Weaviateの日本語対応とマルチモーダル・多言語検索デモ

2024/12/11

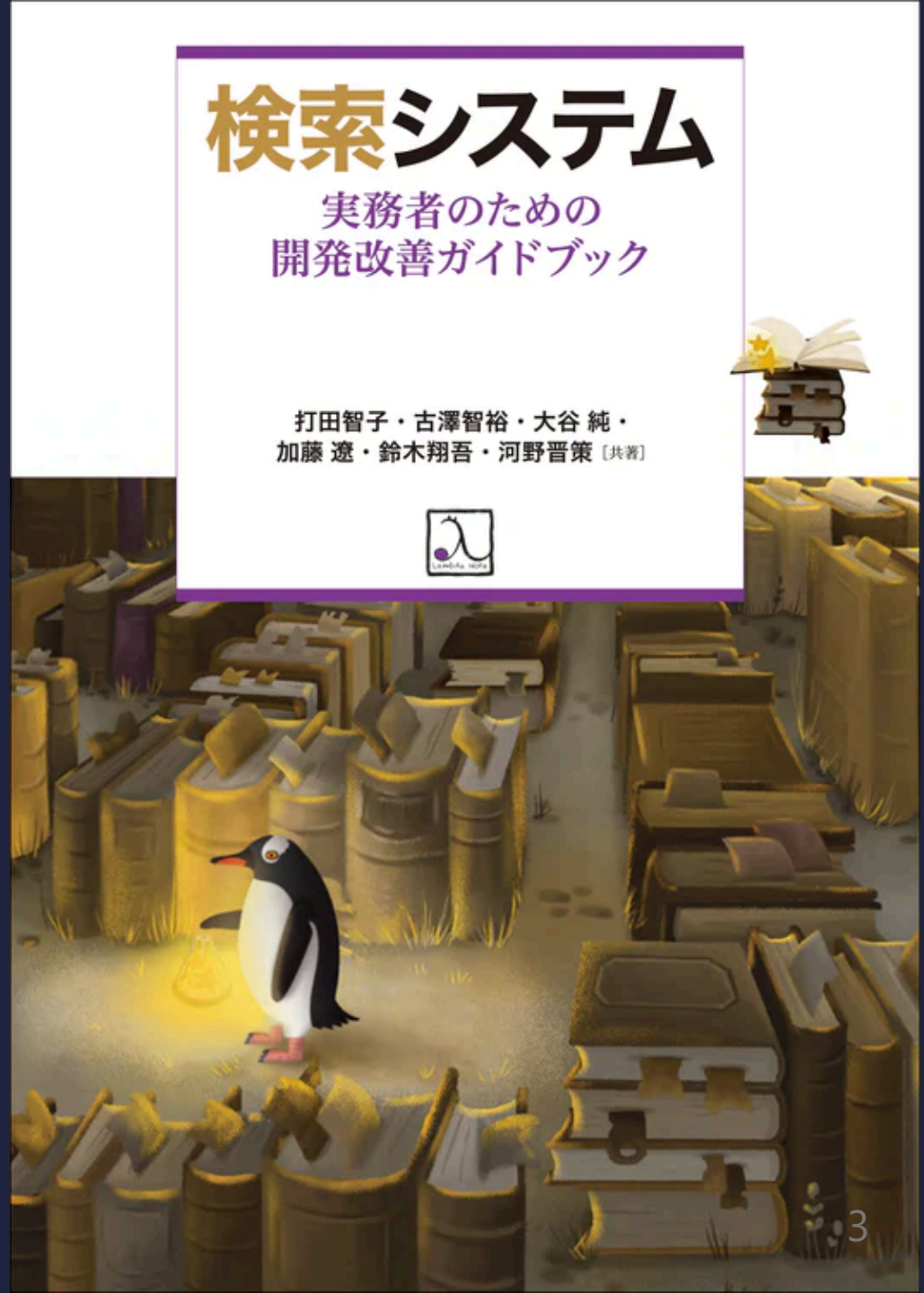
Jun Ohtani / @johtani

自己紹介

- フリーランスエンジニア
  / コンサルタント
- 検索技術勉強会主催者の一人
- 検索システムの著者の一人
  （ラムダノートより出版）

宣伝 絶賛発売中

- 検索についてはこちらの本が
  いいらしいです
- （ベクトル検索については説明はない
  ですが。。。）
- 以下、書籍のページへ



検索とは

