Université Claude Bernard Lyon 1

Rapport de TER :

Reconnaissance d'entités géographiques dans des documents textuels

Réalisé par : **Jonathan BROU** Encadrant : **Fabien DUCHATEAU**

Du 18/01 au 3/03/2016

Résumé : Le Web est devenu une source riche en informations, seulement, celles-ci ne sont pas structurées et sont uniquement présentes sous forme textuelle. Le but de ce projet de recherche a été de développer un outil capable de reconnaître les entités géographiques dans un référentiel à partir de leurs mentions dans un texte et de comparer leur résultat avec des outils existants. Dans ce projet nous nous sommes limités aux entités spatiales afin de créer un outil plus efficace que ceux existants qui sont eux plus généralistes.

Mots-clés : Reconnaissance d'entités, Entity Linking, Geonames, DBPedia, POSTagging, NED, NER

Table des matières

1. Introduction

Ce projet s'inscrit dans le cadre de l'UE Mif20 « Projet de recherche « qui s'est déroulé du 18 janvier au 3 mars 2015. Le sujet de mon TER est « Reconnaissance d'entités géographiques dans des documents textuels » et celui-ci a été encadré par Fabien DUCHATEAU.

Notre approche est basée sur l'Entity Linking qui est une évolution du NER (Named entity recognition). Les outils NER permettent uniquement de reconnaître les mentions d'entités dans un texte ainsi que leur catégorie (Nom de personne, nom d'organisation ou d'entreprise, quantités, distances, valeur, dates, etc…). Contrairement aux outils utilisant l'Entity Linking, les mentions ne sont pas liées à une entité dans une base de connaissance.
Les mentions sont la représentation d'une entité dans un texte. Il peut s'agit d'un mot ou d'un groupe de mots.

L'une des difficultés pour reconnaitre la bonne entité est qu'une mention d'entité dans un texte peut correspondre à plusieurs entités dans le référentiel (par exemple, la mention « Washington « peut correspondre à la fois à la capitale des Etats-Unis, mais aussi à l'état de Washington).
Mais également qu'une même entité peut avoir plusieurs mentions correspondantes (par exemple la ville de Lyon peut à la fois correspondre aux mentions « Lyon », son surnom « la Ville des Lumières » ou son nom romain « Lugdunum »).

2. Travaux existants

De nombreux outils existent déjà et nous permettent de reconnaitre les entités correspondantes à leur mention dans un document textuelle. Il y a par exemple AlchemyAPI [2], Textrazor [3] ou encore DBPedia Spotlight [4]. Contrairement à notre outil, ces outils renvoient l'ensemble des entités reconnus dans le texte au lieu de reconnaitre uniquement les entités spatiales.

3. Approches

Comme les autres approches décrites dans l'article Survey [1], plusieurs étapes sont nécessaires à mon approche pour reconnaitre des entités dans un texte.

3.1 Détection de mentions

Le but de cette étape sera de détecter les mentions d'entités dans le texte. Pour réaliser cette tâche nous utilisons RDRPOSTagger [7], un outil de POSTagging qui permet de reconnaître la nature des mots dans une phrase. Ainsi, une succession de mots sera considérée comme une probable mention

d'entité si elle contient uniquement des noms communs, des noms propres, des déterminants, des prépositions, des articles, des nombres ou des adjectifs. Elle devra contenir au moins un nom propre pour qu'elle soit considérée comme une mention d'entité. Seulement, cette méthode ne nous permet pas d'identifier la catégorie de la mention de l'entité, ce qui nous aurait permis d'éliminer les entités non spatiales. Par exemple la phrase « La basilique de Notre-Dame de Fourvière se trouve à Lyon » nous donne le résultat suivant :

La	**basilique**	**de**	**Notre-Dame**	**de**	**Fourvière**	**se**	**trouve**	**à**	**Lyon.**
Dét	NC	Prép	NP	Prép	NP	PP	Verb	Prép	NP

Dét : Déterminant
NC : Nom commun
Prép : Préposition
NP : Nom propre
PP : Pronom personnel
Verb : Verbe

Ce qui nous permet d'identifier les deux mentions d'entités : « **basilique de Notre-Dame de Fourvière** » et « **Lyon** ».
La sortie de cette étape est un ensemble d'entités géographiques mais aussi non-géographiques.

