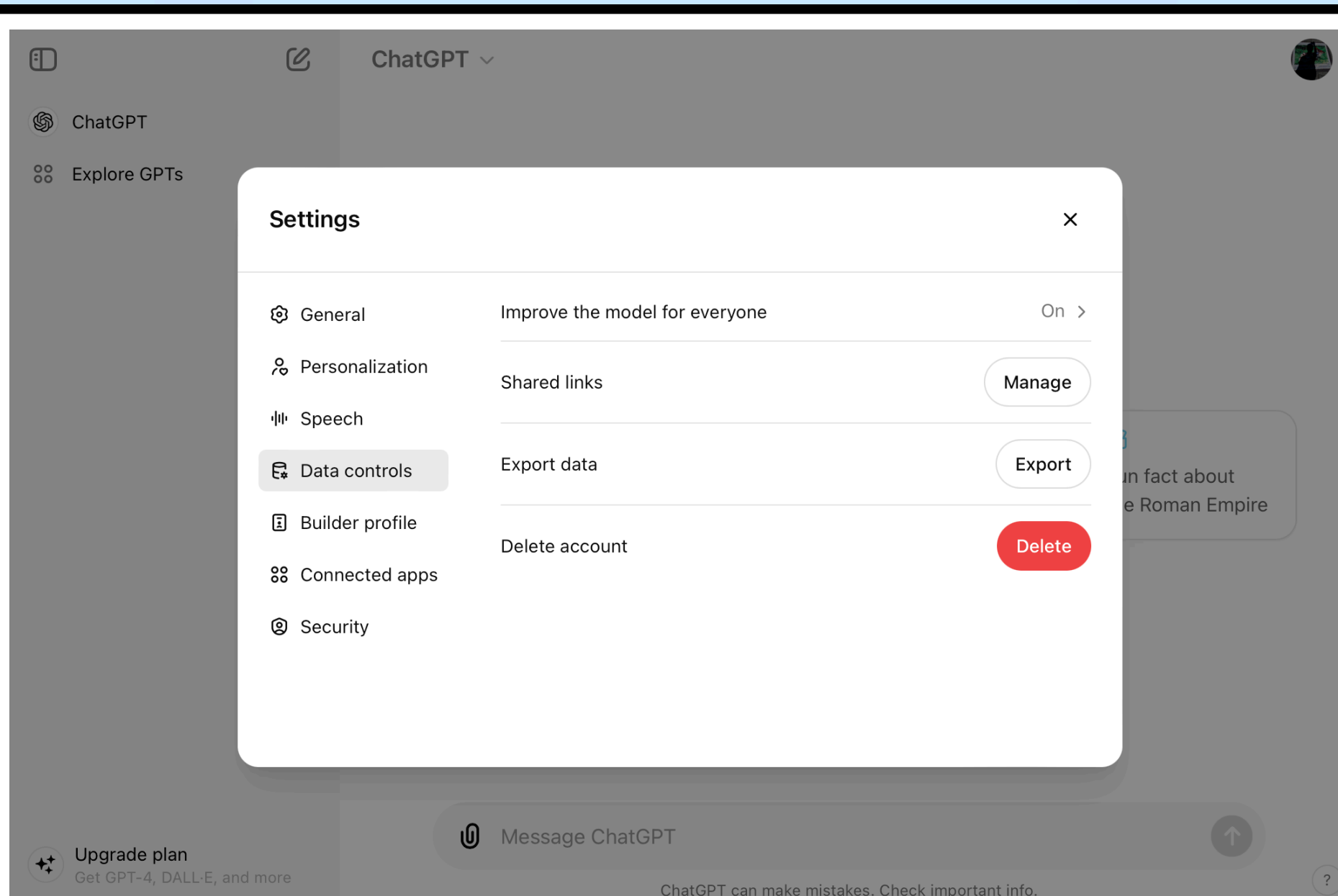
右上のロゴをクリックした後に、「Settings(設定)」をクリック。

ChatGPTの環境設定(2)



“General(一般)” : Theme, total deletion of history etc.