> 検索とは、データの集合の中から
> 目的のデータを探し出すことである。
> また、目的のデータが存在しないことの
> 確認行為も含まれる。

-- Wikipediaより https://ja.wikipedia.org/wiki/検索

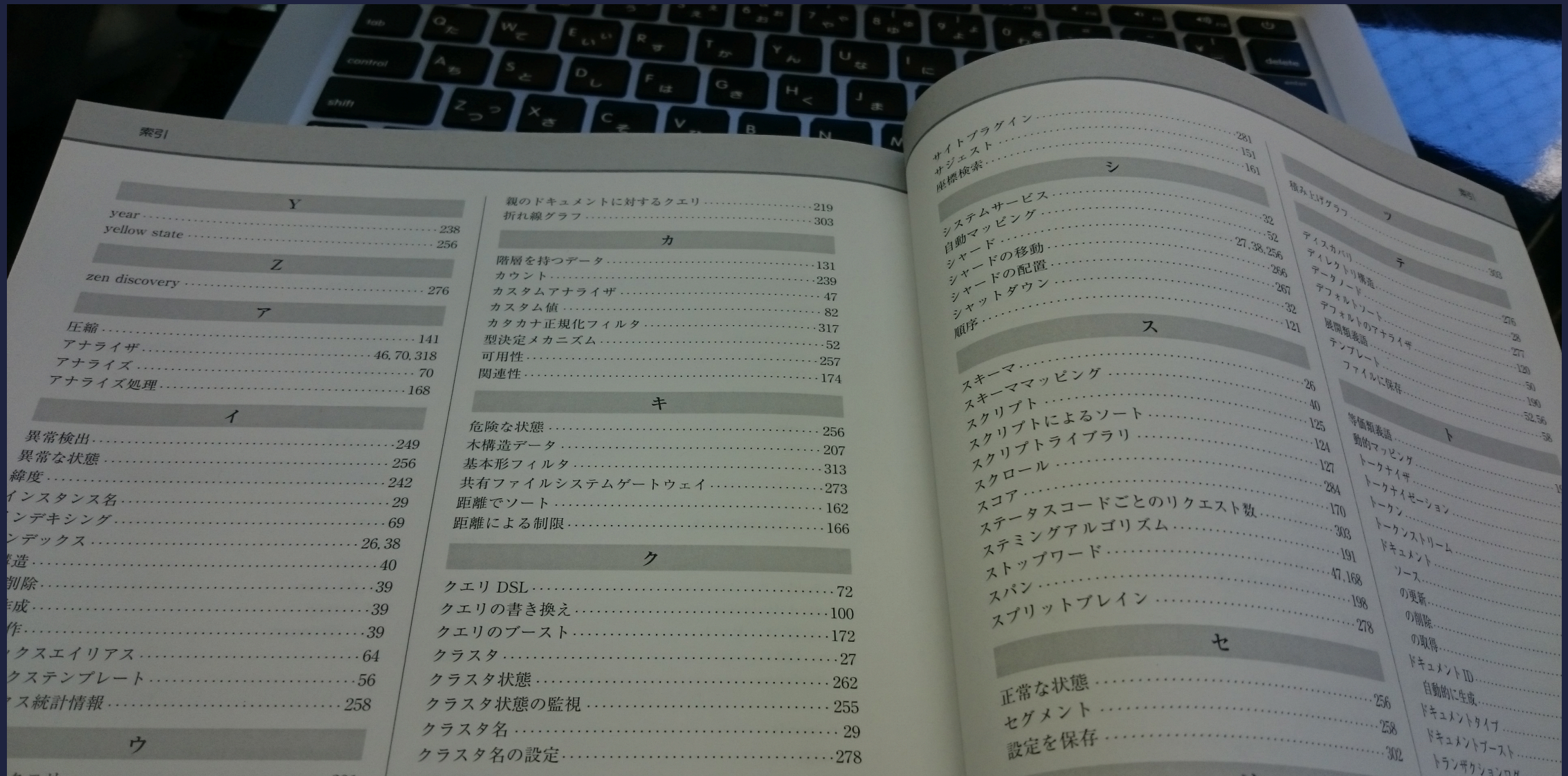
検索の手法

- おもな検索システムの仕組み
  - Grep（ファイル内検索）
    - 文字の一致検索（先頭から文字列を見ていく）
  - キーワード検索
    - 転置インデックス
  - ベクトル検索
    - ベクトル表現による近傍検索

