埋め込みモデルとベクトル検索

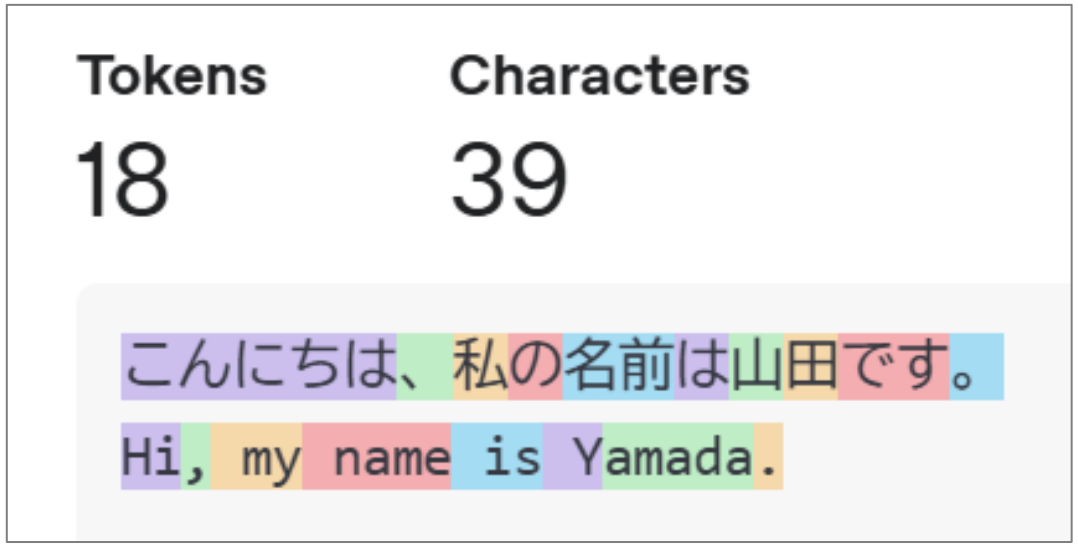
AI-102

補足資料

トークン

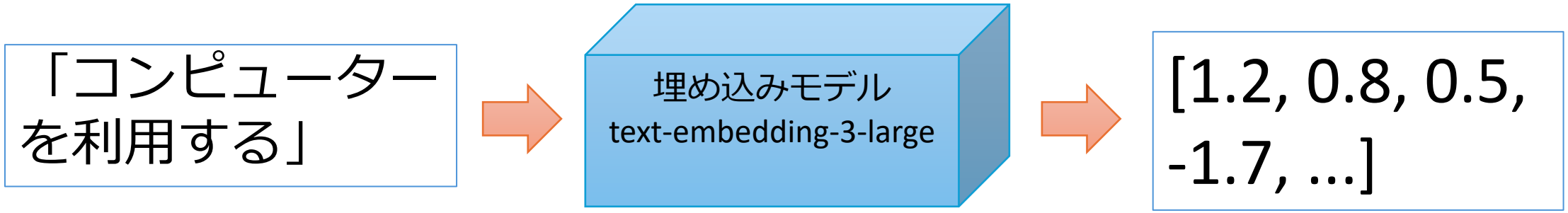
- 生成AIモデルがテキストを処理する際の処理単位・料金単位
- 入力されたテキストはトークンに分解される
- 出力では、あるトークンの次に出現する確率が最も高いトークンを予測しながら、新しいテキストを生成する
  - "日本の首都は？" : ？の部分に出現する確率が高いトークンは「東京」
- 入力トークンと出力トークンの量に比例した料金がかかる
- https://platform.openai.com/tokenizer で、トークンを確認できる

こんにちは、私の名前は山田です。
Hi, my name is Hiromichi.