ChatGPTの環境設定(3)



“Data controls(データコントロール)”：

“Improve the model for everyone”

→ モデルの学習のためにコンテンツを利用することを許可するかどうか。

ChatGPTの環境設定(4)



“Improve the model for everyone”

ON：過去の入力内容を学習するので、使いやすくなる。ただし、チャット履歴は他の人のChatGPTを学習するためにも使用されます。

Model improvement

Improve the model for everyone



Allow your content to be used to train our models, which makes ChatGPT better for you and everyone who uses it. We take steps to protect your privacy. [Learn more](#)

Done

How to use ChatGPT



プロンプトとは？

LLMに特定の応答を生成させるための入力指示。

- プロンプトの種類とその例

- 質問形式

ユーザーがAIに対して情報や解答を求める際に使用する形式

✓ 「シリコンのバンドギャップはどれくらいですか？」

- 命令形式

ユーザーが特定のタスクの実行をAIに指示する形式

✓ 「分子動力学シミュレーションの基本原理を説明してください。」

- 情報提供形式

ユーザーが自身の状況や考えを共有し、それに基づいてAIからの提案や助言を求める形式

✓ 「新しい材料の熱伝導率を予測するための手法について教えてください。」

How to use ChatGPT



カスタム指示機能

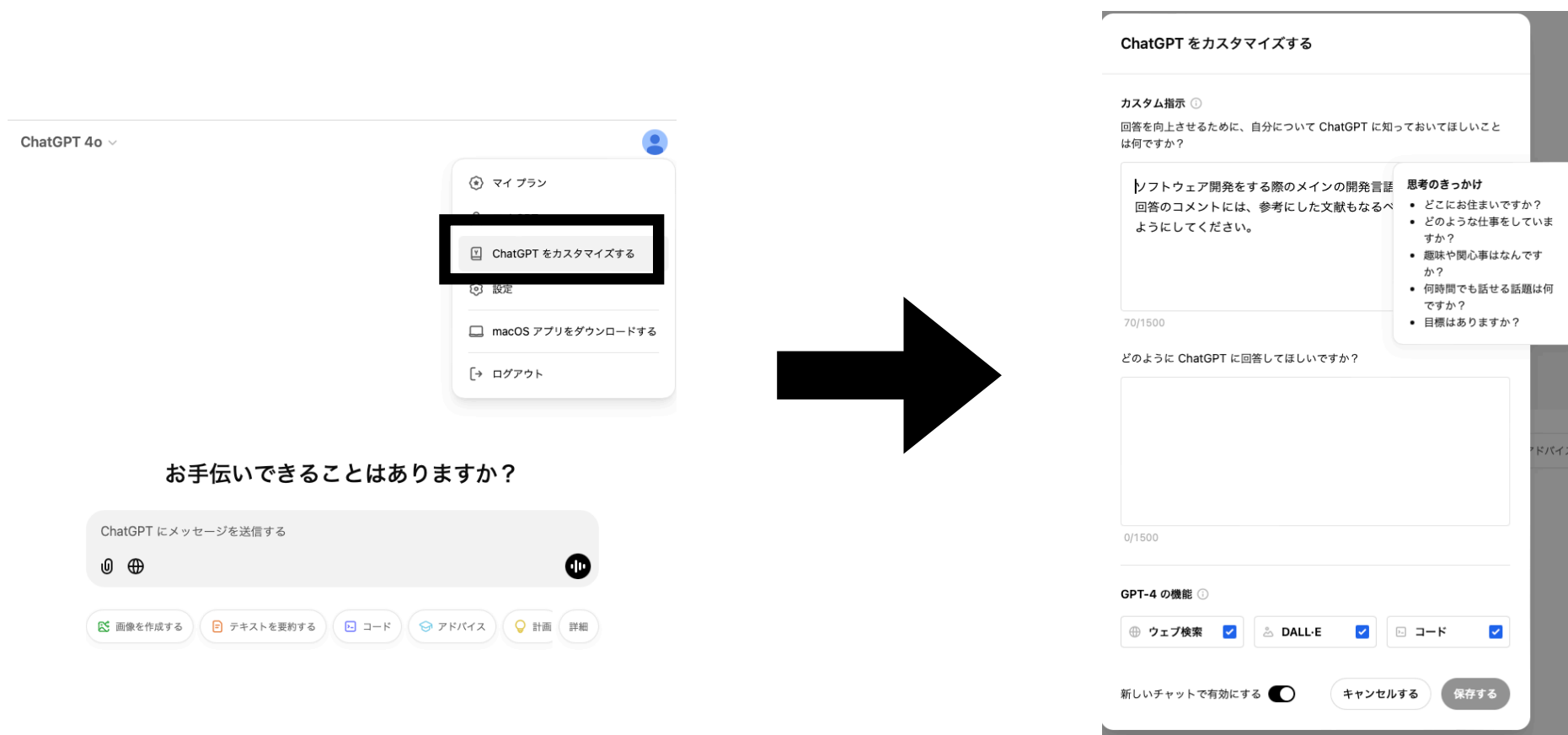
- ユーザーがChatGPTに対して自身の情報や希望する応答スタイルを事前に設定することで、よりパーソナライズされた対話を実現する機能
- 特徴
 - **ユーザー情報の提供:** 自身の職業、興味、専門分野などを設定
 - **応答スタイルの指定:** フォーマル、カジュアル、詳細な説明、簡潔な回答など、希望する応答の形式やトーンを指定
- 注意点:
 - **情報の更新:** ニーズや状況に応じて適宜変更する必要あり。
 - **プライバシー:** 提供する情報には個人情報を含めないよう注意し、プライバシーを保護する。