検索ユースケース - *Grep*

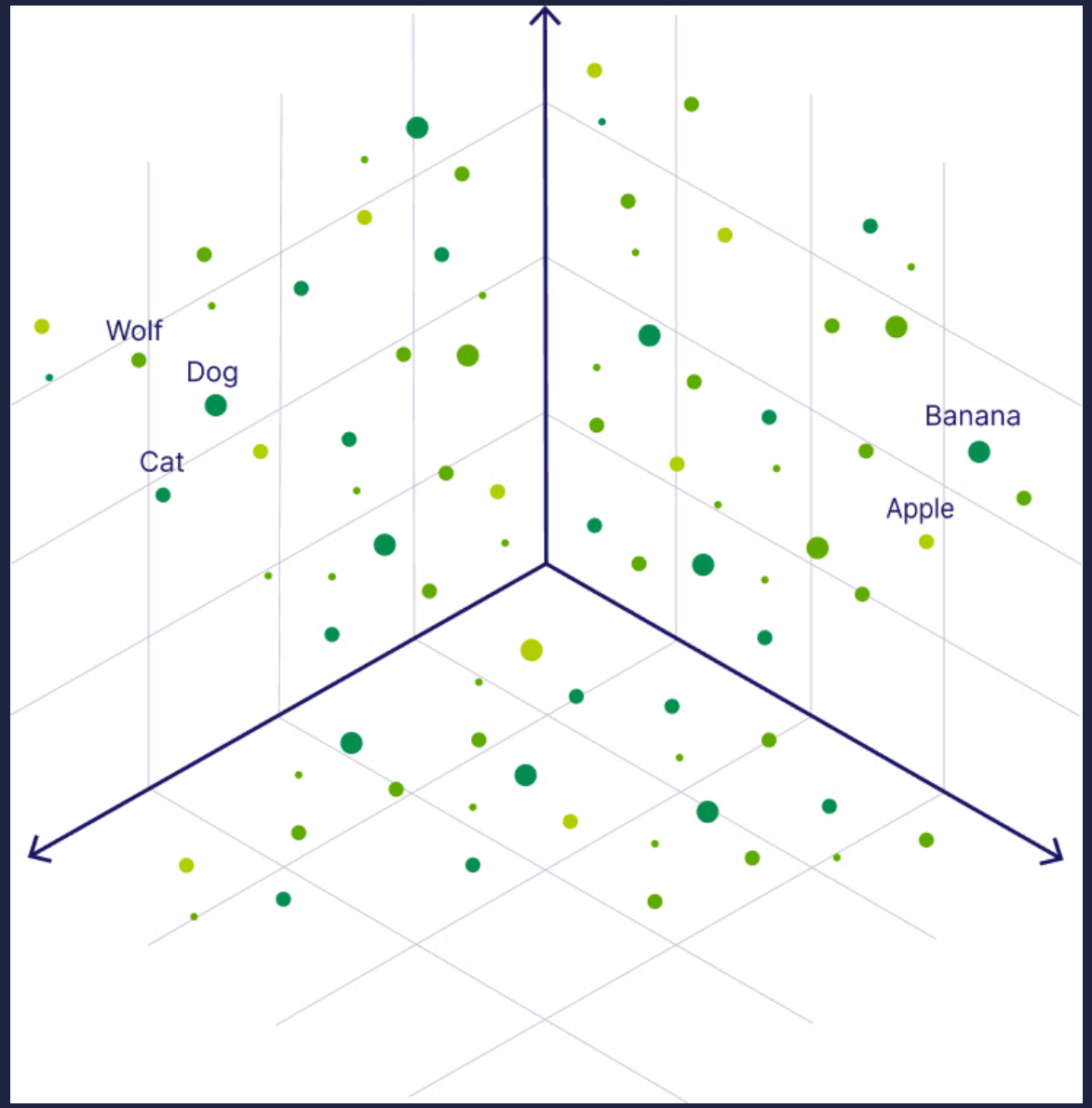
```
johtani@johtanis-macbook-pro-16 ~/projects/blog/hugo/blog_generator/content/post master ± grep java */*
2013/2013-06-19-introduction-kibana3.md:$ git clone https://github.com/elasticsearch/kibana.git kibana-javascript
2013/2013-06-19-introduction-kibana3.md:$ cp -R kibana-javascript /var/www/html
2013/2013-06-19-introduction-kibana3.md:ElasticSeachサーバとKibana3のApacheのサーバが別のサーバの場合やElasticSearchサーバのポートが異なる場合はkibana-javascript/config.jsファイルの編集が必要になります。
2013/2013-06-19-introduction-kibana3.md:```javascript
2013/2013-06-19-introduction-kibana3.md:http://hogehoge/kibana-javascript/
2013/2013-07-04-schemaless-example.md:java -Dsolr.solr.home=example-schemaless/solr -jar start.jar
2013/2013-07-04-schemaless-example.md:        <str name="valueClass">java.lang.Boolean</str>
2013/2013-07-04-schemaless-example.md:        <str name="valueClass">java.util.Date</str>
2013/2013-07-04-schemaless-example.md:        <str name="valueClass">java.lang.Long</str>
2013/2013-07-04-schemaless-example.md:        <str name="valueClass">java.lang.Integer</str>
2013/2013-07-04-schemaless-example.md:        <str name="valueClass">java.lang.Number</str>
2013/2013-08-02-morphlines-loadsolr.md:なんか調べることになってたので、関係しそうなMorphlinesの[LoadSolr](https://github.com/cloudera/cdk/blob/master/cdk-morphlines/cdk-morphlines-solr-core/src/main/java/com/cloudera/cdk/morphline/solr/LoadSolrBuilder.java)コマンドを調べてみました。