埋め込み (embeddings) とは？

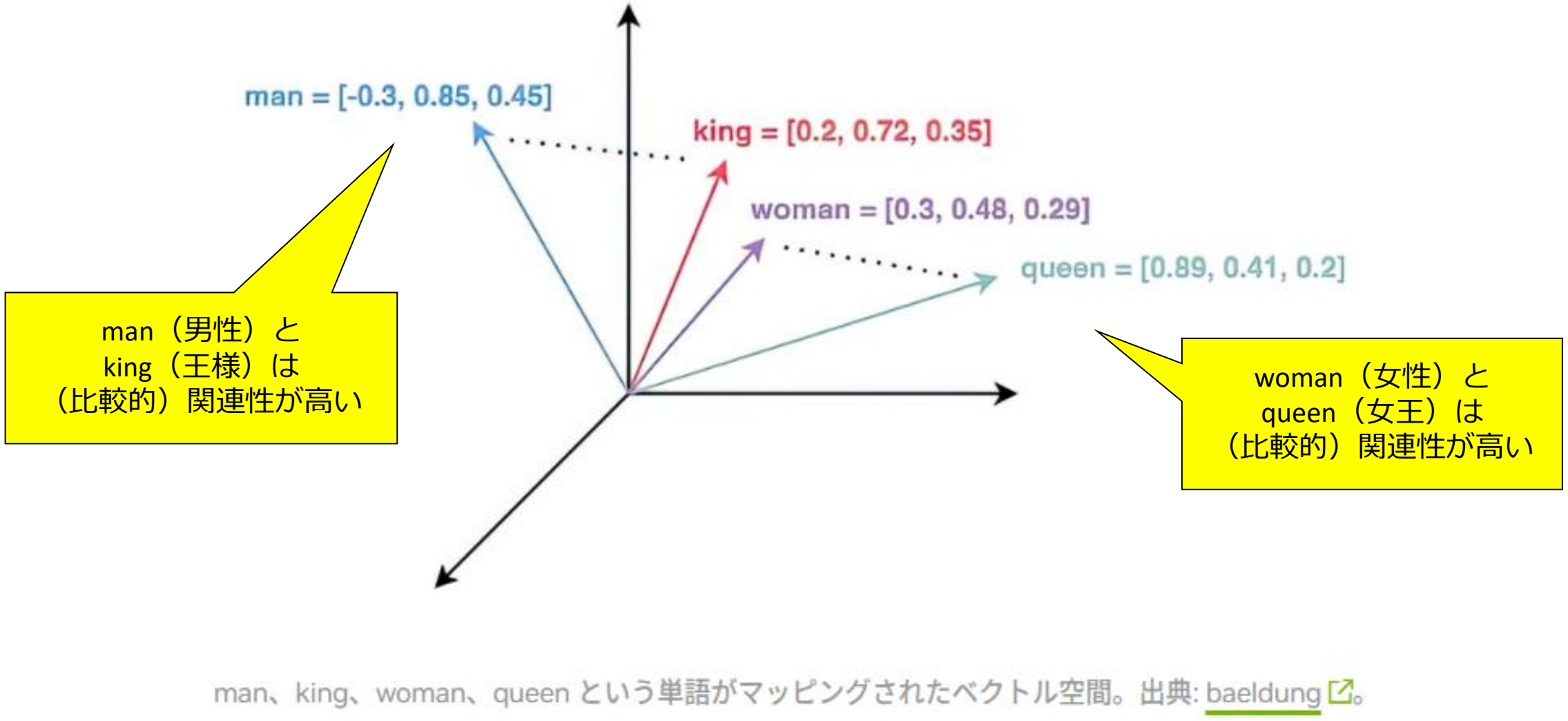
- 「埋め込み」は、機械学習モデルと関連アルゴリズムで利用される特別な形式のデータ表現
- 各「埋め込み」は浮動小数点数のベクトル（数値の配列）で表現される
  - ベクトルの例: [1.2, 0.8, 4.5, -1.7, ...]
- 「text-embedding-3-large」などの「埋め込みモデル」を使用して、任意のテキストをベクトルに変換できる。



