NFE204

Recherche d'information

Auteurs : Raphaël Fournier-S'niehotta, Philippe Rigaux
(fournier@cnam.fr, philippe.rigaux@cnam.fr)

EPN Informatique
Conservatoire National des Arts & Métiers, Paris, France

Plan du cours

Lors du cours précédent

Introduction générale à la RI

- Contexte d'utilisation
- Évaluation des moteurs
- Index inversés : structure de données performante
- Présentation d'ElasticSearch

Aujourd'hui

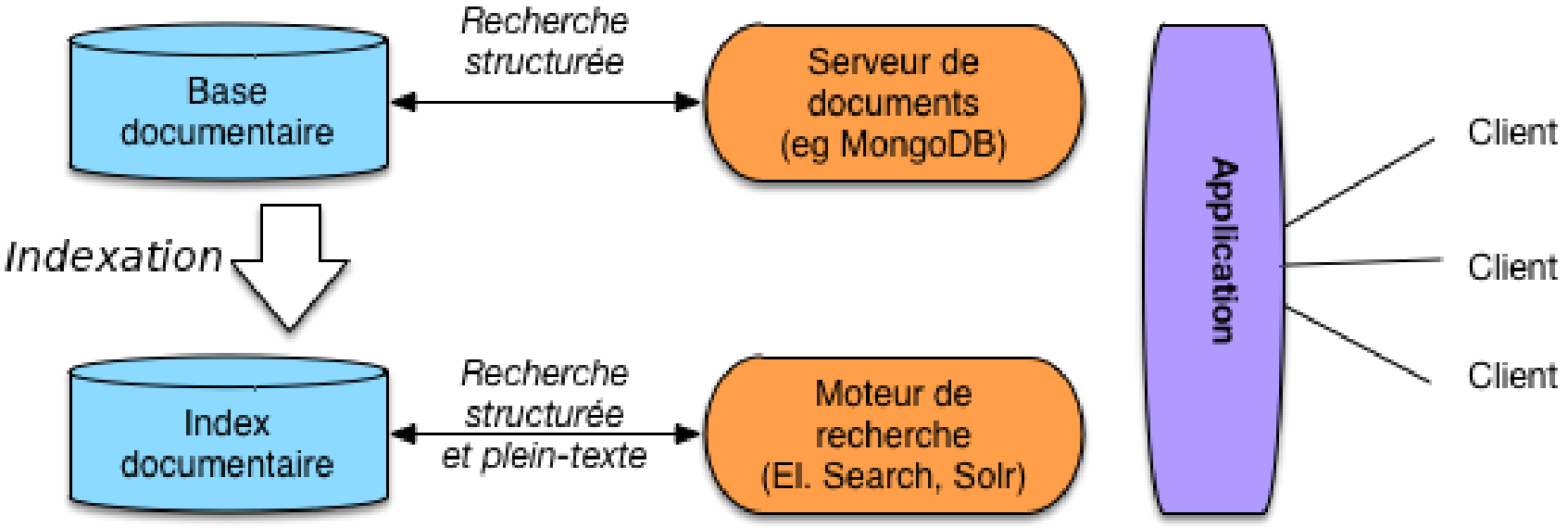
- Comment transformer les données pour des recherches efficaces ?
- Utilisation d'un schéma
- Détail de l'analyse du texte

- Reprise de la mise en pratique d'ElasticSearch
- Interactions avec l'API REST
- Premiers documents, premières requêtes

Plan du cours

2 Indexation
- Tokenisation et normalisation
- Racinisation
- Mots vides, synonymes, cas particuliers...

Bases documentaires et moteur de recherche



Pré-traitement

- Dès qu'un document est un peu complexe, on ne peut pas le découper arbitrairement sans appliquer un pré-traitement réfléchi
- sinon :
  - que devient "pomme de terre" ?
  - on stocke des listes très longues pour certains mots...Retour à ES
  - l'utilisateur doit faire très attention ("loup", "Loup", "loups" sont différents)
- il faut procéder à une analyse du contenu

Rôle de l'analyse

L'analyse des textes permet d'effectuer une forme de normalisation / unification pour être moins dépendant de la forme du texte.