2013/2013-08-02-morphlines-loadsolr.md:別途、[sanitizeUnknownSolrFilds](https://github.com/cloudera/cdk/blob/master/cdk-morphlines/cdk-morphlines-solr-core/src/main/java/com/cloudera/cdk/morphline/solr/SanitizeUnknownSolrFieldsBuilder.java)というコマンドが用意されていて、Solrのスキーマにないものはこのコマンドを使って、無視するフィールド名に変えたり、雑多なデータを入れるためのフィールド名にするといった処理ができるようです。このコマンド内部で、Solrのスキーマ設定を元に、Solrのフィールドに合致する物があるかをチェックして処理しています。
2013/2013-08-02-morphlines-loadsolr.md:Solrへの登録処理自体はLoadSolrクラス内部でDocumentLoaderというクラスのload()メソッドを呼び出しているだけでした。ということで、[DocumentBuilder](https://github.com/cloudera/cdk/blob/master/cdk-morphlines/cdk-morphlines-solr-core/src/main/java/com/cloudera/cdk/morphline/solr/DocumentLoader.java)クラスを少し調査。
2013/2013-08-02-morphlines-loadsolr.md:2. なければ、[SolrServerDocumentLoader](https://github.com/cloudera/cdk/blob/b6f98cff4a027af04f97fdec9abf729785d74cf5/cdk-morphlines/cdk-morphlines-solr-core/src/main/java/com/cloudera/cdk/morphline/solr/SolrServerDocumentLoader.java)をnewしたものを利用
2013/2013-08-02-morphlines-loadsolr.md:Solrへの接続情報とか設定ファイルとかSolrCloud用のZooKeeperとかは[SolrLocatorクラス](https://github.com/cloudera/cdk/blob/master/cdk-morphlines/cdk-morphlines-solr-core/src/main/java/com/cloudera/cdk/morphline/solr/SolrLocator.java)に設定される内容が利用されます。
2013/2013-08-02-morphlines-loadsolr.md:1.のパターンは、どうやら、[Cloudera SearchのMapReduceIndexerToolのクラス](https://github.com/cloudera/search/blob/master/search-mr/src/main/java/org/apache/solr/hadoop/morphline/MorphlineMapper.java)にあるMyDocumentLoaderかなぁと。
2013/2013-08-02-morphlines-loadsolr.md:内部処理は、HadoopのContext.writeメソッドにでSolrInputDocument（＝MorphlinesのRecord）を書きだして、ReducerでSolrOutputFormatでインデックス作成の流れかなと。たぶん、[MorphlineMapRunner](https://github.com/cloudera/search/blob/master/search-mr/src/main/java/org/apache/solr/hadoop/morphline/MorphlineMapRunner.java)あたり
```