※実際には3072次元のベクトルとなる
（つまり、3072個の浮動小数点数となる）

Vector embeddings - OpenAI API

「埋め込み」によって得られるベクトルのイメージ。
「man」「woman」「king」「queen」などの単語を3次元のベクトルとして表現した例。
関連性の高い単語はベクトルが似ている。
※実際の「埋め込み」は3072次元といった多次元データである



man、king、woman、queen という単語がマッピングされたベクトル空間。出典: baeldung。

ベクトル データベースとは何か? その仕組みとは? | NVIDIA

（参考）「埋め込み」と「ベクトル」の違い

- 埋め込み（embeddings）
  - テキスト（等のデータ）の特徴を数値化したもの
  - 似たようなテキスト（等のデータ）は同じような「埋め込み」に変換される
  - OpenAIの text-embedding-3-large などのモデルで、テキストの埋め込みを計算できる
  - Azure AI VisionのAPIを使用して、画像の埋め込みを計算できる
  - MetaのWav2Vecを使用して、音声の埋め込みを計算できる
  - **埋め込みはベクトルとして表現・記録される**
- ベクトル、ベクター（vector）
  - 平面や空間での有向線分（長さと向きを持つ線分）
  - コンピューターに記録された、数値の1次元配列
    - CPUのSIMD（Single Instruction, Multiple Data）命令を使って高速に計算できる
    - データベースによってはベクトルを扱うためのVECTOR型などを持つ場合がある

マルチモーダル埋め込みの概念 - Image Analysis 4.0 - Azure AI services | Microsoft Learn

Wav2vec

（参考）「埋め込み」と「ベクトル」の違い

- 「埋め込み」は「ベクトル」として表現されるが、
  必ずしも「ベクトル」＝「埋め込み」とは限らない。
- たとえばゲームの中であるキャラクターが動いているとき、
  その動きの向きや速度を表すのにも「ベクトル」が使用できる

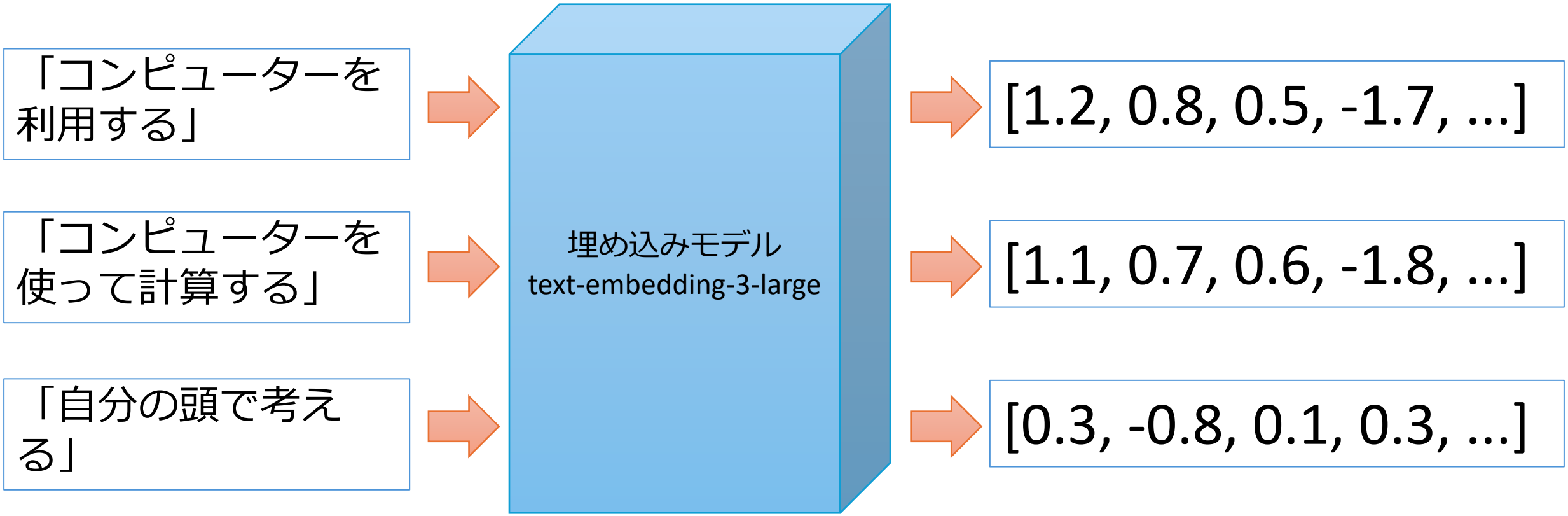
- 「ベクトル」はいろいろなもの（埋め込みやゲームキャラの動き）を表すのに使用できる
- ただし、AIや機械学習の文脈では、「ベクトル」は「埋め込み」によって得られたデータを表す言葉として使用されることが多い