- un document parle de loup même si on y trouve les formes "loups", "Loup", "louve", etc.
- un document parle de travail quelle que soit la forme du verbe "travailler" ou de ses variantes.
- Jusqu'où va-t-on? Traductions (loup = wolf = lupus) ? Synonymes (loup = prédateur) ? (sujet de recherches)

Pour moins dépendre de la forme on applique des transformations.

Impact de l'analyse

Bien comprendre :

Plus on normalise, plus on diminue la précision.
Car des mots distincts sont unifiés (cote, côte, côté, etc.)

Plus on normalise, plus on améliore le rappel.

Car on met en correspondance les variantes d'un même mot, d'une même signification (conjugaisons d'un verbe).

Impact de l'analyse

Très important

La même transformation doit être appliquée aux documents et à la requête.

Pourquoi ?

- sinon, il se crée un décalage entre ce qui a été analysé et ce qui peut être réellement recherché

Les phases de l'analyse

Important: identification de quelques méta-données (la langue), prise en compte du contexte (quels documents pour quelle application).

Puis, de manière générale :

- Tokenization : découpage du texte en "mots"
- Normalisation : majuscules ? acronymes ? apostrophes ? accents ?
  Exemple : *Windows* et *window*, U.S.A vs USA, *l'étudiant* vs *les étudiants*.
- Stemming ("racinisation"), lemmatization
  Prendre la racine des mots pour éviter le biais des variations (étudier, étudiant, étude, etc.)
- Stop words, quels mots garder ?
  Mots très courants peu informatifs (le, un à, de).

C'est de l'art et du réglage... Dans ce qui suit : introduction / sensibilisation aux problèmes.

Identification de la Langue

Comment trouver la langue d'un document ?

- Méta information (dans l'entête HTTP p.e.): pas fiable du tout.
- Par le jeu de caractères, pas assez courant!

한글
カタカナ
ⵜⵉⴼⵉⵏⴰⵖ
Għarbi
þorn

Identification de la Langue

Comment trouver la langue d'un document ?

- Méta information (dans l'entête HTTP p.e.): pas fiable du tout.
- Par le jeu de caractères, pas assez courant!

  한글
  カタカナ
  ދިވެހި
  Għarbi
  Þorn

  Respectivement: Coréen, Japonais, Maldives, Malte, Islandais.
- Par extension: séquences de caractères fréquents, (*n*-grams)
- Par techniques d'apprentissage (classifiers)

Des librairies font ça très bien (e.g., Tika, `http://tika.apache.org`)

Tokenisation

Principe

Séparation du texte en tokens (“mots”)

Pas du tout aussi facile qu’on le dirait !

- Dans certaines langues (Chinois, Japonais), les mots ne sont pas séparés par des espaces.
- Certaines langues s’écrivent de droite à gauche, de haut en bas.
- Que faire (et de manière cohérente) des acronymes, élisions, nombres, unités, URL, email, etc.

Tokenisation

- Mots composés : les séparer en *tokens* ou les regrouper en un seul ?
  1. Anglais : *hostname*, *host-name* et *host name*, ...
  2. Français : Le Mans, aujourd'hui, pomme de terre, ...
  3. Allemand : *Levensversicherungsgesellschaftsangestellter* (employé d'une société d'assurance vie)

  Que faire si l'utilisateur cherche *hostname* et qu'on a normalisé en *host-name* ?

Majuscules, ponctuation ? Une solution simple est de normaliser (minuscules, pas de ponctuation).

Exemple pour notre petit jeu de données

On met en minuscules, on retire la ponctuation.

d_1	le loup est dans la bergerie
d_2	le loup et les trois petits cochons
d_3	les moutons sont dans la bergerie
d_4	spider cochon spider cochon il peut marcher au plafond
d_5	un loup a mangé un mouton les autres loups sont restés dans la bergerie
d_6	il y a trois moutons dans le pré et un mouton dans la gueule du loup
d_7	le cochon est à 12 euros le kilo le mouton à 10 euros le kilo
d_8	les trois petits loups et le grand méchant cochon

On considère que l'espace est le séparateur de tokens.