検索ユースケース - 本


索引
Y
year 238
yellow state 256
Z
zen discovery 276
ア
圧縮 141
アナライザ 46, 70, 318
アナライズ 70
アナライズ処理 168
イ
異常検出 249
異常な状態 256
緯度 242
インスタンス名 29
インデキシング 69
インデックス 26, 38
構造 40
削除 39
作成 39
操作 39
エイリアス 64
テンプレート 56
統計情報 258
ウ
親のドキュメントに対するクエリ 219
折れ線グラフ 303
カ
階層を持つデータ 131
カウント 239
カスタムアナライザ 47
カスタム値 82
カタカナ正規化フィルタ 317
型決定メカニズム 52
可用性 257
関連性 174
キ
危険な状態 256
木構造データ 207
基本形フィルタ 313
共有ファイルシステムゲートウェイ 273
距離でソート 162
距離による制限 166
ク
クエリ DSL 72
クエリの書き換え 100
クエリのブースト 172
クラスタ 27
クラスタ状態 262
クラスタ状態の監視 255
クラスタ名 29
クラスタ名の設定 278
サイトプラグイン 281
サジェスト 151
座標検索 161
シ
システムサービス 32
自動マッピング 52
シャード 27, 38, 256
シャードの移動 266
シャードの配置 267
シャットダウン 32
順序 121
ス
スキーマ 26
スキーママッピング 40
スクリプト 125
スクリプトによるソート 124
スクリプトライブラリ 127
スクロール 284
スコア 170
ステータスコードごとのリクエスト数 303
ステミングアルゴリズム 191
ストップワード 47, 168
スパン 198
スプリットブレイン 278
セ
正常な状態 256
セグメント 258
設定を保存 302
積み上げグラフ
フ
ディスカバリ
ディレクトリ構造
データノード
デフォルトソート
デフォルトのアナライザ
展開類義語
テンプレート
ファイルに保存
テ
等価類義語
動的マッピング
トークナイザ
トークナイゼーション
トークン
トークンストリーム
ドキュメント
ソース
の更新
の削除
の取得
ドキュメントID
自動的に生成
ドキュメントタイプ
ドキュメントブースト
トランザクションログ
ト

ベクトル検索

- キーワード検索よりも柔軟な検索が可能
  - 「意味」が近いものを検索
  - 同義語、類義語など
- テキスト以外のデータも検索可能



デモ

- CLIPを用いて文章から画像を検索する
- デモに使用したデータ
  - MS COCOの画像のうち商用利用可能なライセンスの画像
    - https://cocodataset.org/#home
  - 日本語のキャプションデータ
    - http://captions.stair.center/

デモ画面

Multi Modal/Language CLIP Hybrid Search

12584 images you can search

bm25-gse　bm25-kagome　vector　hybrid　vector+filter by kagome

Put your words what what you want?