ChatGPTの環境設定(5)

カスタム指示機能



- 通常使う使用条件はプリコンディションとして設定



アカウントのロゴをクリック後に、"ChatGPTをカスタマイズする" を選択する。

ChatGPTの環境設定(6)

カスタム指示の設定例



- 自分についてChatGPTに知っておいてもらいたいこと
 - 「私は計算物質科学の研究者で、主に第一原理計算と分子動力学シミュレーションを専門としています。現在、半導体材料の電子構造解析に取り組んでおり、PythonとC++を使用しています。」
 - 効果：ユーザーの専門分野や使用言語を理解し、関連する情報やアドバイスを提供しやすくなる。
- どのようにChatGPTに回答してもらいたいのか？
 - 「技術的な質問には、専門用語を適切に使用し、詳細かつ正確な情報を提供してください。コード例を示す際には、Pythonを使用し、適切なコメントを付けてください。」
 - 効果：ユーザーの期待に沿った専門的で詳細な応答を提供するよう調整される。

How to use ChatGPT



メモリ機能

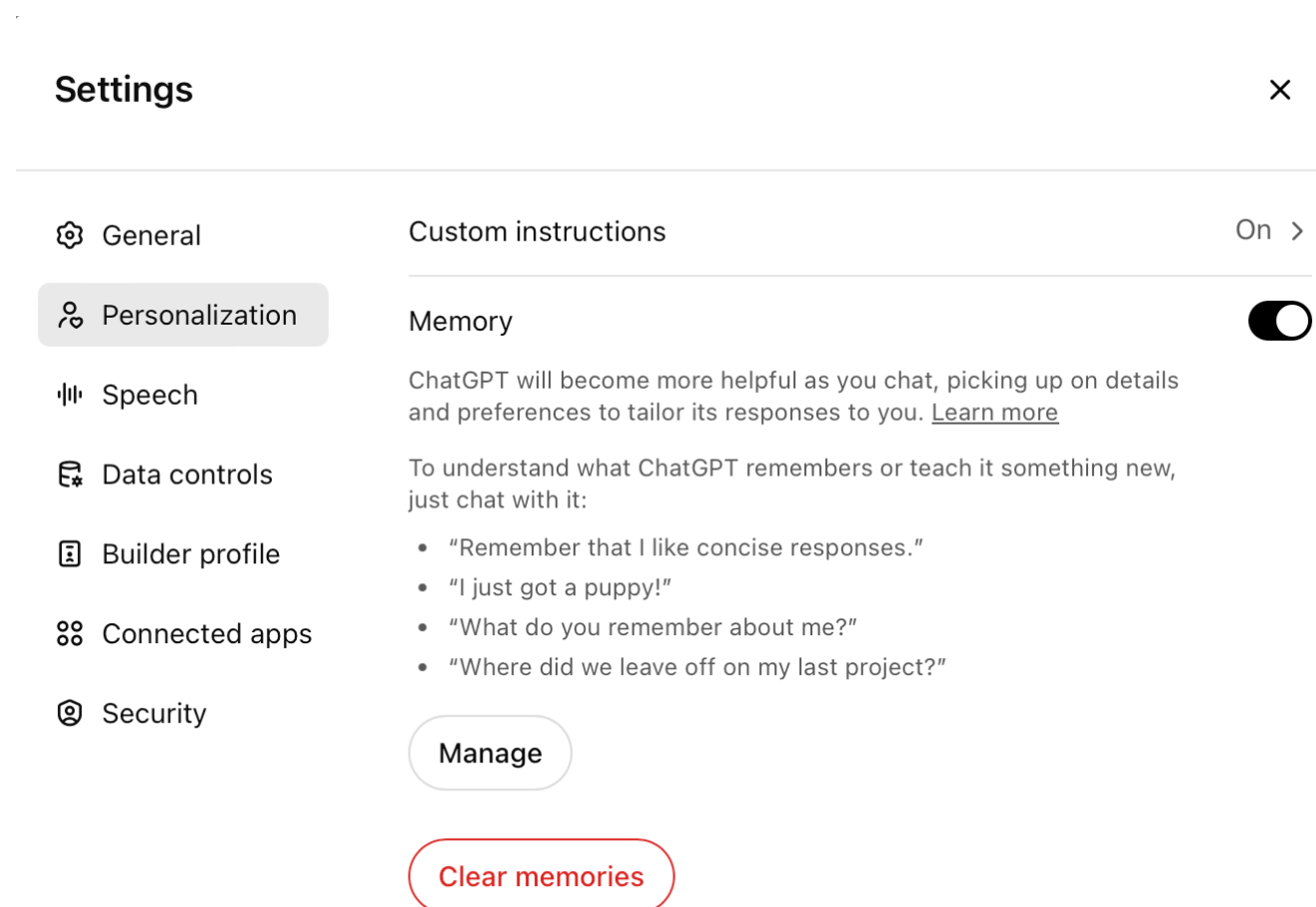
- ユーザーとの対話内容や設定を記憶し、よりパーソナライズされた応答を提供
- 特徴
 - **会話の記憶**: 過去の対話内容を保持し、関連する情報を次回以降の会話で活用。
 - **設定の保存**: ユーザーの好みや指示を保存し、以降の応答に反映。
- カスタム指示との違い:
 - **カスタム指示**: ユーザーが明示的に設定する情報や応答のスタイル。