Stemming (racine), lemmatization

Principe

Confondre toutes les formes d'un même mot, ou de mots apparentés, en une seule racine.

Stemming Morphologique. Retire les pluriels, marque de genre, conjugaisons, modes, etc.

- Très dépendant de la langue : *geese* pluriel de *goose*, *mice* de *mouse*
- Difficile à séparer d'une analyse linguistique ("Les poules du couvent couvent", "la petite brise la glace" : où est le verbe ?)

Stemming lexical Fondre les termes proches lexicalement : "politique, politicien, police (?)" ou "université, universel, univers (?)"

Stemming phonétique. Correction fautes de frappes, fautes orthographes

Exemple de stemming

On retire les pluriels, on met le verbe à l'infinitif.

d_1	le loup etre dans la bergerie
d_2	le loup et les trois petit cochon
d_3	les moutons etre dans la bergerie
d_4	spider cochon spider cochon il pouvoir marcher au plafond
d_5	un loup avoir manger un mouton les autres loups etre rester dans la bergerie
d_6	il y avoir trois mouton dans le pre et un mouton dans la gueule du loup
d_7	le cochon etre a 12 euro le kilo le mouton a 10 euro le kilo
d_8	les trois petit loup et le grand mechant cochon

Suppression des *Stop Words*

Principe

On retire les mots porteurs d'une information faible afin de limiter le stockage.

articles: *le*, *le*, *ce*, etc.

verbes "fonctionnels" *être*, *avoir*, *faire*, etc.

conjunctions: *that*, *and*, etc.

etc.

♣ Maintenant moins utilisé car (i) espace de stockage peu coûteux et (ii) pose d'autres problèmes ("pomme de terre", "Let it be", "Stade de France")

Autres problèmes, en vrac

Majuscules / minuscules

Lyonnaise des Eaux, Société Générale, etc.

Acronymes

CAT = *cat* ou *Caterpillar Inc.* ? M.A.A.F ou MAAF ou Mutuelle ... ?

Dates, chiffres

Monday 24, August, 1572 – 24/08/1572 – 24 août 1572 10000 ou 10,000.00 ou 10,000.00

Accents, ponctuation

résumé ou résume ou resume...

- ♣ Dans tous les cas, les même règles de transformation s'appliquent aux documents ET à la requête.

Exemple avec suppression des *stop words*

Voici une solution possible.

d_1	loup etre bergerie
d_2	loup trois petit cochon
d_3	mouton etre bergerie
d_4	spider cochon spider cochon pouvoir marcher plafond
d_5	loup avoir manger mouton autres loups etre rester bergerie
d_6	avoir trois mouton pre mouton gueule loup
d_7	cochon etre 12 euro kilo mouton 10 euro kilo
d_8	trois petit loup grand mechant cochon

On a gardé les verbes fonctionnels (être, avoir).

Plan du cours

Indexation : motivation

- Par défaut, ElasticSearch essaie d'analyser les documents qu'on lui fournit.
- Pour chaque champ, il tente de trouver le type des données dont il s'agit (entier, date, IP, texte en français, en anglais, etc.)
- Si cela suffit pour des données simples (texte brut), on souhaite généralement faire mieux
- "on connaît toujours mieux ses données qu'ElasticSearch"...
- Il faut donc entrer dans le détail de la configuration et dans approfondir notre connaissance des techniques de RI

Schéma de l'index

- Le schéma donne la liste de tous les champs d'un document
- Nombreuses options :
  - type (numérique, entier)
  - possibilité de calcul de la valeur du champ à partir d'un autre
  - traitements divers sur les valeurs du champ

Notre premier document


```
{
  "title": "Vertigo",
  "year": 1958,
  "genre": "drama",
  "summary": "Scottie Ferguson, ancien inspecteur de police, est sujet au vertige depuis qu'il a
              vu mourir son collègue. Elster, son ami, le charge de surveiller sa femme,
              Madeleine, ayant des tendances suicidaires. Amoureux de la jeune femme Scottie ne
              remarque pas le piège qui se trame autour de lui et dont il va être la victime... ",
  "country": "DE",
  "director":         {
    "_id": "artist:3",
    "last_name": "Hitchcock",
    "first_name": "Alfred",
    "birth_date": "1899"
  },
  "actors": [
  {
    "_id": "artist:15",
    "first_name": "James",
    "last_name": "Stewart",
    "birth_date": "1908",
    "role": "John Ferguson"
  },
  {
    "_id": "artist:282",
    "first_name": "Arthur",
    "last_name": "Pierre",
    "birth_date": null,
    "role": null
  }
  ]
}
```

