Évaluation d’outils d’extraction d’entités nommées

Présenté par:
Wafa DJERAD
Sawdiatou SAMB
Maryam Soheilimajd

Plan

Introduction generale

1/ Le modèle doremus
- **1.1 Technologies du web sémantique**
- **1.2 Extractions d'entité**

2/ Etude techniques
- **2.1 Les données**
- **2.2 construction du benchamark**
- **2.3 Approche d'extraction d'entités**
  - **2.3.1 Polyglot**
  - **2.3.2 Stanford**
  - **2.3.3 TextRazor**
  - **2.3.4 OpenNLP**
- **2.4 Mesure d'évaluation des systèmes de NER**

3/ Implémentation

4/ Analyse

5/ Conduite de projet

Conclusion et perspective

Introduction

DOREMUS, projet touchant la recherche sur le web sémantique, a comme objectif de développer des outils et des méthodes pour publier, partager, connecter, enrichir les catalogues d'oeuvres et d' événements musicaux dans le web de données. Son objectif global est de faciliter l'accès aux données et connaissances musicales via la construction d'un modèle de connaissance commun (une ontologie).

Les données sur lesquelles nous travaillons sont sous la forme de graphe RDF parfaitement structurées et sémantisées selon le modèle musicale DOREMUS.

Dans le but de créer une relation d'équivalence entre des œuvres provenant de deux institutions différentes, plusieurs approches et méthodes ont été mises en place. Naturellement la première approche, c'est de les comparer directement puisqu'ils ont été représentés suivant le même modèle. C'est-à-dire comparer d'une part les classes, qui sont des ressources ou des données brutes, entre elles et d'autres part les propriétés ou prédicats.

Introduction

Ils s’avèrent que quelques problèmes ont été rencontrés dans cette étape, parmi lesquels la comparaison des classes de type string. Ces dernières sont des données brutes, non structurées donc difficilement comparable au niveau informatique.

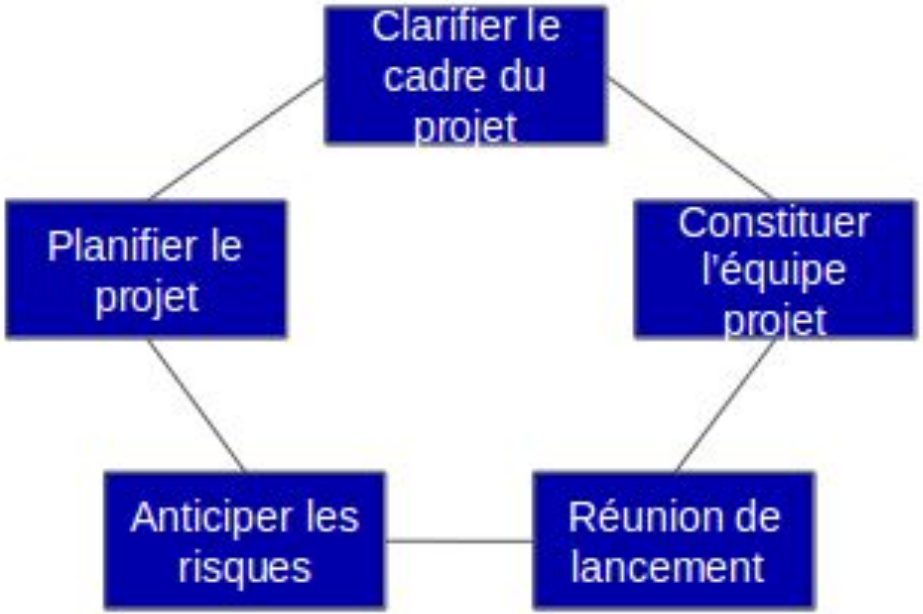
Notre travail consiste à extraire ces données qui représentent une description importante de l’œuvre et de les structurer. Pour y parvenir, des méthodes de reconnaissance d’entité nommée, en partie, répondent à une telle problématique. Elles consistent à reconnaître puis extraire à partir des données textuelles, les termes utilisés dans un contexte spécifique.

La problématique

Ils s’avèrent que quelques problèmes ont été rencontrés dans cette étape, parmi lesquels la comparaison des classes de type string. Ces dernières sont des données brutes, non structurées donc difficilement comparable au niveau informatique.

Notre travail consiste à extraire ces données qui représentent une description importante de l’œuvre et de les structurer. Pour y parvenir, des méthodes de reconnaissance d'entité nommée, en partie, répondent à une telle problématique. Elles consistent à reconnaître puis extraire à partir des données textuelles, les termes utilisés dans un contexte spécifique.

Organisation du travail



Organisation du travail

Dans la phase 1 : périmètre du projet.

- comprendre le contexte du travail
- faire des recherches sur les éléments clés du sujet.

Phase 2:formulation du projet

- définir les travaux à réaliser
- préciser les objectifs à atteindre
- les stratégies: le choix des langages de programmation, la durée d'apprentissage des technologies du web sémantique.

Phase de planification:

- identifié et ordonnancé les tâches à réaliser.
- déterminé les profils nécessaires à leur réalisation

Organisation du travail

Établir un plan d’action
déterminer les séquencements et le parallélismes de l’exécution des tâches

Aménagé une marge de flexibilité pour l’ordonnancement de chaque tâche en prévision des risques qu’on pourrait rencontrer dans l’évolution du projet.