 - **メモリ機能**: 会話を通じて自動的に学習・記憶する情報。
- 注意点:
 - **プライバシー**: メモリ機能はユーザーの情報を記憶するため、個人情報の取り扱いに注意が必要です。
 - **データの使用**: メモリ機能で収集されたデータがモデルのトレーニングに使用される場合があります。

How to use ChatGPT

メモリ機能



- ChatGPTの設定メニューを開く。
- 「Personalization」セクションで機能をオンまたはオフに切り替える。
- 必要に応じて、「Manage」から保存されたメモリを管理・削除する。



How to use ChatGPT



質問をする際に気をつけること

1. ChatGPTは万能ではない

- 生成される回答は必ずしも正確・完全ではないため、参考情報として活用。

2. 質問内容を明確かつ具体的にする

- 例: 「材料の熱伝導率を教えてください。」ではなく、「銅の熱伝導率はどれくらいですか？」

3. 背景情報や状況を明示する

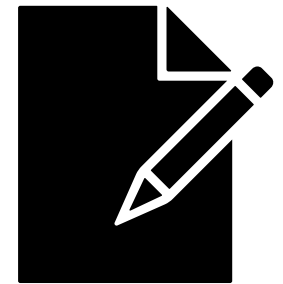
- 例: 「学生向けのプレゼン資料を作成しています。銅の熱伝導率について簡潔に説明してください。」

4. 複雑な質問は分割する

- 例: 「銅の電子構造とその熱伝導特性への影響を詳しく教えてください。」
→ 「銅の電子構造を詳しく説明してください。」 → 「銅の熱伝導特性を説明してください。」
→ 「銅の電子構造がその熱伝導特性にどのように影響するかを教えてください。」