「埋め込み」の計算

- azure_openai.create_embeddings関数を使用すると、テキストを埋め込み（ベクトル）に変換できる
- 戻り値の型は real[] 型である

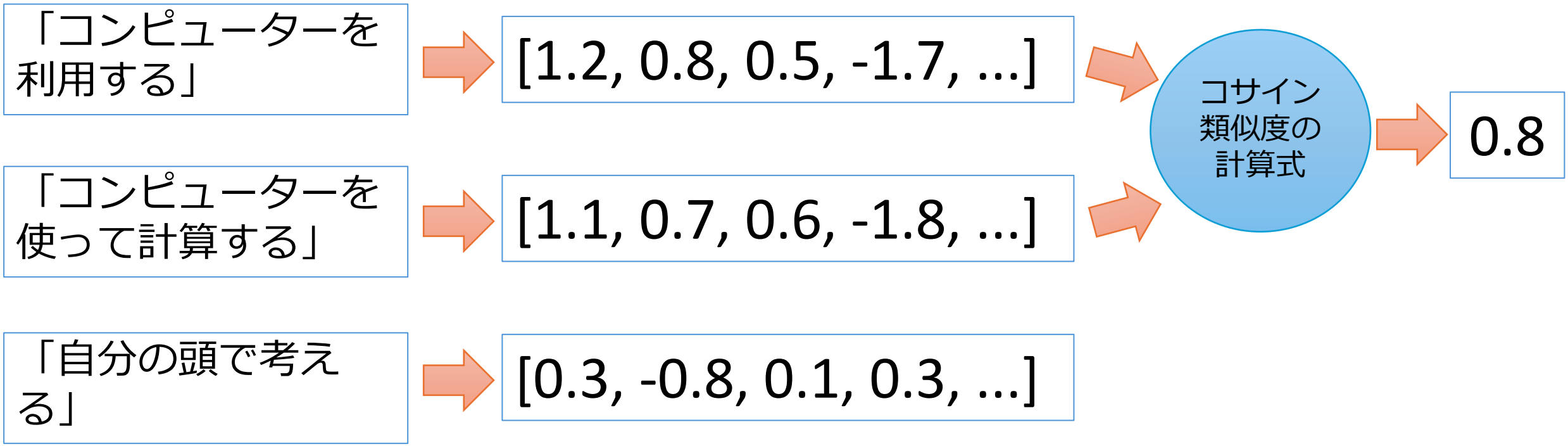
「埋め込み」の特徴

- 意味が似ている2つの文章では、それらの埋め込み（ベクトル）も似たものとなる



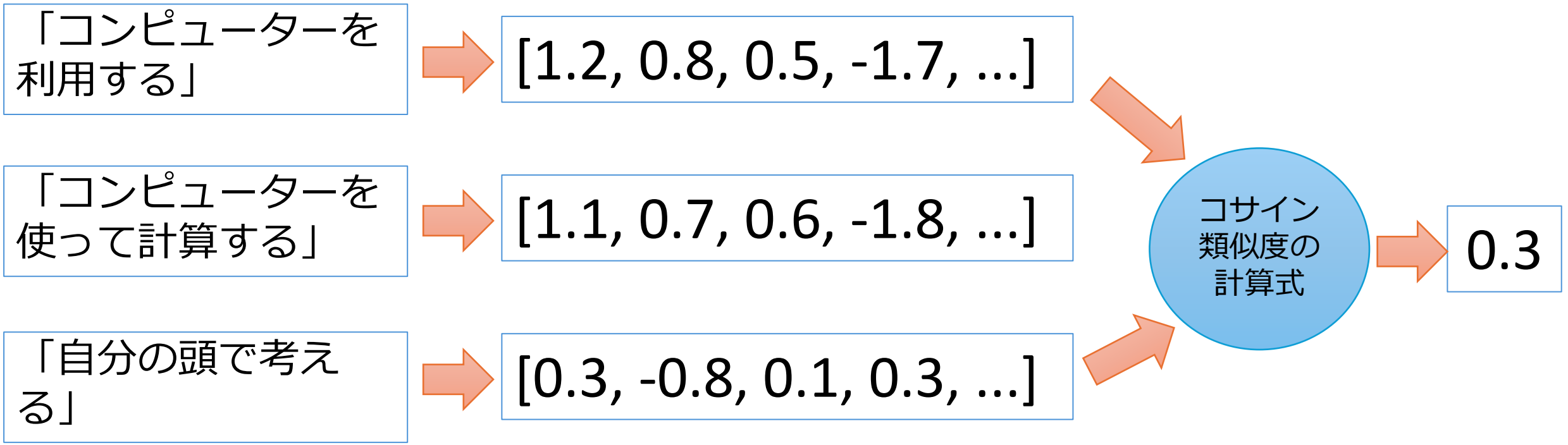
コサイン類似度

- 2つのベクトルから「コサイン類似度」（-1～1）を計算することで、2つのデータがどの程度「似ているか」を1つの数で表現できる
- 2つのベクトル（2つのテキスト）が「似ている」場合、「コサイン類似度」は1に近くなる



コサイン類似度

- 2つのベクトルから「コサイン類似度」（-1～1）を計算することで、2つのデータがどの程度「似ているか」を1つの数で表現できる
- 2つのベクトル（2つのテキスト）が「似ている」場合、「コサイン類似度」は1に近くなる



コサイン類似度の計算式と計算例

$$\cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|}$$

ベクトル A = [1, 2, 3]

ベクトル B = [4, 5, 6]

① 内積（A・B）を求める

$$A \cdot B = (1 \times 4) + (2 \times 5) + (3 \times 6) = 4 + 10 + 18 = 32$$

---

② ノルム（長さ）を求める

$$\|A\| = \sqrt{1^2 + 2^2 + 3^2} = \sqrt{14}$$

$$\|B\| = \sqrt{4^2 + 5^2 + 6^2} = \sqrt{77}$$

---

③ 式に当てはめてコサイン類似度を計算

$$\cos(\theta) = \frac{32}{\sqrt{14} \cdot \sqrt{77}} = \frac{32}{\sqrt{1078}} \approx 0.9746$$

cosine_similarity 関数を使用して2つのテキストを比較してみる

```
15  SELECT cosine_similarity('コンピューターを利用する', 'コンピューターを使って計算する')
16
```

Data Output　Messages　Notifications

	cosine_similarity double precision
1	0.8070400838279789

```
16  SELECT cosine_similarity('コンピューターを利用する', '自分の頭で考える')
17
```

Data Output　Messages　Notifications

	cosine_similarity double precision
1	0.34848690936047944

（参考）コサイン距離

- 2つのベクトル vec1 と vec2 から「コサイン距離」を計算する **<=>** 演算子と、 **cosine_distance**関数 がある。
  - `vec1 <=> vec2`
  - または **cosine_distance**`(vec1, vec2)`