青いバスが走っている

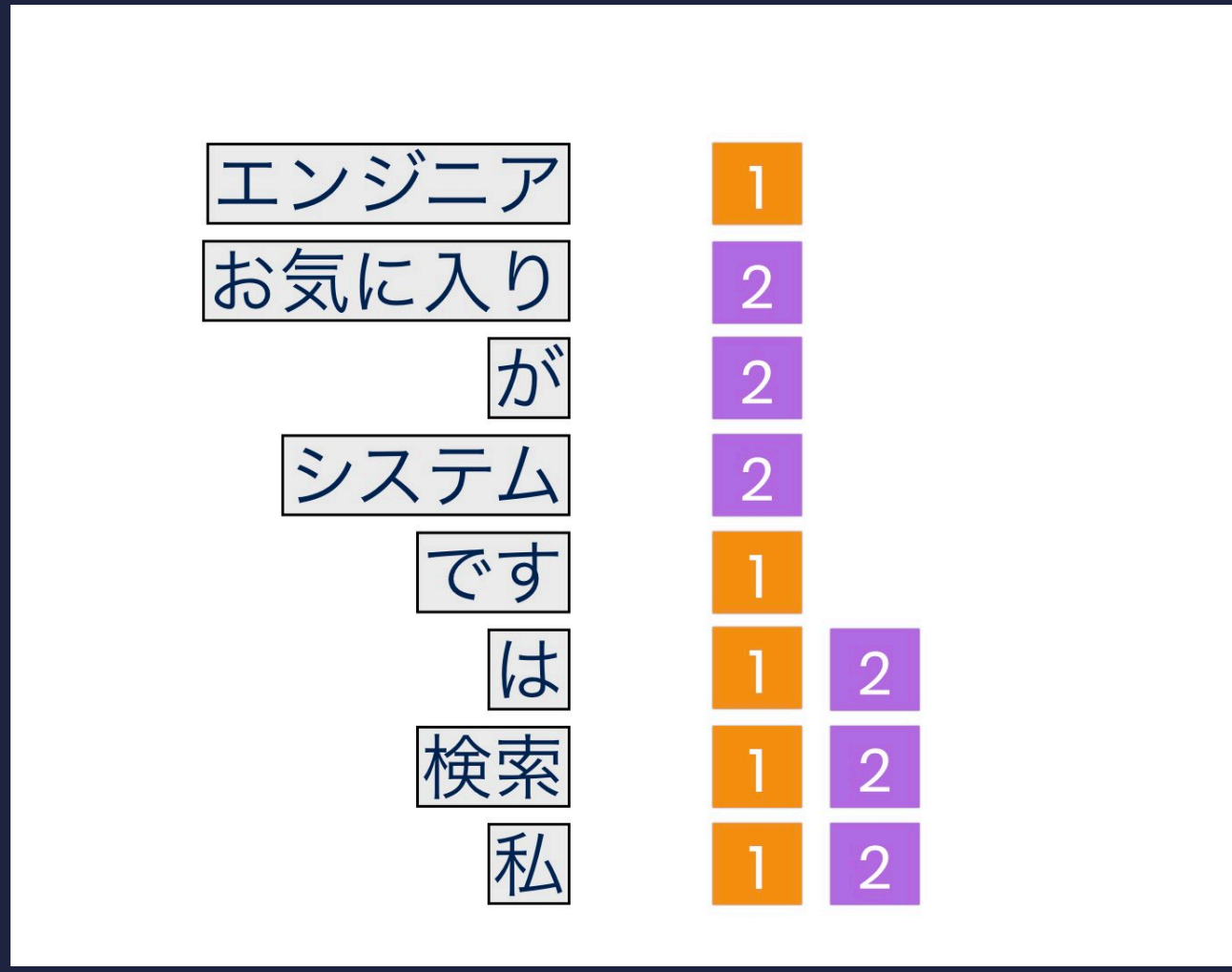
Filter word?

何が苦手？何が得意？

- ベクトル的に近いとは？
  - 意味合いの「似たもの」が検索できる
  - 近いか近くないかの判断基準が難しい
- この「キーワード」を含む（含まない）検索
  - 特定の単語を必ず含む検索が難しい

キーワード検索

- 転置インデックス
- キーワードからIDのリストを取得
- リストをもとにORやAND、NOT検索も可能
- キーワードが違えば違うデータ



キーワード？

- どうやって文章からキーワードを作る？
  - 「Weaviate is awesome」
  - 「Weaviateはすばらしい」
  - 「Weaviate ist erstaunlich」
  - 「Weaviate는 훌륭합니다.」

英語と日本語の違い

- 英語でキーワードは空白区切り
  - 「Weaviate is awesome」
  - 「Weaviate」「is」「awesome」
- じゃあ、日本語でキーワードは？
  - 「Weaviateはすばらしい」
  - 「Weaviate」「は」「すばらしい」

どうして日本語対応？

- 日本語の場合はそもそもキーワードを選び出すのが大変
  - 選び出す方法は１つだけではない
- ベクトルだとLLMなどがいい感じにベクトルにしてくれる
  - モデルが日本語に対応していればいい
  - （実際にはどんな日本語に対応しているか？など考慮すべき点はある）

Weaviateでの日本語キーワード検索

- Tri-gram
  - 文字N-gram（N=3）
- GSE
  - 形態素解析ライブラリ
  - https://github.com/go-ego/gse
- Kagome JP（*NEW!*）
  - 形態素解析ライブラリ
  - https://github.com/ikawaha/kagome