3.2 Interrogation des référentiels

Dans cette étape, nous cherchons l'entité correspondante dans Geonames [5] puis DBPedia [6]. Pour cela nous commençons par interroger Geonames grâce à son API. Celui-ci retournera une liste d'entités candidates à la mention donnée. Parfois interroger Geonames avec la mention complète ne nous permet pas d'obtenir de résultat. Par exemple « basilique de Fourvière » peut ne peut pas donner de résultat si l'entité est nommé « Notre-Dame de Fourvière » dans Geonames. Nous effectuerons du coup plusieurs requêtes avec pour commencer la mention complète de l'entité puis en requêtant des groupes de mots à l'intérieur de la mention en sélectionnant uniquement les noms communs, les noms propres et les adjectifs. Ainsi dans la mention « basilique de Notre-Dame de Fourvière » les requêtes suivantes seront envoyés à Geonames :

- basilique de Notre-Dame de Fourvière
- basilique Notre-Dame
- Notre-Dame Fourvière
- Basilique
- Notre-Dame
- Fourvière

On pourra ensuite retrouver l'entité équivalente sur DBPedia à celle trouvée sur Geonames en cherchant tous les objets qui ont pour valeur de la propriété « sameAs », l'URI de l'entité sur Geonames.

La requête SPARQL suivante sera utilisée pour interroger DBPedia :

```
PREFIX owl:http://www.w3.org/2002/07/owl#
SELECT ?u {
        ?u owl:sameAs <http://sws.geonames.org/{geonamesID}/>.
```

}

On remplacera {geonamesID} par l'identifiant de l'entité correspondante sur Geonames.

3.3 Tri des entités candidates

Dans cette étape nous attribuerons une note aux entités candidates obtenues à l'étape précédente puis nous les trierons en fonction des scores et enfin nous éliminerons les entités ayant un score trop faible.

Notre approche est d'analyser le contexte autour de l'entité dans le texte pour la comparer avec la description de l'entité candidate dans DBPedia [6] afin de lui donner une note. Pour cela, l'outil que nous avons conçu va sélectionner tous les mots à une distance maximum de 10 mots autour de la mention. Pour chacun de ces mots, le score sera incrémenté de 1 s'il apparaît dans la description sur DBPedia [6]. Le score sera également incrémenté de 3 pour chaque mot de la mention qui est contenu dans la description. Enfin le score sera incrémenté de 20/(nombre de mots dans l'entité candidates).

On tri alors les entités candidates avec en premier celle ayant le meilleure score puis on élimine toutes celles qui ont un score trop faible.

4. Implémentation

Notre outil se présente sous la forme d'une interface Web gérée par un serveur Tomcat [13]. Notre serveur est développé en Java à l'aide du framework Spring [12]. Nous utilisons Maven [14] pour gérer la production et les dépendances de notre projet. Pour le développement, l'IDE Netbeans a été utilisé.

Pour comparer les résultats des outils existants (AlchemyAPI, TextRazor et DBPedia SpotLight), nous utilisons leur API. Cela permet de récupérer les entités reconnus par ces derniers à partir d'un même document textuel.

Tous les calculs se font côté serveur. Celui est composé de :

-L'API de TextRazor sous forme de dépendance Maven.

-L'API de AlchemyAPI qui n'était pas disponible avec Maven, nous avons dû télécharger le code afin de l'intégrer dans notre projet.

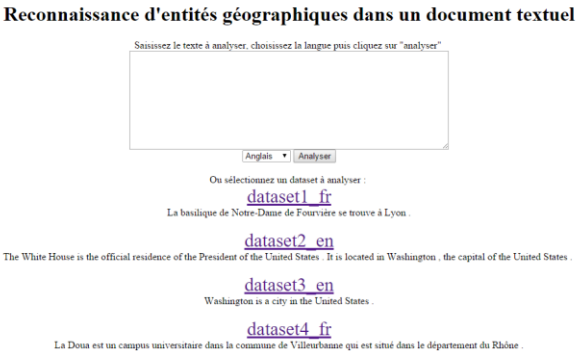
-L'API de Geonames WebClient [15] sous forme de dépendance Maven.

-L'API de RDRPosTagger [7]qui a été un peu modifié pour fonctionner avec le projet.

-L'outil développé en Java.