- 1から「コサイン距離」を引くと「コサイン類似度」が得られる。
  - `1 - (vec1 <=> vec2)`
  - または `1 -` **cosine_distance**`(vec1, vec2)`

- なお「コサイン類似度」を計算する関数はPostgreSQL本体・「azure_ai拡張機能」・「pgvector拡張機能」には定義されていないが、「コサイン距離」から簡単に計算できる。

ベクトル検索

- 従来のデータ検索では、ユーザーが「検索キーワード」を指定し、そのキーワードを含む文書などを検索していた。
- 検索キーワードを正しく入力しなければ、目的のテキストを発見することができなかった。

- 埋め込み（ベクトル）とコサイン類似度を使用することで、ユーザーが指定したテキストに意味的に近い（関連性が高い）テキストを検索することが可能となった。
- これをベクトル検索という。

ベクトル検索の動作原理

出張で使用できる予算の上限は？

埋め込みモデル

[1.2, 0.7, 0.6, -1.7, ...]

コサイン類似度の計算式

テキスト	埋め込み	コサイン類似度
今年の目標・・・	[0.5, -1.1, -0.3, 0.9, ...]	0.2
平素は格別のご高配を・・・	[-0.3, 0.4, 0.1, 0.5, ...]	
出張規程【2025】・・・	[1.5, 0.3, -1.1, 0.1, ...]	
会議の議事録・・・	[0.5, -1.1, -0.3, 0.9, ...]	
製品Xのマニュアル・・・	[0.2, 0.3, -1.2, -0.5, ...]	

ベクトル検索の動作原理

出張で使用できる予算の上限は？

埋め込みモデル

[1.2, 0.7, 0.6, -1.7, ...]

コサイン類似度の計算式

テキスト	埋め込み	コサイン類似度
今年の目標・・・	[0.5, -1.1, -0.3, 0.9, ...]	0.2
平素は格別のご高配を・・・	[-0.3, 0.4, 0.1, 0.5, ...]	0.1
出張規程【2025】・・・	[1.5, 0.3, -1.1, 0.1, ...]	
会議の議事録・・・	[0.5, -1.1, -0.3, 0.9, ...]	
製品Xのマニュアル・・・	[0.2, 0.3, -1.2, -0.5, ...]	

ベクトル検索の動作原理

出張で使用できる予算の上限は？

埋め込みモデル

[1.2, 0.7, 0.6, -1.7, ...]

コサイン類似度の計算式

テキスト	埋め込み	コサイン類似度
今年の目標・・・	[0.5, -1.1, -0.3, 0.9, ...]	0.2
平素は格別のご高配を・・・	[-0.3, 0.4, 0.1, 0.5, ...]	1
出張規程【2025】・・・	[1.5, 0.3, -1.1, 0.1, ...]	0.9
会議の議事録・・・	[0.5, -1.1, -0.3, 0.9, ...]	
製品Xのマニュアル・・・	[0.2, 0.3, -1.2, -0.5, ...]	

ベクトル検索の動作原理

出張で使用できる予算の上限は？

埋め込みモデル

[1.2, 0.7, 0.6, -1.7, ...]

コサイン類似度の計算式

テキスト	埋め込み	コサイン類似度
今年の目標・・・	[0.5, -1.1, -0.3, 0.9, ...]	0.2
平素は格別のご高配を・・・	[-0.3, 0.4, 0.1, 0.5, ...]	0
出張規程【2025】・・・	[1.5, 0.3, -1.1, 0.1, ...]	0.9
会議の議事録・・・	[0.5, -1.1, -0.3, 0.9, ...]	0.3
製品xのマニュアル・・・	[0.2, 0.3, -1.2, -0.5, ...]	

ベクトル検索の動作原理

出張で使用できる予算の上限は？

埋め込みモデル

[1.2, 0.7, 0.6, -1.7, ...]

