OpenSearch を使用したベクトル検索

Shunsuke Goto (後藤駿介)

アマゾンウェブサービスジャパン合同会社
プロトタイプエンジニアリング本部
Prototyping Engineer

ベクトル検索とは？

ベクトル：テキスト、画像、音声のようなメディアの情報を N 次元の数値列に符号化したもの

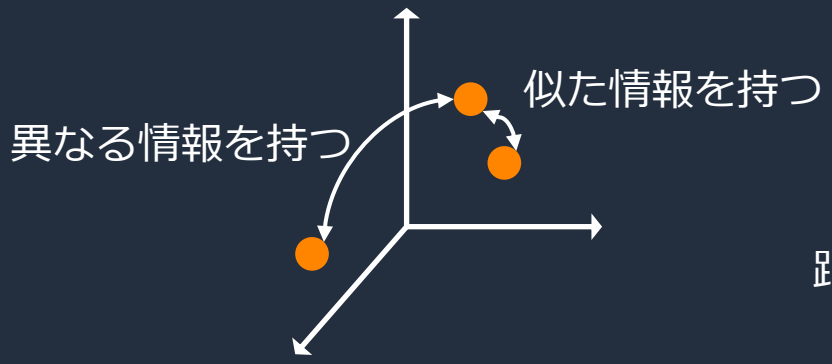
テキスト、画像などのメディア

ML モデル

[1.23, 0.21, 0.08, -1.21, …, 4.23]

[0.12, -0.89, 2.12, 0.05, …, -3.21]

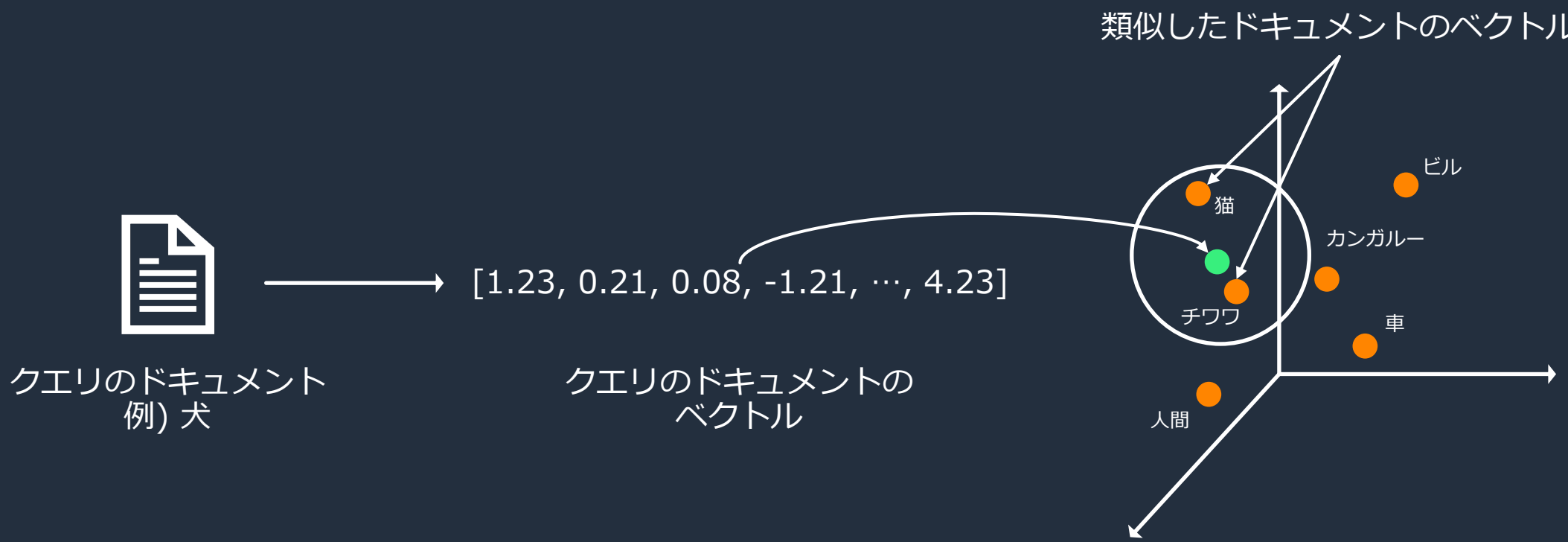
ベクトル



距離の近いベクトル同士は似ている情報を持ち、
距離の遠いベクトル同士は異なる情報を持つように
ML モデルは学習される

ベクトル検索とは？

ベクトル検索：入力のクエリに近いベクトルを取得する検索



ベクトル検索の代表的なユースケース

レコメンド

例）商品情報をベクトル化し、
ユーザーが閲覧した商品履歴を
元に、おすすめの商品を出す

画像検索

例）画像をベクトル化し、
入力した画像と類似した画像を
取得する

生成 AI アプリケーション
(例: RAG)