5. 期待する回答の形式や例を提示する

- 例: 「箇条書きで銅の特性を3つ挙げてください。」



GitHub Copilotの環境設定

利用前の準備

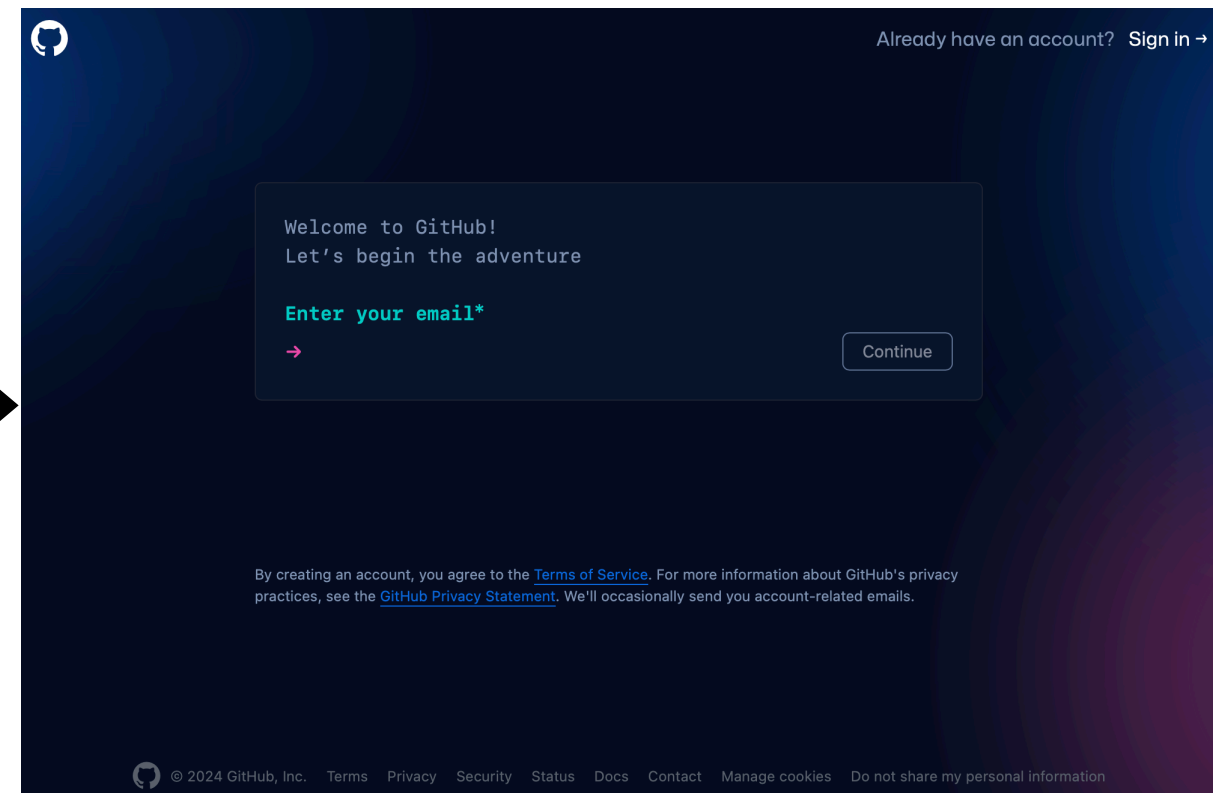
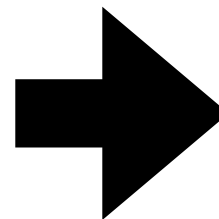
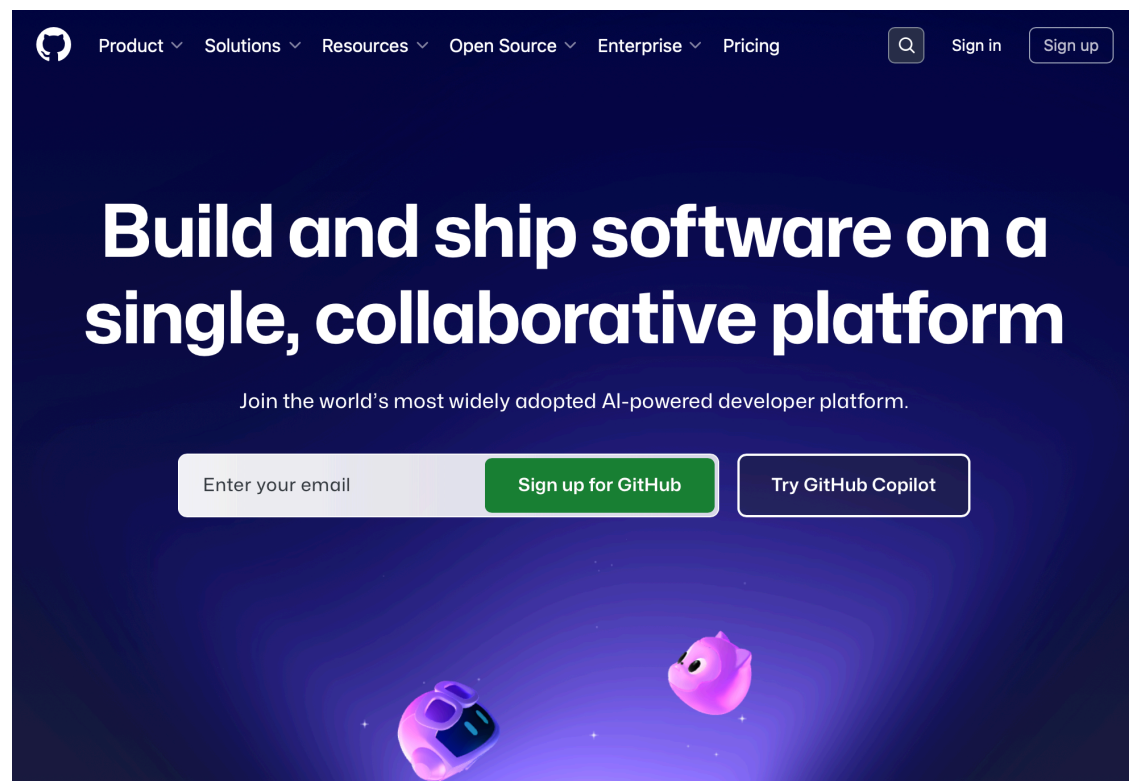