De plus, nous avons rajouté des contraintes, c’est à dire qu’on a associé à chaque tâche les dates au plus tôt et les dates au plus tard de l’exécution de la tâche.

Enfin, nous avons modélisé le réseau de dépendance entre tâches sous forme graphique (Planning GANTT).

Web sémantique

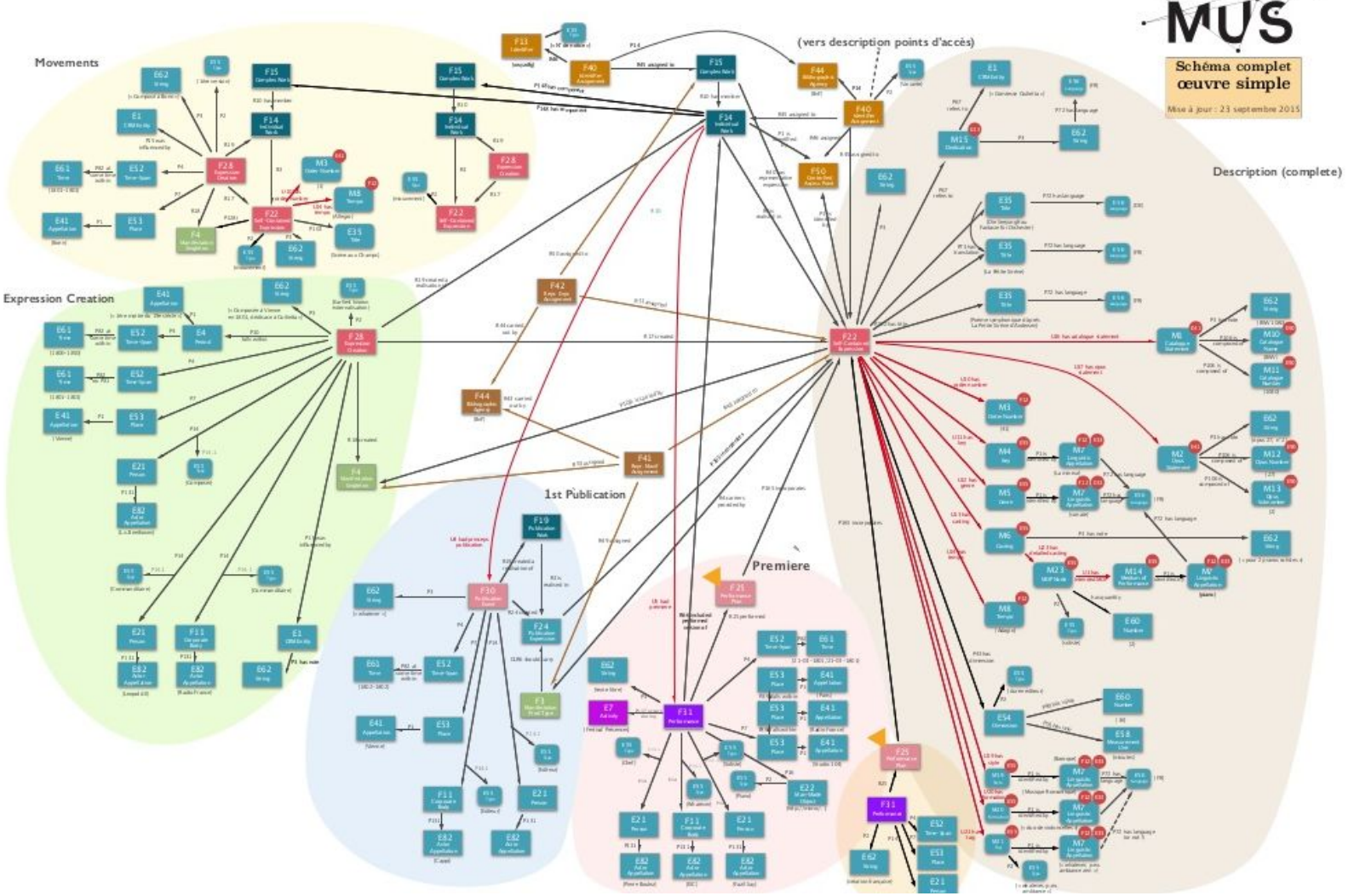
- Évolution du Web pour rendre l'information plus accessible
- Un vaste espace de documents semi-structurés (XML)
- Décrire le contenu avec un formalisme à base de connaissances (RDF)
- Utiliser des ontologies communes (RDF Schema, OWL) pour annoter les documents

Le modèle (ontologie) DOREMUS

Le modèle DOREMUS est un modèle basé sur :

- ➔ le modèle FRBRoo (**un modèle conceptuel de données bibliographiques**)
- ➔ et CIDOC CRM (**un modèle conceptuel de** l'**information muséographique**) .