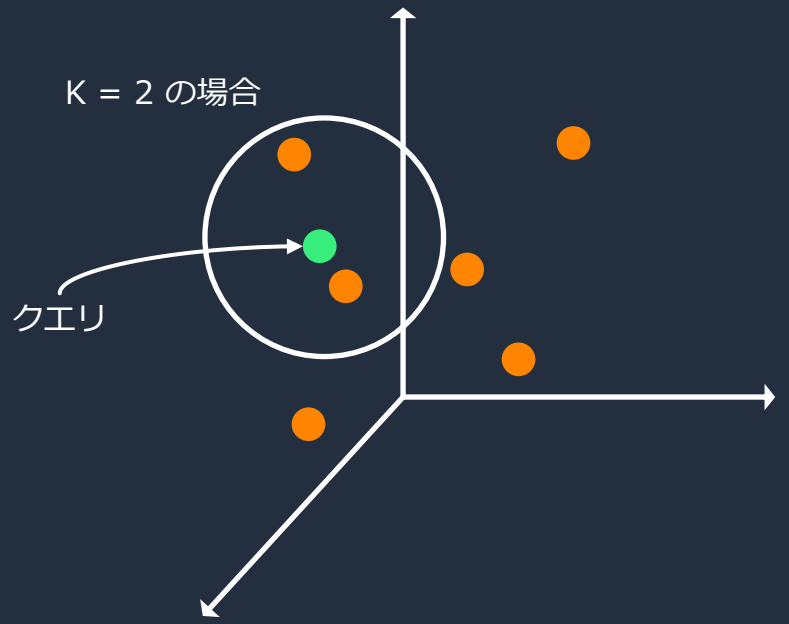
例）社内のドキュメントをベクトル
化し、ユーザーの質問に関連するド
キュメントをベクトル検索で取得。
それらをコンテキストとして LLM
に渡してユーザーの質問に回答する

ベクトル検索の仕組み (k-NN 検索)