コサイン類似度の計算式

テキスト	埋め込み	コサイン類似度
今年の目標・・・	[0.5, -1.1, -0.3, 0.9, ...]	0.2
平素は格別のご高配を・・・	[-0.3, 0.4, 0.1, 0.5, ...]	0.1
出張規程【2025】・・・	[1.5, 0.3, -1.1, 0.1, ...]	.9
会議の議事録・・・	[0.5, -1.1, -0.3, 0.9, ...]	0.3
製品xのマニュアル・・・	[0.2, 0.3, -1.2, -0.5, ...]	0.2

ベクトル検索の動作原理

出張で使用できる予算の上限は？

埋め込みモデル

[1.2, 0.7, 0.6, -1.7, ...]

テキスト	埋め込み	コサイン類似度
今年の目標・・・	[0.5, -1.1, -0.3, 0.9, ...]	0.2
平素は格別のご高配を・・・	[-0.3, 0.4, 0.1, 0.5, ...]	0.1
出張規程【2025】・・・	[1.5, 0.3, -1.1, 0.1, ...]	0.9
会議の議事録・・・	[0.5, -1.1, -0.3, 0.9, ...]	0.3
製品xのマニュアル・・・	[0.2, 0.3, -1.2, -0.5, ...]	0.2

ベクトル検索の動作原理

出張で使用できる予算の上限は？

埋め込みモデル

[1.2, 0.7, 0.6, -1.7, ...]

テキスト	埋め込み	コサイン類似度
今年の目標・・・	[0.5, -1.1, -0.3, 0.9, ...]	0.2
平素は格別のご高配を・・・	[-0.3, 0.4, 0.1, 0.5, ...]	0.1
出張規程【2025】・・・	**[1.5, 0.3, -1.1, 0.1, ...]**	**0.9**
会議の議事録・・・	[0.5, -1.1, -0.3, 0.9, ...]	0.3
製品Xのマニュアル・・・	[0.2, 0.3, -1.2, -0.5, ...]	0.2

参考: ベクトルの最近傍探索アルゴリズム

- 実際には、検索のたびに、全てのベクトルとの比較（類似度計算）を行うと、時間がかかりすぎる。
- 実用的なシステム（ベクトルデータベース）では、指定したベクトルに近いベクトルを素早く探し出すために、より効率的なアルゴリズムとデータ構造（インデックス）が使用される
  - ANN（Approximate Nearest Neighbor, 近似最近傍探索）: 近似的な方法でベクトルを高速に検索するアルゴリズム。数百万件といった大量のベクトルを高速に処理できる。
  - HNSW（Hierarchical Navigable Small World, 階層的ナビゲーブルスモールワールド）: ANN検索の中でもよく使用されるアルゴリズム。階層的なグラフ構造を使って近いベクトルを効率的に探す。事前に階層インデックスを構築する。
- Azure Database for PostgreSQLでもHNSWなどが利用できる

興味がある方は
こちらをチェック

https://learn.microsoft.com/ja-jp/azure/postgresql/flexible-server/how-to-optimize-performance-pgvector

まとめ

埋め込みモデル	「text-embedding-3-large」などの「埋め込みモデル」を使用して、テキストなどのデータの「埋め込み」（ベクトル）を計算できる。
埋め込み （embeddings）	機械学習モデルと関連アルゴリズムで利用される浮動小数点数の配列（ベクトル）であり、元のデータの特徴を含む。
コサイン類似度	2つのデータの「埋め込み」（ベクトル）をそれぞれ計算し、そのコサイン類似度を計算することで、2つのデータの類似度（-1～1）を求めることができる。1に近いほど類似度が高い（意味的に近い）。
ベクトル検索	コサイン類似度などを使用して、類似度が高い（意味的に近い）データを検索する方法。
最近傍探索 アルゴリズム	ベクトルデータベースなどでは、HNSW（Hierarchical Navigable Small World）などの最近傍探索（近似最近傍探索）アルゴリズムとデータ構造を使用して、高速なベクトル検索を実現している。