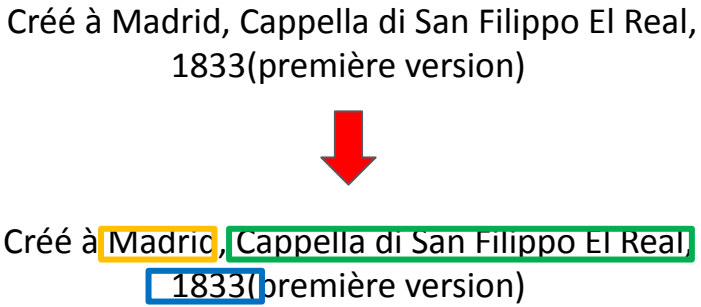
Les classes et les propriétés expriment les éléments importants dans la description des oeuvres musicales.


MUS
Schéma complet
œuvre simple
Mise à jour : 23 septembre 2015
Movements
(vers description points d'accès)
Description (complete)
Expression Creation
1st Publication
Premiere

La reconnaissance d’entités nommées

NE implique l'identification de tokens dans les textes et les classifie dans un ensemble de catégories d'intérêt prédéfinies.

- **Trois catégories universellement acceptées:**
  personne, **lieu** et **organisation**
- **Autres tâches courantes:**
  reconnaissance des expressions **date** / heure, adresses email, etc.
- **Autres entités spécifiques au domaine:**
  noms de médicaments, références bibliographiques etc.



Les données

Les données textuelles faisant l’objet du traitement sont identifiées dans les classes E62 de type String du modèle. Ce sont des notes ou commentaires rédigées indépendamment par différents experts du domaine et correspondent à la valeur d’une propriété P3_has_note du graphe.

Corpus de référence et d'évaluation

annoter chaque unité du texte, avec un jeu d'étiquettes sémantiques générales

Le Benchmark comporte 223 oeuvres.
Chaque oeuvre se présente selon le modèle illustré ci dessous et comporte donc :

- Un titre (facultatif)
- Des informations relatives à son origine : URI
- Des informations de catégorisation, sous forme de TAG ou mots-clés
- Un contenu sous forme de paragraphes.

Titre = {

URI :

Paragraphes entités|TAGS : |LOCATION |PERSON |ORGANIZATION |DATE

}

Langage de programmation et outil	Format d'annotation
Données brutes	d24 = { http://data.doremus.org/expression/0759843a-eb24-3294-a1bc-3a31ca26c07b : 19 avril 1967 à Berlin par Siegfried Palm, violoncelle (dédicataire de l'oeuvre), orchestre de la Radio de Berlin sous la direction de Henrick Czyz.}
Python (Polyglot-NER)	d24 = { http://data.doremus.org/expression/0759843a-eb24-3294-a1bc-3a31ca26c07b :19 avril 1967 à <LOCATION> Berlin </LOCATION> par <PERSON> Siegfried Palm </PERSON>, violoncelle (dédicataire de l'oeuvre), orchestre de <ORGANIZATION> la Radio de <LOCATION> Berlin</LOCATION> </ORGANIZATION> sous la direction de <PERSON> Henrick Czyz </PERSON>.}
Java (Stanford & TextRazor & Open NLP)	24 = { http://data.doremus.org/expression/0759843a-eb24-3294-a1bc-3a31ca26c07b : 19/DATE avril/DATE 1967/DATE à Berlin/LOCATION par Siegfried/PERSON Palm/PERSON, violoncelle (dédicataire de l'oeuvre), orchestre/ORGANIZATION de/ORGANIZATION la/ORGANIZATION Radio/ORGANIZATION de/ORGANIZATION Berlin/ORGANIZATION sous la direction de Henrick/PERSON Czyz/PERSON.}

Choix de l’outil

Après avoir réalisé une étude comparative des différents outils NER, nous avons été amené à faire un choix des méthodes qu’on va tester parmi plusieurs existantes dans la littérature.

- taux et qualité de reconnaissance manifestement raisonnables
- une offre gratuite minimale non limitée dans le temps.
- désambiguïsation des entités nommées basée sur des ressources exploitables et existence.
- Reconnaissance d’entité en langue française.
- Disponibilité des ressources et de la documentation ….

Évaluation des systèmes de REN

avril 1967 à **Berlin** par **Siegfried Palm**, dédicataire de l'oeuvre → avril 1967 à **Berlin** par **Siegfried Palm**, dédicataire de l'**oeuvre**

Date: 1
Person: 1
Location: 1
Organization: 0

Person: 1
Location: 1
Organization: 1

Calculer l'erreur de suppression d'entités et les erreurs d'insertion d'entités

$$\textbf{Précision} = \frac{VP}{(VP + FP)} \qquad \textbf{Rappel} = \frac{VP}{(VP + FN)} \qquad \textbf{F-mesure} = 2 \times \frac{P + R}{P \times R}$$

Outils de NER Sélectionnés

Polyglot NER

- 40 langues principales.
- Formé sur les articles Wikipédia
- Entités: Personne, Lieu, Organisation
- Approche d'apprentissage automatique (Word embedding).