ベクトル検索では、k-NN (k-最近傍) 検索と呼ばれるアルゴリズムが使用される

k-NN 検索は、入力のクエリに対して、最も近い* k 個のベクトルを取得するアルゴリズム

* ベクトル同士の近さは、距離メトリクス (ユークリッド距離、コサイン距離 など) に依存する



k-NN 検索と近似 k-NN 検索

正確に k-NN アルゴリズムを実行しようとすると、データ量に比例して計算量が増大する

近似 k-NN 検索 (Approximate k-NN) を使用することで、データ量が数千万、数億のオーダーまで増大しても、高速で処理することができる

近似 k-NN アルゴリズムの例

- HSNW (Hierarchical Navigable Small Worlds) アルゴリズム
- IVF (Inverted File) アルゴリズム

大規模なデータに対してベクトル検索を行うには、近似 k-NN 検索が必須

➡ OpenSearch を使用することで、これらのアルゴリズムを使用して、大規模なデータに対して高速にベクトル検索をすることが可能

OpenSearch

コミュニティ主導・Apache 2.0 ライセンスのオープンソース検索・分析スイート

データストア、検索エンジンの OpenSearch、可視化、UI ツールの OpenSearch Dashboards から構成されている

セキュリティ、パフォーマンス分析、機械学習、k-NN など様々なプラグインによる機能拡張が可能

OpenSearch の利用方法

1

JSON 形式の**ドキュメント**を**クラスター**の **REST API** エンドポイントに送信

2

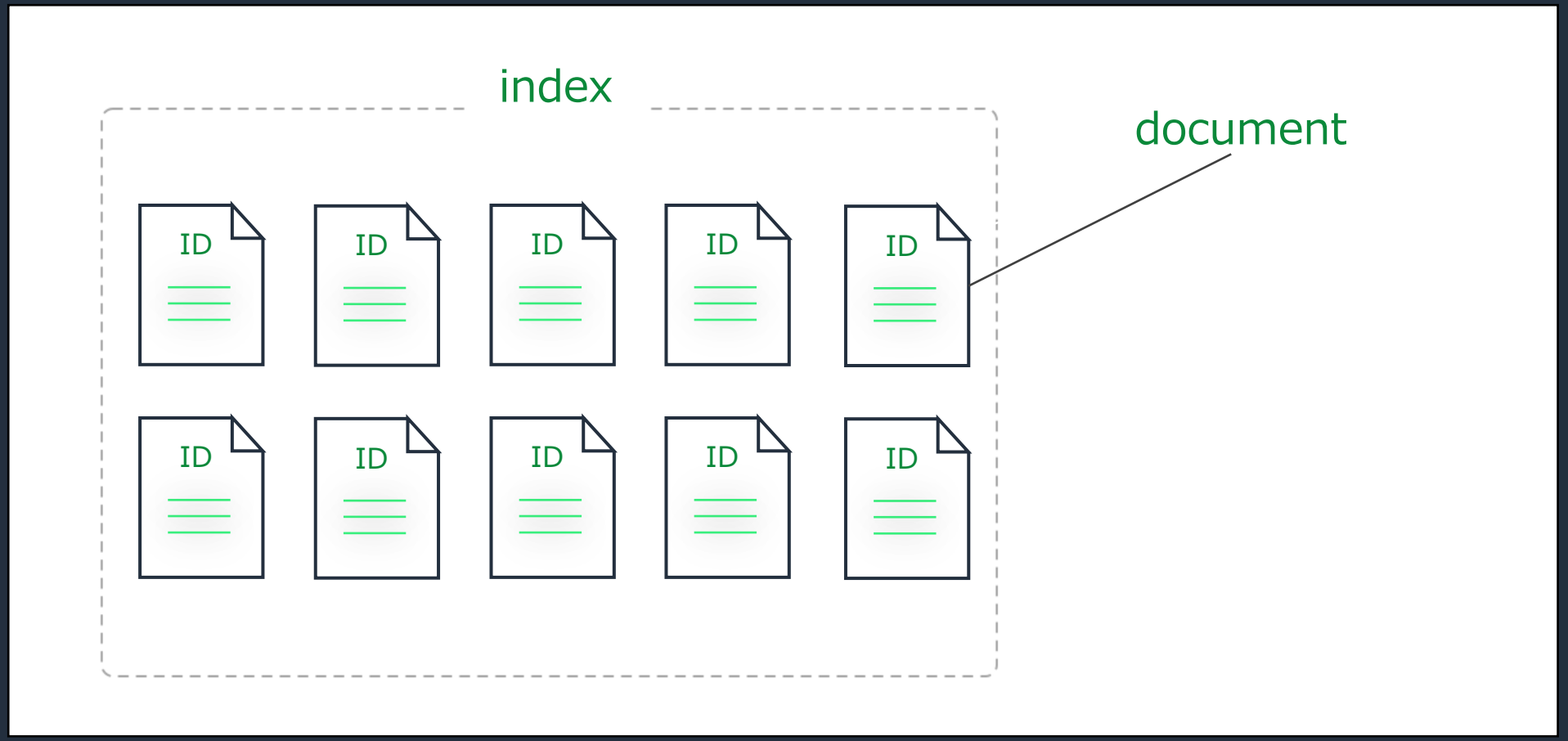
クラスターはドキュメントを**インデックス**に格納する。

3

- クラスターの REST API エンドポイントにクエリを発行し、JSON 形式のドキュメントを取得
- **ダッシュボード**に Web ブラウザからアクセス、可視化や分析作業を実施

インデックスとドキュメント

ドキュメントはインデックスに格納されることで検索可能となる



ドキュメント

ドキュメント = JSON 形式で表現される単一のデータ

一般的なデータベースのレコードに相当

各フィールドは文字列型や数値型、Boolean 型など、特定の型を持つ
(ベクトル検索では、このフィールドの 1 つにベクトル型を持つ)