Schéma

```
{
  "mappings": {
    "movie": {
      "_all":       { "enabled": true  },
      "properties": {
        "title":    { "type": "string"  },
        "year":     { "type":   "date", "format": "yyyy" },
        "genre":      { "type": "string" },
        "summary": { "type": "string"},
        "country": {"type": "string"},
        "director":     {
          "properties": {
            "_id": { "type":"string" },
            "last_name": { "type":"string"},
            "first_name": { "type":"string"},
            "birth_date": { "type": "date", "format":"yyyy"}
          }},
        "actors": {
          "type": "nested",
          "properties": {
            "_id": { "type":"string"}
            "first_name": { "type":"string"}
            "last_name": { "type":"string"}
            "birth_date": { "type": "date", "format":"yyyy"}
            "role": { "type":"string"}
          }
        }
      }
    }
```

Détaillons le schéma

- la liste des champs, dans l'élement "mappings"
- le champ `_all` est spécial, tout y est par défaut concaténé, de façon à permettre une recherche sur tous les champs
- la liste des **types de champ**, dans l'élément "properties".

Définition des types et de la clé

- Chaque champ utilisé dans un document doit apparaître dans un des élements
- Sinon, heureusement, ElasticSearch essaie de deviner et propose un "Dynamic Mapping"
- ElasticSearch fournit tout un ensemble de types pré-définis qui suffisent pour les besoins courants; on peut associer des options à un type
- Les options indiquent d'éventuels traitements à appliquer à chaque valeur du type avant son insertion dans l'index
- ex: type **string** (remplacé par **text** depuis 5.0)
  - on lui définit un "analyzer" "standard"
  - se charge de découper le texte en **tokens** pour une recherche plein-texte (détails plus tard)
  - retenir : cela permet d'indexer chacun des mots, et donc de faire des recherches sur toutes les combinaisons de mots

Définition des champs

```
{
        "genre": {
                "type": "string",
                "index": "no",
                "store": "true",
                "index_options": "offsets"
},
```

- Les attributs de l'élement JSON caractérisent le champ
- Le **nom** et le **type** sont les informations de base
- Ensuite, divers attributs (souvent optionnels) :
  - **index** indique simplement si le champ peut être utilisé dans une recherche;
  - **store** indique que la **valeur** du champ est **stockée** dans l'index, et qu'il est donc possible de récupérer cette valeur comme résultat d'une recherche, **sans avoir besoin de retourner à la base principale**; en d'autres termes, "store" permet de traiter l'index **aussi** comme une base de données;
  - remarquez la structure pour le réalisateur (**directors**)
  - enfin, **nested** pour les champs ayant plusieurs valeurs, soit, concrètement, un **tableau** en JSON; c'est le cas par exemple pour le nom des acteurs.

Définition des champs

Les champs **index** et **store** sont très importants

Toutes les combinaisons de valeur sont possibles :

- **index=true**, **store=false**: on pourra interroger le champ, mais il faudra accéder au document principal dans la base documentaire si on veut sa valeur;
- **index=true**, **store=true**: on pourra interroger le champ, **et** accéder à sa valeur dans l'index;
- **index=false**, **store=true**: on ne peut pas interroger le champ, mais on peut récupérer sa valeur dans l'index;
- **index=false**, **store=false**: n'a pas de sens à priori; le seul intérêt est d'ignorer le champ s'il est fourni dans le document Solr.

Définition des champs (suite)

Comment peut-on indexer un champ sans le stocker ?