OpenNLP

- 5 langues principales
- basée sur des algorithmes d'apprentissage automatique (l'entropie maximale et Perception) qui sont sous une forme de régression logistique multinomiale
- Entités: Personne, Lieu, Organisation

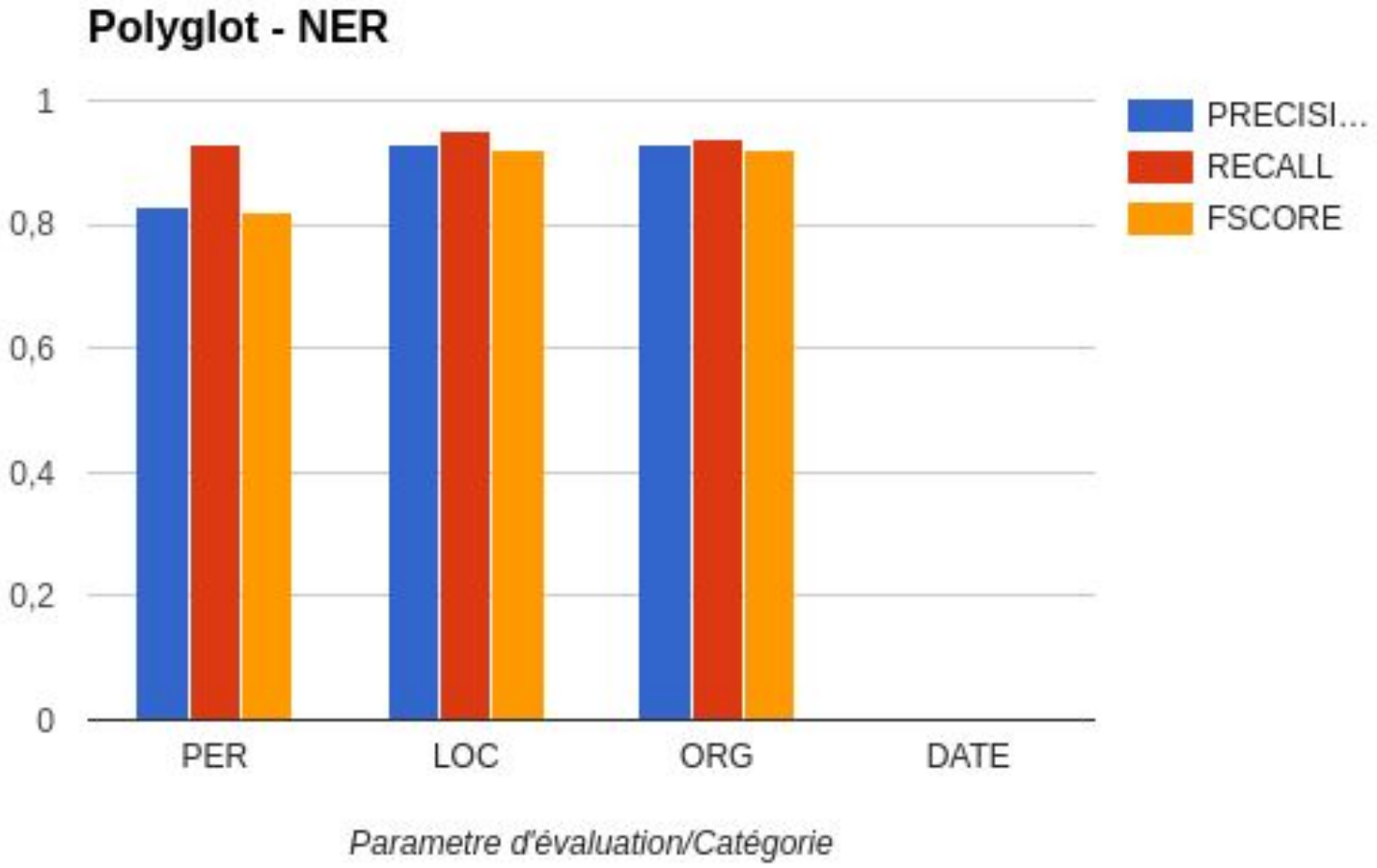
Stanford NER

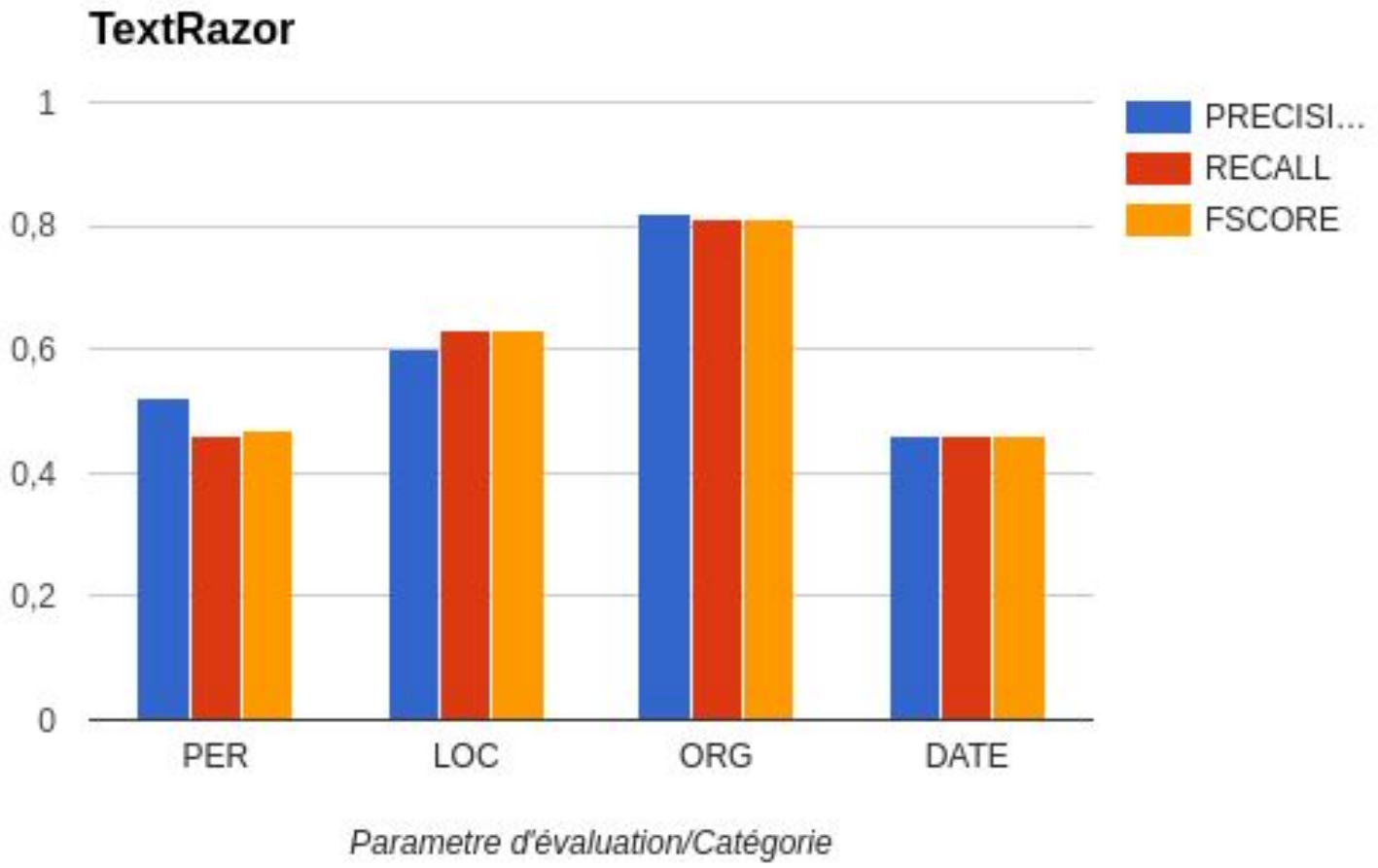
- Formé sur CoNLL, MUC et ACE
- Entités: Personne, Lieu, Organisation, Money, Percent, Date, Time
- Modèles avec et sans les caractéristiques de similarité de distribution