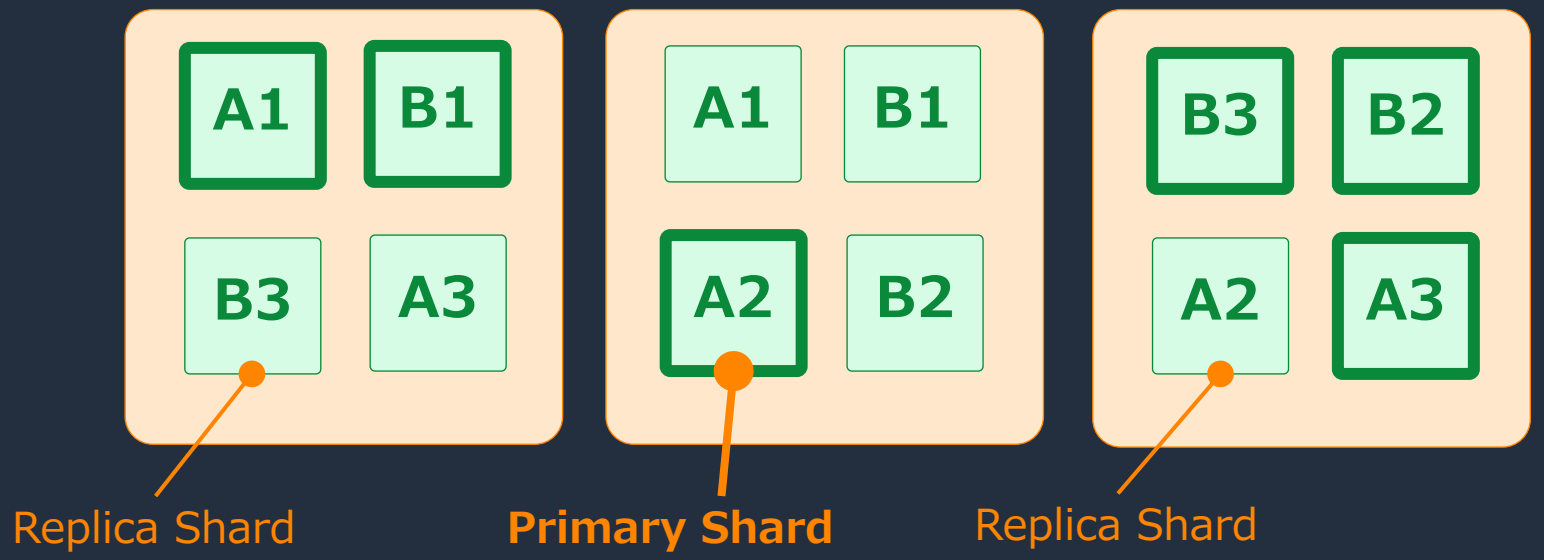
Field

```
{
"name": "John Smith",
"age": 42,
"confirmed": true,
"join_date": "2014-06-01",
"home": { "lat": 51.5, "lon": 0.1 },
"accounts": [
   { "type": "facebook", "id": "johnsmith" },
   { "type": "twitter", "id": "johnsmith" }
 ]
}
```

OpenSearch アーキテクチャの概要

インデックスを複数のシャードに分割し、データノードに分散配置することで、
高いスケーラビリティを獲得

レプリカを設定することで耐障害性とスループットの向上を実現




```
PUT A
{
  “settings” : {
    “index” : {
      “number_of_shards” : 3,
      "number_of_replicas" : 1
    }
  }
}
```

Amazon OpenSearch Service

Amazon OpenSearch Service

OpenSearch を簡単にデプロイ・管理、
スケール可能なフルマネージドサービス

フルマネージド: リソースのデプロイ、
管理に費やす時間を削減

セキュリティ: 認証、認可、暗号化、監査、
およびコンプライアンスのための高度な
セキュリティを維持

データ分析・オブザーバビリティ:
潜在的な脅威を体系的に検出し、機械学習、
アラート、可視化を活用して対処

コスト最適化: 各種リソースを最適化し、
戦略的な作業に注力

OpenSearch Service を使用したベクトル検索 (k-NN 検索)

OpenSearch では Exact kNN および 近似 k-NN を利用可能
以下のライブラリ・アルゴリズムの組み合わせに対応

- nmslib (HNSW)
- Faiss (HNSW, IVF, PQ) * OpenSearch 1.2 以降
- Lucene (HNSW) * OpenSearch 2.3 以降

参考) OpenSearch Service 近似 k-NN アルゴリズムの違い

	NMSLIB-HNSW	FAISS-HNSW	FAISS-IVF	Lucene-HNSW
最大次元数	16,000	16,000	16,000	1024
フィルター	Post filter	Post filter	Post filter	Filter while search
トレーニングの必要性	No	No	Yes	No
距離関数	l2, innerproduct, cosinesimil, l1, linf	l2, innerproduct	l2, innerproduct	l2, cosinesimil
ベクトル数	Tens of billions	Tens of billions	Tens of billions	< Ten million
インデックスのレイテンシー	Low	Low	Lowest	Low
クエリのレイテンシーと精度	Low latency & high quality	Low latency & high quality	Low latency & low quality	High latency & high quality
ベクトルの圧縮	Flat	Flat Product Quantization	Flat Product Quantization	Flat
メモリ消費	High	High Low with PQ	Medium Low with PQ	High

https://aws.amazon.com/jp/blogs/big-data/amazon-opensearch-services-vector-database-capabilities-explained/

OpenSearch Service を使用したベクトル検索の流れ

1. 格納するデータ量 (ベクトルの次元数・データ数・アルゴリズム等によって決まる) から OpenSearch クラスターのサイジングを行い、クラスターを作成する。
2. インデックスを作成する (ここで使用するフィールドの定義やアルゴリズムを決定する)
3. データを OpenSearch クラスターに投入する
   (事前に ML モデルなどを使用して検索したい情報をベクトル化する必要あり)
4. k-NN 用のクエリを用いて、検索する