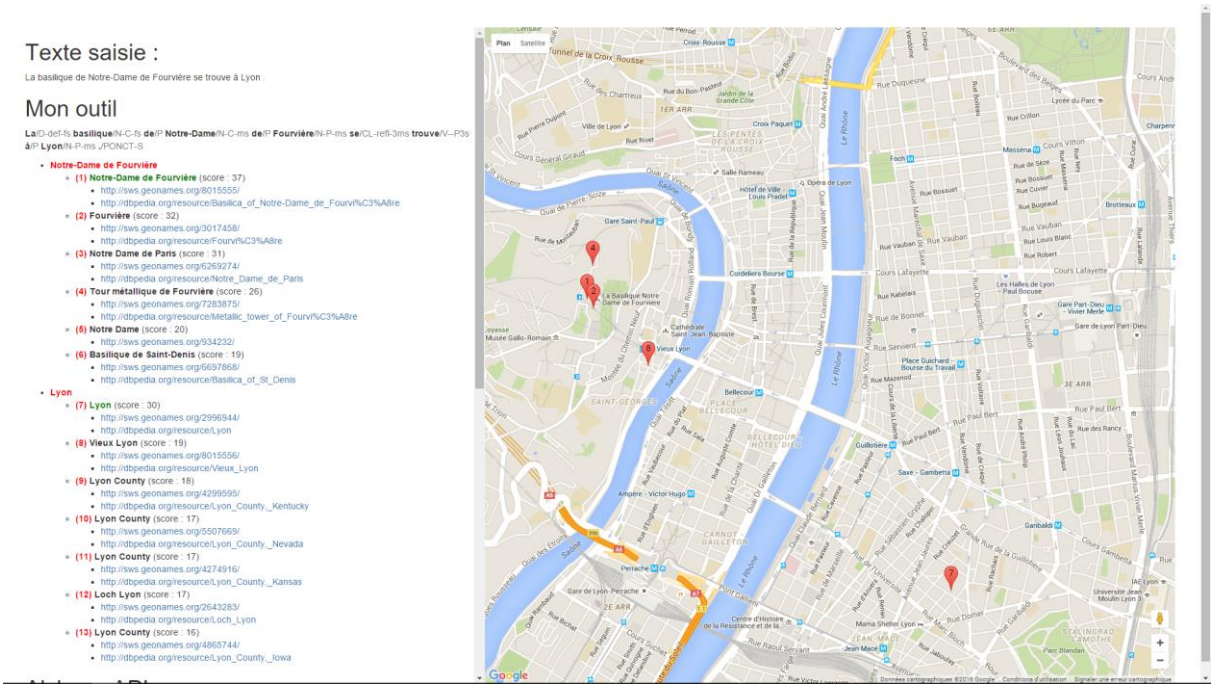
L’interface Web est composée de deux pages.

Une page d’accueil :



Dans celle-ci, il est possible soit de saisir un texte à analyser, soit de sélectionner un Dataset avec un texte prédéfini et les entités géographiques correspondantes qui devrait être obtenu. Cela nous permet alors de comparer les différents outils en leur attribuant une note en fonction des entités correctement reconnues.

Une page de résultat :



Cette page est partagée en 2 parties avec à gauche le texte saisie, le résultat du POSTagging, et les entités candidates reconnus par les différents outils. Une carte Google Maps à droite permet de repérer les entités géographiques sur une carte. Nous utilisons pour cela l'API de Google Maps [8].

5. Expériences

5.1 Protocole

Environnement de test :

Système d'exploitation : Windows 10 Professionnel
Processeur : Intel Core i7
Mémoire vive : 8 Go
Bande passante : Environ 3Mbit/s à l'université et 190Mbit/s à domicile

Il faut noter que la qualité du système d'exploitation, du processeur ou de la mémoire vive à peu d'importance sur les performances de l'application. Seul une bande passante permet d'améliorer les performances du fait du nombre important de requête.

6. Conclusion et améliorations

Une amélioration possible serait de faire tourner l'application sur un ensemble de machine se partageant les requêtes.

7. Références

[1] W. Shen, H. Wang et J.Han. “Entity linking with a knowledge base: Issues, techniques, and solutions.”

[2] http://www.alchemyapi.com/

[3] https://www.textrazor.com/

[4] https://dbpedia-spotlight.github.io/demo/

[5] http://www.geonames.org/

[6] http://fr.dbpedia.org/

[7] http://rdrpostagger.sourceforge.net/

[8] https://developers.google.com/maps/documentation/javascript/?hl=fr

[9] https://en.wikipedia.org/wiki/Named-entity_recognition

[10] https://en.wikipedia.org/wiki/Entity_linking

[11] https://en.wikipedia.org/wiki/Part-of-speech_tagging

[12] https://spring.io/

[13] http://tomcat.apache.org/

[14] https://maven.apache.org/

[15] http://www.geonames.org/source-code/