1. GitHubアカウントの作成

GitHubの公式サイト (<https://github.com/>) でアカウントを作成

2. Copilotのサブスクリプション

個人利用の場合、GitHub Copilotのサブスクリプションを取得

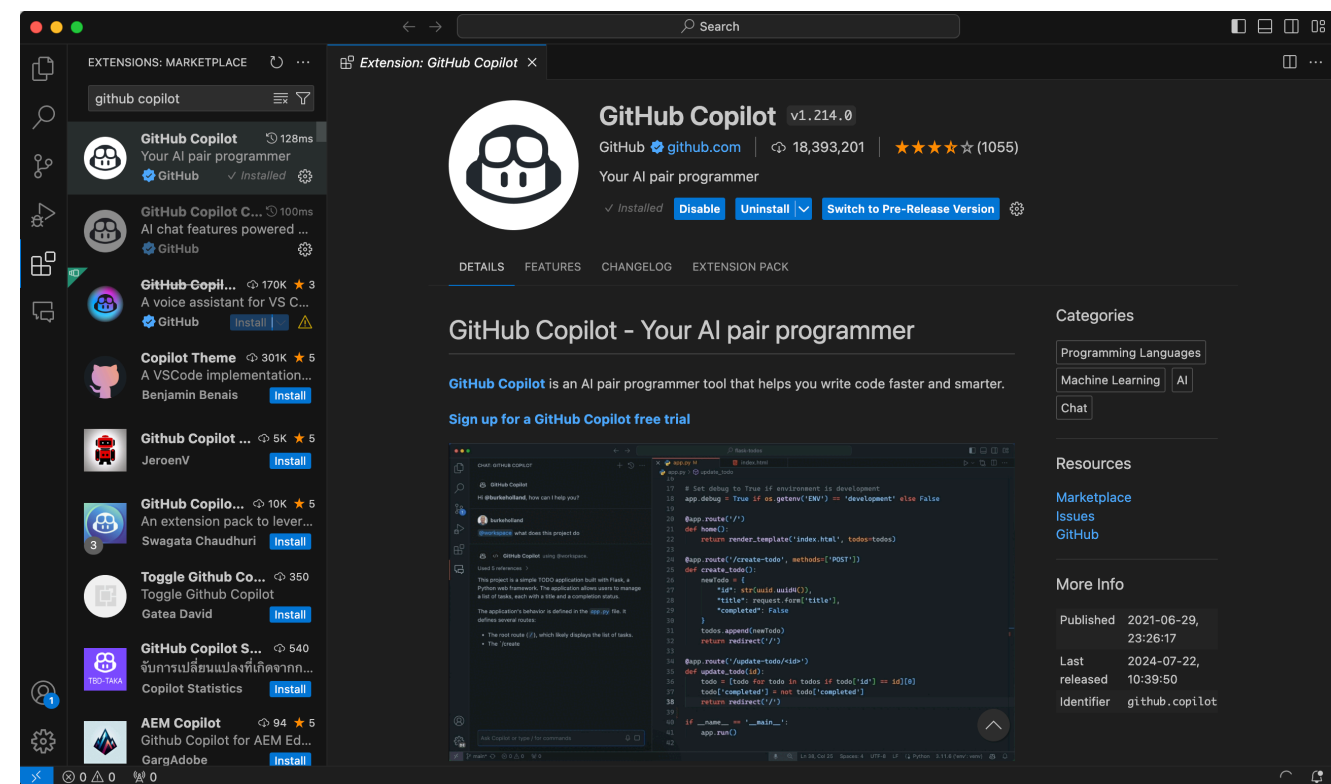
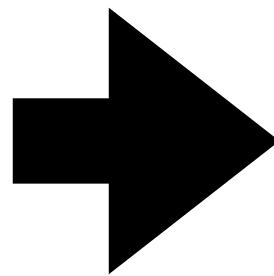
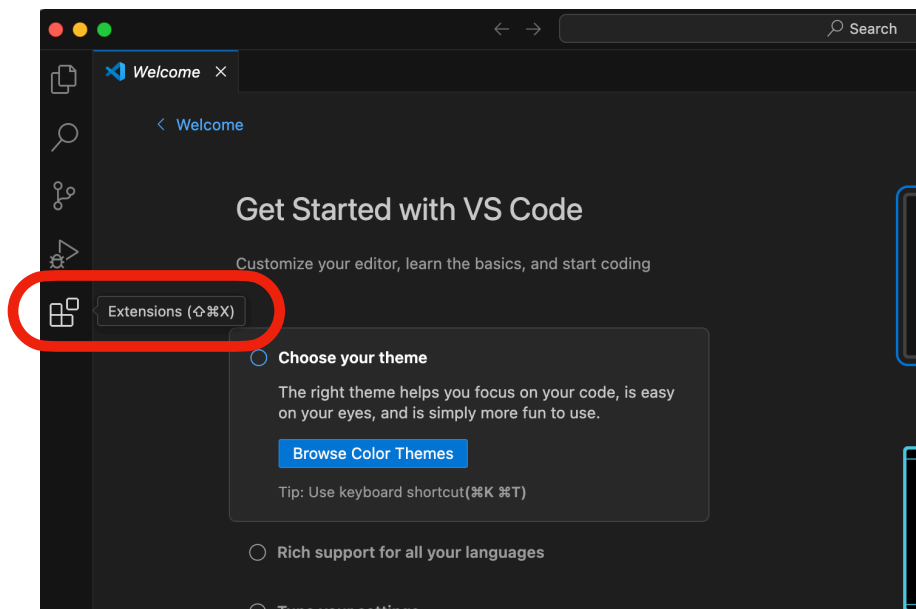


How to Set Up and Use GitHub Copilot

インストール～Visual Studio Code



1. Visual Studio Codeを開き、ウィンドウのサイドバーにあるアクティビティバーの拡張機能アイコンをクリックして、拡張機能ビューに移動。
2. 「GitHub Copilot」を検索し、拡張機能をインストール。
3. インストール後、再起動して有効化



How to Set Up and Use GitHub Copilot



サインインと設定

1. プロンプトが表示されたらGitHubにサインインし、GitHub Copilotがアカウントにアクセスすることを許可する。
2. 拡張機能のオプションで設定を調整し、自分の好みに合わせて提案をカスタマイズする。
 - ✓ 提案の受け入れ方法
Tabキーで提案を受け入れるなど、ショートカットキーを設定。
 - ✓ 言語別の有効化/無効化
特定のプログラミング言語でのCopilotの動作を設定。

How to Set Up and Use GitHub Copilot

プライバシーとセキュリティ設定 (1)