インデックス作成クエリの例 (PUT /sample-index)

```
{
  "settings": {
    "index.knn": true
  },
  "mappings": {
    "properties": {
      "my_vector_field": {
        "type": "knn_vector",
        "dimension": 2,
        "method": {
          "name": "hnsw",
          "engine": "faiss"
        }
      }
    }
  }
}
```

設定内容

- my-vector-field という名前の knn_vector 型のフィールドを持つ
- Faiss エンジンの HNSW アルゴリズムを使用
- ベクトルの次元数は 2

OpenSearch Service を使用したベクトル検索の流れ

1. 格納するデータ量 (ベクトルの次元数・データ数・アルゴリズム等によって決まる) から OpenSearch クラスターのサイジングを行い、クラスターを作成する。
2. インデックスを作成する (ここで使用するフィールドの定義やアルゴリズムを決定する)
3. データを OpenSearch クラスターに投入する
   (事前に ML モデルなどを使用して検索したい情報をベクトル化する必要あり)
4. k-NN 用のクエリを用いて、検索する

検索クエリの例 (GET /sample-index/_search)

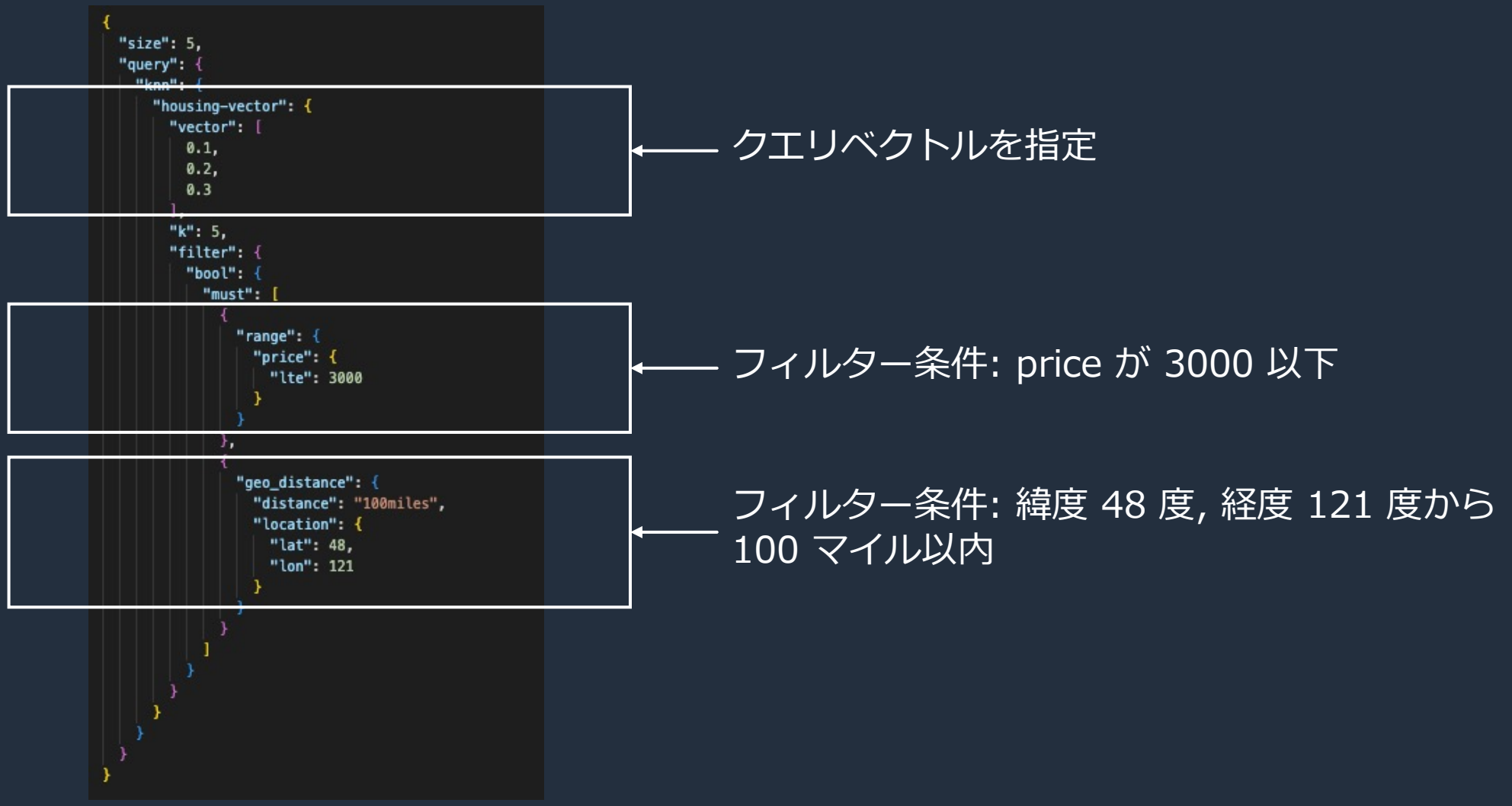
```
{
  "size": 2,
  "query": {
    "knn": {
      "my_vector_field": {
        "vector": [2, 3],
        "k": 2
      }
    }
  }
}
```