- c'est notamment le cas pour les textes qui sont décomposés en **termes**: chaque terme est indexé indépendamment
- très difficile pour l'index de reconstituer le texte
- d'où l'intérêt de conserver ce dernier dans son intégralité, à part

C'est une question de compromis:

- **stocker** une valeur prend plus d'espace que **l'indexer**
- Dans la situation la plus extrême, on dupliquerait la base documentaire en stockant chaque document **aussi** dans l'index
- un stockage plus important dégrade les performances

Champ par défaut, document original

Le champ `_source` contient le document, mais n'est pas indexé. Il est possible ainsi de récupérer la totalité du document passé à l'indexation. Ce comportement par défaut peut être désactivé, totalement ou partiellement.

Le squelette de schéma comprend un champ *par défaut*, le champ `_all`.

Par défaut, tous les champs sont copiés dans ce champ, analysé avec un découpage simple (espaces), puis indexé mais pas stocké. Cela permet de retrouver les documents contenant une valeur, sans savoir dans quel champ elle se trouve.

Champ calculé

Il est possible de créer des champs `all` spécifiques, correspondants à des besoins précis, à l'aide d'une instruction `copy_to`

- Ainsi, le nom et le prénom peuvent être regroupés au sein d'un champ "nom complet".
- Cela permet aussi d'indexer certains champs plusieurs fois, de différentes façons.
- par exemple le contenu d'un titre est indexé comme une chaîne de caractères dans le champ **title**, et comme un texte "tokenisé" quand on le copie dans le champ **text**;

Exemple de champ calculé

```
{
  "mappings": {
    "my_type": {
      "properties": {
        "first_name": {
          "type": "text",
          "copy_to": "full_name"
        },
        "last_name": {
          "type": "text",
          "copy_to": "full_name"
        },
        "full_name": {
          "type": "text"
        }
      }
    }
  }
}
```

Schéma

On peut stocker le schéma dans un fichier json et créer l'index dans ElasticSearch comme ceci :

```
curl -XPUT 'localhost:9200/monindex' -H 'Content-Type: application/json' -d movie-schema-2.4.json
```

Mise à jour de l'index

- Avec ElasticSearch, il est (plus) difficile de faire évoluer un schéma (qu'avec BDD classique)
- Sauf rares exceptions, il faut en général créer un nouvel index et réindexer les données
- ElasticSearch permet la copie d'un index vers l'autre avec l'API `reindex`

```
 curl -XPOST 'localhost:9200/_reindex?pretty' -H 'Content-Type: application/json' -d'
{
  "source": {
    "index": "nfe204"
  },
  "dest": {
    "index": "new_nfe204"
  }
}'
```

Définir une chaîne d'analyse avec Elasticsearch

- Les analyseurs sont composés d'un tokenizer, et TokenFilters (0 ou plus)
- Le tokenizer peut être précédé de CharFilters
- les filtres (filter) examinent les tokens un par un et décident de les conserver, de les remplacer par un ou plusieurs autres;
- l'analyseur est une chaîne de traitement (pipeline) constituée de tokeniseurs et de filtres.

Définir une chaîne d'analyse avec Elasticsearch

```
{
  "settings": {
    "analysis": {
      "filter": {
        "custom_english_stemmer": {
          "type": "stemmer",
          "name": "english"
        }
      },
      "analyzer": {
        "custom_lowercase_stemmed": {
          "tokenizer": "standard",
          "filter": [
            "lowercase",
            "custom_english_stemmer"
          ]
        }
      }
    }
  },
  "mappings": {
    "test": {
      "properties": {
        "text": {
          "type": "string",
          "analyzer": "custom_lowercase_stemmed"
        }
      }
    }
  }
}'
```

tester les analyseurs

```
{
  "analyzer": "english",
  "text": "j'aime les couleurs de l'arbre durant l'été"
}
{
  "analyzer": "french",
  "text": "j'aime les couleurs de l'arbre durant l'été"
}
{
  "analyzer": "standard",
  "filter":  [ "lowercase", "asciifolding" ],
  "text": "j'aime les COULEURS de l'arbre durant l'été"
}
```

Plan du cours

La semaine prochaine

- Classer les résultats pour des performances encore meilleures
  - Plusieurs métriques
  - PageRank
  - Classement avec ElasticSearch