Tri-gram

- 入力：「Weaviateはすばらしい」
- 出力：「wea」「eav」「avi」「via」「iat」「ate」「te(は」「e(はす」「(はすば」「すばら」「ばらし」「らしい」

Tri-gram

- メリット
  - 辞書不要
  - 部分一致する
  - Weaviate Cloudでも使える
- デメリット
  - 不自然な単語がある
  - 出てくる単語が多数

GSE

- 入力：「Weaviateはすばらしい」
- 出力：「w」「e」「a」「v」「i」「a」「t」「e」「はす」「すばらし」「すばらしい」「ばら」「ばらし」「らし」「らしい」「しい」

GSE

- メリット
  - 辞書にある単語をできるだけ
  - 部分一致する
- デメリット
  - 英単語は1文字ずつに分割
  - 不自然な単語がある
  - 辞書が必要
  - Weaviate Cloudでは使えない（まだ）

Kagome（今回追加したもの）

- 入力：「Weaviateはすばらしい」
- 出力：「weaviate」「は」「すばらしい」

Kagome（今回追加したもの）

- メリット
  - 辞書にある単語
  - 不要な単語が出ない
- デメリット
  - 単語中の部分一致はしない
  - 辞書が必要
  - Weaviate Cloudでは使えない（まだ）

実際にデモで検索してみよう

- CLIPを用いて文章から画像を検索する
- デモに使用したデータ
  - MS COCOの画像のうち商用利用可能なライセンスの画像
    - https://cocodataset.org/#home
  - 日本語のキャプションデータ
    - http://captions.stair.center/

デモ

- グリンピースのサラダ
- バスケットボールの試合

設定(環境変数)

- 起動時に環境変数で設定（GSEとKagome）
  - `ENABLE_TOKENIZER_GSE=true`
  - `ENABLE_TOKENIZER_KAGOME_JA=true`
  - `ENABLE_TOKENIZER_KAGOME_KR=true`
    - 韓国語用もあります
- Weaviate Cloudだと？

設定(スキーマ in Go)

```
Properties: []*models.Property{
  ...
  {
    DataType:     []string{schema.DataTypeTextArray.String()},
    Description:  "Caption in Japanese",
    Name:         "caption_ja_gse",
    Tokenization: models.PropertyTokenizationGse,
  },
  {
    DataType:     []string{schema.DataTypeTextArray.String()},
    Description:  "Caption in Japanese",
    Name:         "caption_ja",
    Tokenization: models.PropertyTokenizationKagomeJa,
  },
  ...
}
```

設定(スキーマ in Python)

```
...
properties=[
    wc.Property(name="caption_ja_gse",
                data_type=wc.DataType.TEXT,
                tokenization=wc.Tokenization.GSE
                ),
    wc.Property(name="caption_ja",
                data_type=wc.DataType.TEXT,
                tokenization=wc.Tokenization.KAGOME_JA
                ),
]
...
```

もっと日本語検索をいい感じにするためには？

- 文字正規化
  - 半角全角
  - 合字
- カスタム辞書
  - 類義語
  - ユーザー辞書（形態素解析）
- フレーズ検索

文字正規化

- 半角・全角
  - カタカナ、ｶﾀｶﾅ
  - ｋａｔａｋａｎａ、katakana
  - １２３、123
- 合成済み文字
  - 「ガ」と「ガ」
  - 「`&#x30ab;&#x3099;`」と「`&#x30ac;`」

カスタム辞書

- 類義語、アクロニム
  - 「羽田空港」＝「東京国際空港」
  - 「NASA」＝「National Aeronautics and Space Administration」
- ユーザー辞書
  - 新しい単語はシステム辞書にはない
  - 「圏央道」「副都心線」

フレーズ検索

- 日本語はTokenizationで単語分割される
- GSEやTrigramでは「weaviate」が分割される
  - 「wea」「eav」「avi」「via」「iat」「ate」
  - 「w」「e」「a」「v」「i」「a」「t」「e」
- ORで検索するとノイズが多い
- FR: Phrase Matching on text property queries #2688
  - https://github.com/weaviate/weaviate/issues/2688

まとめ

- Weaviateは日本語も検索できる
  - ベクトルでも
  - キーワードでも
  - ハイブリッドも可能
- キーワード検索での選択肢は３つ
  - Tri-gram、GSE、Kagome
- 今日から試して見れる（かも？）