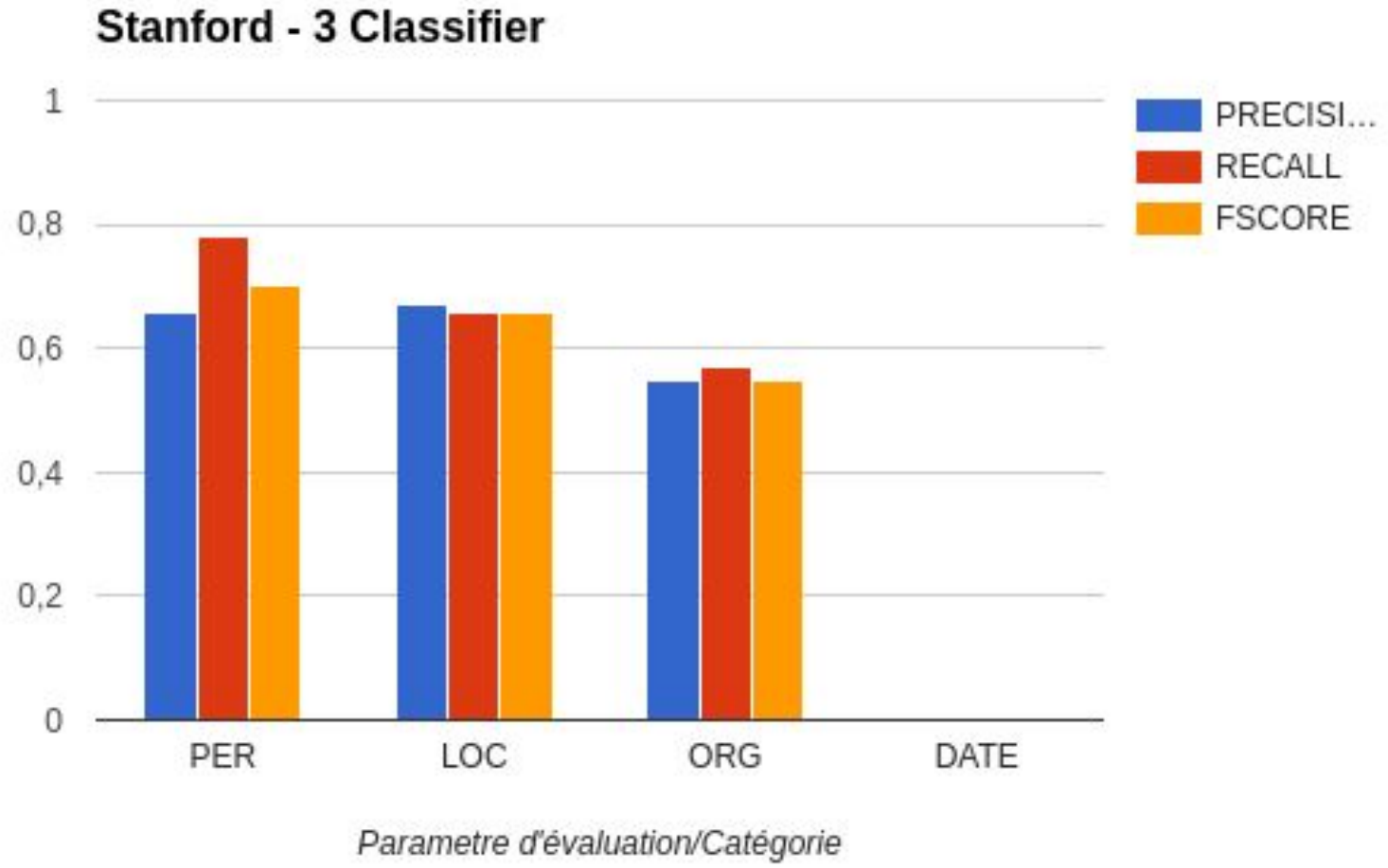
TextRazor

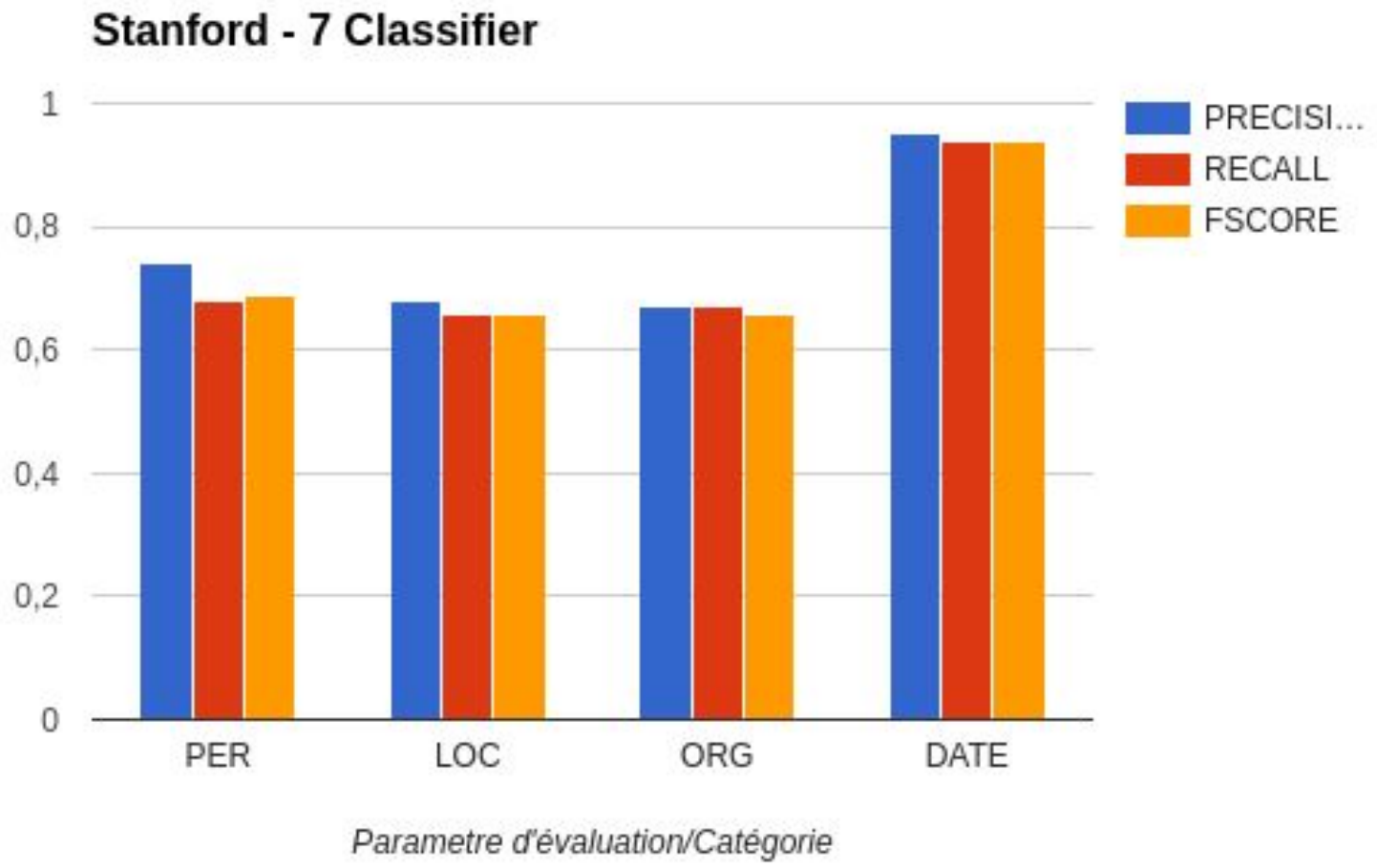
- Formé sur diverses sources Web, y compris Wikipedia, DBPedia et Wikidata.
- Entités: Personne, Lieu, Organisation, les adresses e-mail et les sites Web.
- Modèles avec et sans les caractéristiques de similarité de distribution