設定内容

- [2,3] というクエリに近いベクトルを合計 2 件返すように指定している
  (size は合計の返却数,
  k は各シャードの返却数)

フィルターを用いたベクトル検索

ベクトル検索の際に、フィルターを使用して検索対象を絞ることが可能



Amazon OpenSearch Serverless

Amazon OpenSearch Serverless

クラスターの管理なしに検索と分析ワークロードを実行

管理が容易
クラスタのサイジング、スケーリング、チューニング、シャードとインデックスのライフサイクル管理が不要に

速度
リソースを自動的にスケールし、高速なデータ取り込みレートとクエリ応答時間を一貫して維持

エコシステム
既存の OpenSearch クライアント、パイプライン、API を使用して数秒で利用を開始できる

費用対効果
事前のリソースプロビジョニングは不要

OpenSearch Serverless を使用したベクトル検索

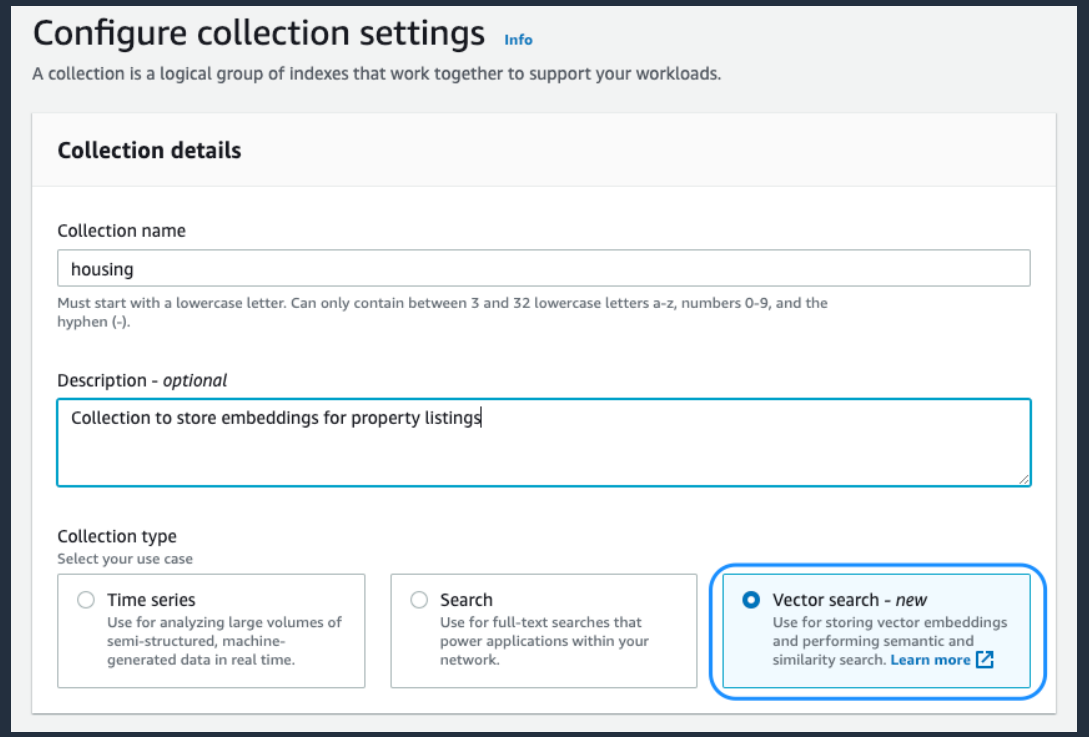
PREVIEW

OpenSearch の k-NN 機能が OpenSearch Serverless でも利用可能に

インフラ管理の手間なくベクトル検索機能を実現することができる

アジアパシフィック(東京)、米国東部(オハイオ、バージニア北部）、米国西部（オレゴン）、アジアパシフィック（シンガポール、シドニー）、ヨーロッパ（フランクフルト、アイルランド）のリージョンでプレビューを開始