パブリックコードに一致する候補の管理
オリジナリティを重視する場合や、ライセンスの懸念がある場合は、パブリックコードに一致する提案をブロックすることが可能

Yoshimi (k-yoshimi)
Your personal account [Switch settings context](#)

Public profile
Account
Appearance
Accessibility
Notifications

Access
Billing and plans
Emails
Password and authentication
Sessions
SSH and GPG keys
Organizations
Enterprises
Moderation

Code, planning, and automation
Repositories
Codespaces
Packages
Copilot
Pages
Saved replies
Security
Code security
Integrations
Applications

GitHub Copilot

GitHub Copilot Individual is active for your account
You currently have an active [Copilot Individual subscription](#).
Get started by installing the extension in your preferred IDE.

[Copilot in your IDE](#) [Copilot in the CLI](#) [Chat in GitHub Mobile](#) [More features](#)

Copilot policies

Copilot in github.com Preview You can use Copilot Chat in github.com, Copilot for pull requests, and all of the preview features. Learn more about Copilot in github.com	Enabled
Copilot in the CLI You can use GitHub Copilot for assistance in terminal .	Enabled
Copilot Chat in the IDE You can use GitHub Copilot Chat in the IDE to explain a piece of code, propose bug fixes, or generate unit tests in a chat interface from your editor.	Enabled
Copilot Chat in GitHub Mobile You can use Copilot Chat in GitHub Mobile personalized to a codebase.	Enabled

Suggestions matching public code (duplication detection filter)
Copilot can allow or block suggestions matching public code. [Learn more about code suggestions.](#)

☒ **Allow GitHub to use my code snippets from the code editor for product improvement**
Allow GitHub, its affiliates and third parties to use my code snippets to research and improve GitHub Copilot.

Blocked
Allowed
GitHub Copilot will show suggestions matching public code.

☒ **Blocked**
GitHub Copilot won't show suggestions matching public code.

GitHubの個人設定で、
「Suggestions matching public code」の項目から
「Allow」または
「Block」を選択。

How to Set Up and Use GitHub Copilot

プライバシーとセキュリティ設定 (2)



データ収集の許可/禁止

プロンプトや提案のデータ収集を許可するかどうかを設定可能

Copilot policies

Copilot in github.com Preview	Enabled
You can use Copilot Chat in github.com, Copilot for pull requests, and all of the preview features. Learn more about Copilot in github.com	
Copilot in the CLI	Enabled
You can use GitHub Copilot for assistance in terminal .	
Copilot Chat in the IDE	Enabled
You can use GitHub Copilot Chat in the IDE to explain a piece of code, propose bug fixes, or generate unit tests in a chat interface from your editor.	
Copilot Chat in GitHub Mobile	Enabled
You can use Copilot Chat in GitHub Mobile personalized to a codebase.	
Suggestions matching public code (duplication detection filter)	Allowed ▾
Copilot can allow or block suggestions matching public code. Learn more about code suggestions .	
☒ Allow GitHub to use my code snippets from the code editor for product improvements *	
Allow GitHub, its affiliates and third parties to use my code snippets to research and improve GitHub Copilot suggestions, related models and product features. More information in About GitHub Copilot privacy .	
Copilot access to Bing	Disabled ▾
Copilot can answer questions about new trends and give improved answers. See Microsoft Privacy Statement .	
🔒 Your organization(s) have not enabled use of this feature.	
Anthropic Claude 3.5 Sonnet in Copilot Preview	Select an option ▾
You can use the latest Claude 3.5 Sonnet model. Learn more about how GitHub Copilot serves Claude 3.5 Sonnet .	

GitHubの個人設定で、
「Allow GitHub to use my code snippets from the code editor for product improvements」のチェックボックスをON, OFFにすることで選択。

まとめ

計算物質科学におけるLLMの活用～基礎編： LLMの基礎と環境構築

- ・ LLMの基礎知識 ～ 言語モデルの仕組みとその進化
 - ・ 言語モデルからTransformerまで説明
- ・ LLMツール(ChatGPTとGitHub Copilot)の概要
 - ・ 各種ツールの使い方・料金プランなどについて説明
- ・ LLMツール(ChatGPTとGitHub Copilot)の環境構築
 - ・ 導入方法および各種設定について説明
- ・ 応用編ではChatGPT、GitHub Copilotに関する実際の応用例について説明します。