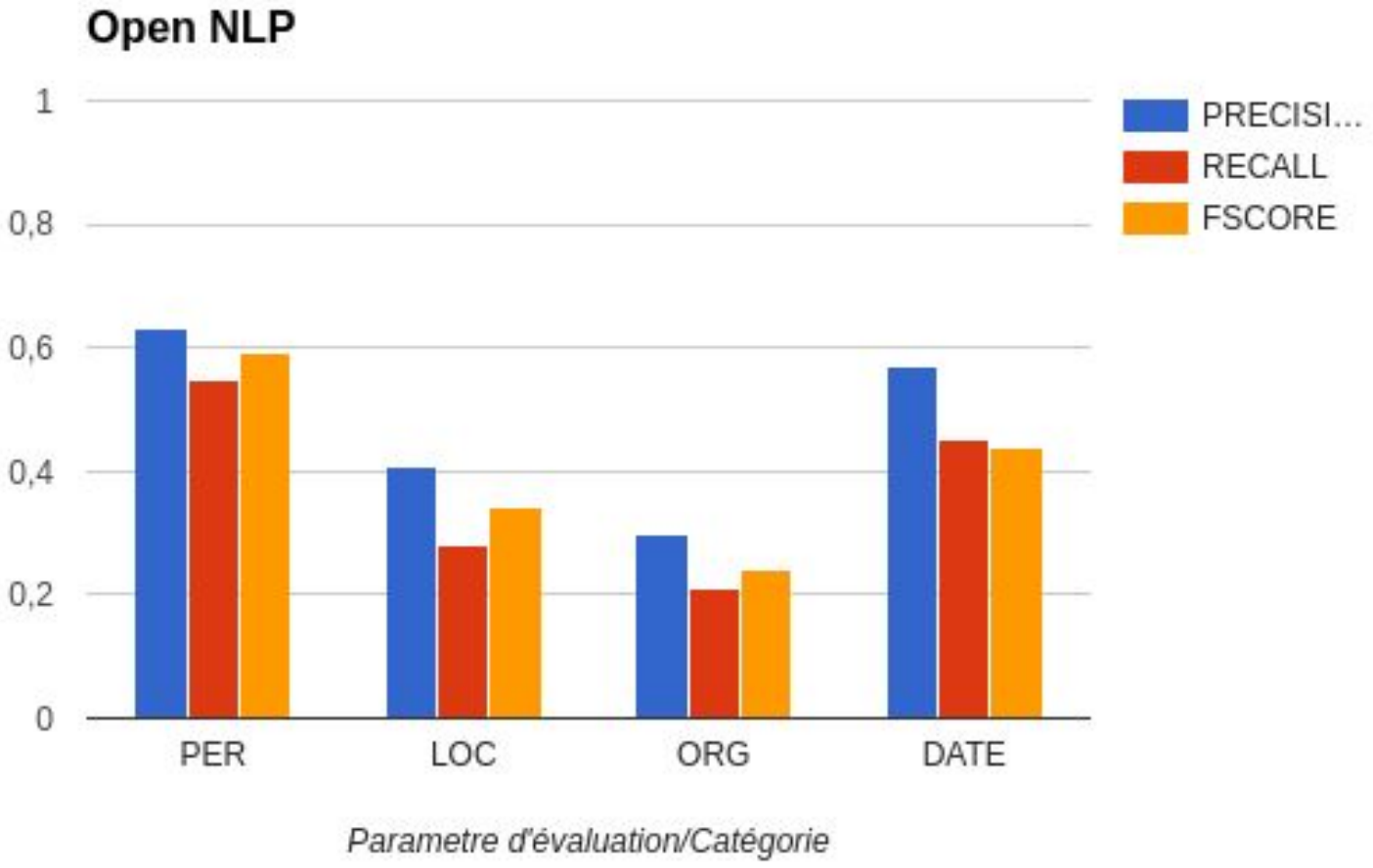
Résultat de l’évaluation


Polyglot - NER
1
0,8
0,6
0,4
0,2
0
PER
LOC
ORG
DATE
PRECISI...
RECALL
FSCORE
Parametre d'évaluation/Catégorie


TextRazor
1
0,8
0,6
0,4
0,2
0
PRECISI...
RECALL
FSCORE
PER
LOC
ORG
DATE
Parametre d'évaluation/Catégorie


Stanford - 3 Classifier
1
0,8
0,6
0,4
0,2
0
PRECISI...
RECALL
FSCORE
PER
LOC
ORG
DATE
Parametre d'évaluation/Catégorie


Stanford - 7 Classifier
1
0,8
0,6
0,4
0,2
0
PRECISI...
RECALL
FSCORE
PER
LOC
ORG
DATE
Parametre d'évaluation/Catégorie


Open NLP
1
0,8
0,6
0,4
0,2
0
PRECISI...
RECALL
FSCORE
PER
LOC
ORG
DATE
Parametre d'évaluation/Catégorie

Résultat

Catégorie / outil	Précision	Rappel	F_ mesure
Person / Stanford	0,81	0,81	0,80
Organisation / TextRazor	0,82	0,81	0,81
Location / Polyglot	0,84	0,94	0,89
Date	0,94	0,93	0,94

conclusion

NER est une technologie utile

Ensuite, nous avons mené nos expérimentations en utilisant le corpus avec les quatre méthodes choisies préalablement : TextRazor, Polyglot, Stanford et OpenNLP.