現在 HNSW アルゴリズムのみ対応している



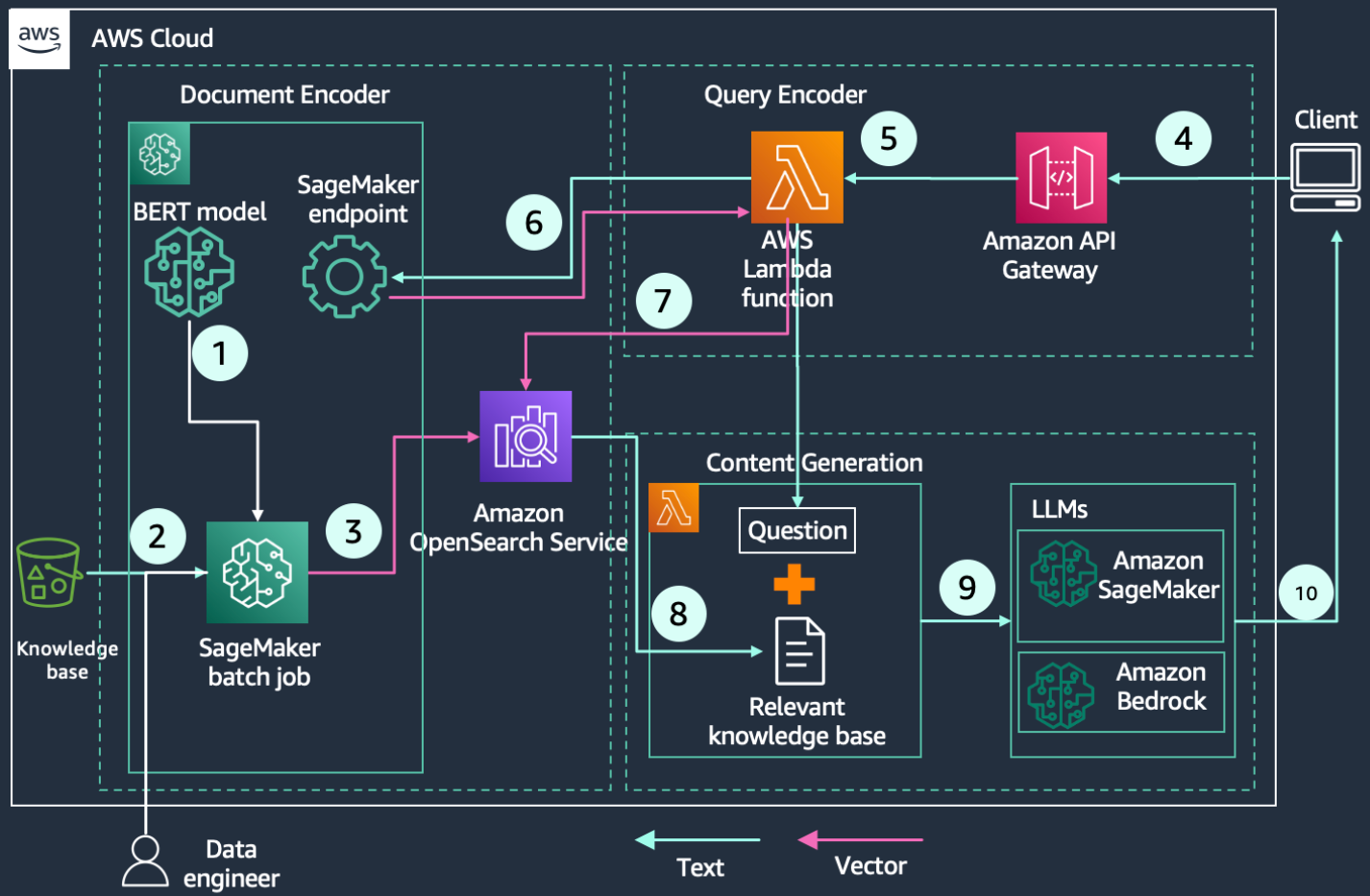
OpenSearch Serverless を使用したベクトル検索の流れ

PREVIEW

1. OpenSearch Serverless の Vector Search コレクションを作成する
   (クラスターのサイジングが不要)
2. インデックスを作成する (ここで使用するフィールドの定義やアルゴリズムを決定する)
3. データを OpenSearch コレクションに投入する
   (事前に ML モデルなどを使用して検索したい情報をベクトル化する必要あり)
4. k-NN 用のクエリを用いて、検索する