A la fin de l'analyse, les trois premiers seront retenus en fonctions des catégories d'entités pour être appliqué sur l'ensemble des données mise a disposition via le endpoint SPARQL DOREMUS.

Parmi les difficultés que nous avons rencontré, nous pouvons distinguer le manque de documentation, l'indisponibilité des codes sources, la création du benchmark qui nécessitait une bonne culture générale et aussi faut reconnaître que c'était un domaine totalement inconnu de tous les membres de l'équipe.

Perspective

mettre en place une application regroupant les meilleurs classifieurs pour chaque catégorie.

Ce sera l’occasion d’essayer d’implémenter des dictionnaires personnalisées et des modèles d’extraction hybrides.

Annexe

Annexe 1

Le **word embedding** est une méthode d'apprentissage automatique

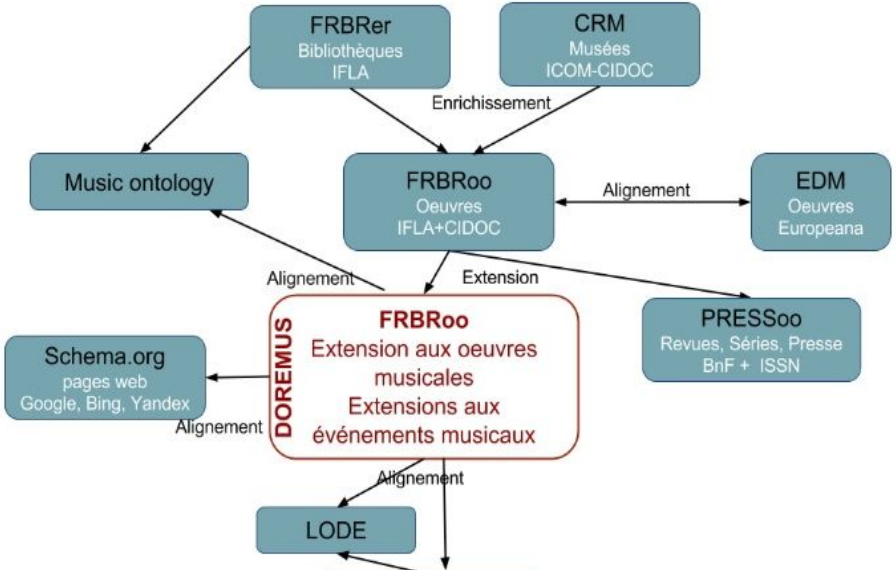
se focalisant sur l'apprentissage d'une représentation de mots.

Cette technique permet de représenter les mots d'un dictionnaire par des vecteurs afin de faciliter leur analyse sémantique et syntaxique.

Ainsi, chaque mot sera représenté par un vecteur de réels et les mots apparaissant dans des contextes similaires auront des vecteurs plus proches que d'autres apparaissant dans des contextes différents.

Ce qui permet de diminuer considérablement l'espace de dimensionnalité (car on ne stocke plus un dictionnaire entier mais uniquement un espace de vecteurs continus).

Les modèles : FRBRoo étendu



Expression régulière benchmark

```
import re

ENTITY_LIST = [ "person", "location", "organization"]

for entity in ENTITY_LIST :

   regex = ur"\<"+entity.upper()+"\> (.+?) \</"+entity.upper()+"\>+?"

   line = """ d223 = { http://data.doremus.org/expression/dcfd1518-e107-3a6b-b3c8-303872b2263d :Texte tiré des Lamentations de <PERSON> Jérémie </PERSON>. Comprend : 1- Prophecy. 2- Profanation. 3- Lamentation. Durée d'exécution : 24 minutes environ} """

   person = re.findall(regex, line)

   print(person)
```

Précision - Rappel - fscore

```
import bcubed

bdict = {
   "PER": set(['Jérémie']),
 }

cdict = {
    "PER": set(['Jérémie','duree']),
}


precision = bcubed.precision(cdict, bdict)
recall = bcubed.recall(cdict, bdict)
fscore = bcubed.fscore(precision, recall)

fscore = bcubed.fscore(precision, recall, beta=2.0)  # weights recall higher
fscore = bcubed.fscore(precision, recall, beta=0.5)  # weights precision higher

print (precision)
print (recall)
print(fscore)
```

Polyglot Extraction d’entités

```
from polyglot.text import Text

outfile = open("/home/wafa/Bureau/r3.txt", 'w')

with open("/home/wafa/PycharmProjects/untitled/TER Music/d3.txt", 'r') as infile:

    for line in infile:

        s = Text(line, hint_language_code="fr")
        for sent in s.sentences:
            for entity in sent.entities:
                tag = entity.tag, entity
                print(tag)
                s = str(tag)
                outfile.write(s)
```