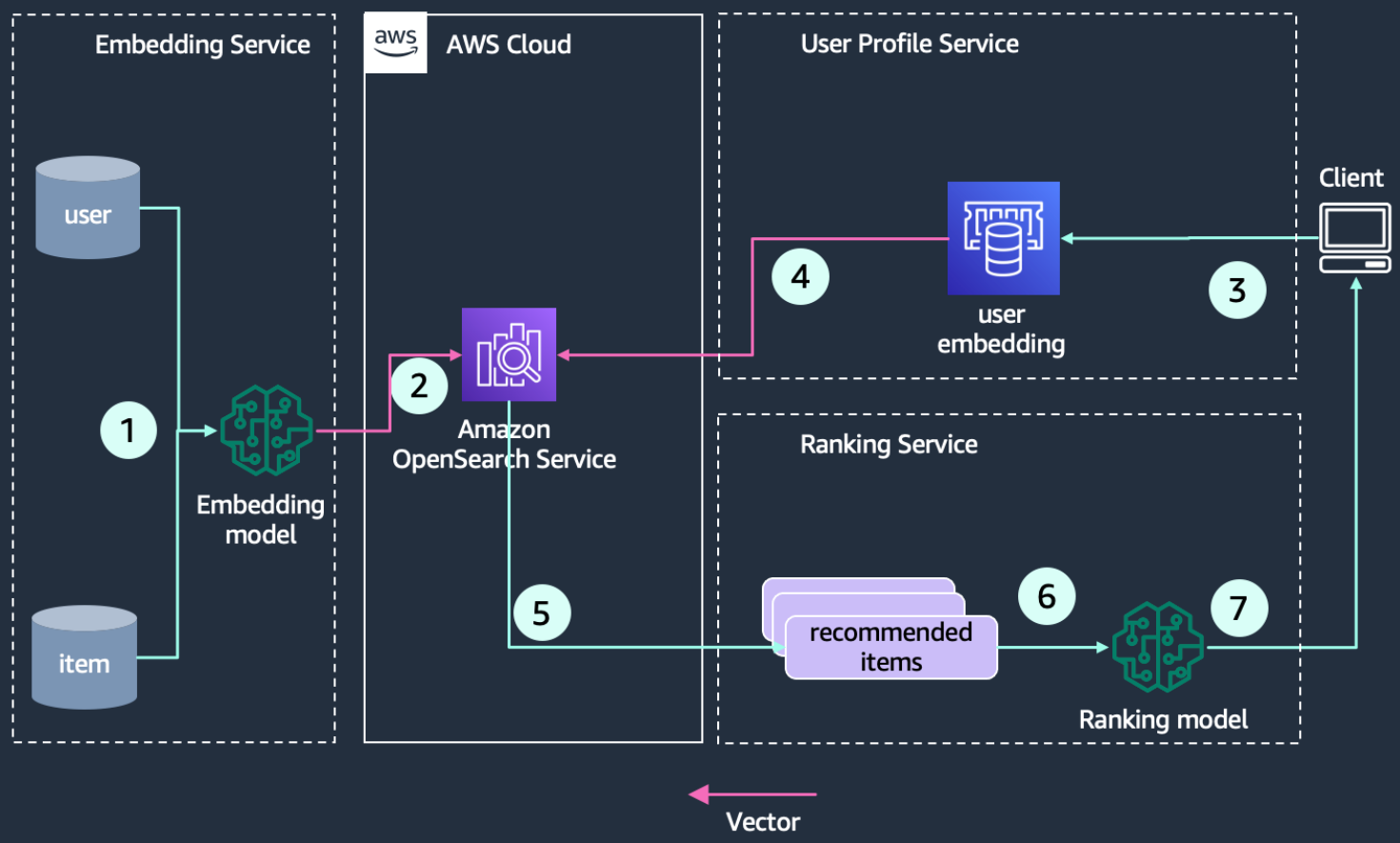
OpenSearch Service のベクトル検索と同様

ベクトル検索 アーキテクチャ (RAG)



1. Load BERT model into OpenSearch
2. Data engineer load knowledge base
3. Use SageMaker language model to generate embeddings for knowledge base
4. Client submit one question
5. API Call Lambda backend service in Lambda
6. Backend service call SageMaker Endpoint to convert search query into vector
7. Use OpenSearch vector search to get relevant documents
8. Return relevant knowledge base to backend service
9. Backend use relevant knowledge base as context, combine user's original question as prompt to LLMs
10. Foundation Models generate factual answer based on relevant knowledge for the original question

ベクトル検索 アーキテクチャ (レコメンド)



1. Data engineer use recommendation algorithms to generate embedding with user and product data
2. Load embedding (vector) into OpenSearch
3. Get user's embedding from cache when user login
4. User OpenSearch vector search to get similar vectors
5. Return recommended items which are vector similar with user's embedding/vector
6. Rank the items with ranking model
7. Return ranked products to user

OpenSearch Serverless ベクトル検索 デモ

まとめ

- ベクトル検索は、レコメンド・画像検索・RAG など、機械学習を用いたアプリケーションで使用される技術
- OpenSearch はベクトル検索機能 (k-NN 機能) を持っており、Amazon OpenSearch Service、Amazon OpenSearch Serverless からも利用可能

Thank you!