



Projet de Web Sémantique - M2 DataScale
Sujet 1 : Extraction d'un graphe de connaissances
à partir d'un texte en langage naturel

Présenté par :

Thivagini SUGUMAR

Nadine AL HAJJ

Lynda ZAIDI

Mohamed OUMEZZAOUCHE

Encadré par :

Zoubida KEDAD

6 mars 2024

Table des matières

1. Introduction.....	2
2. Description de la méthode.....	3
2.1. Schéma.....	3
2.2. Explications.....	4
2.2.1. Prétraitement du texte.....	4
2.2.2. Identification des entités nommées.....	4
2.2.3. Identification des relations.....	7
2.2.4. Enrichissement.....	7
2.2.5. Construction des triplets.....	8
2.2.6. Construction du graphe de connaissance.....	9
2.2.7. Conversion du graphe sous format RDF.....	9
2.2.8. Stockage du graphe RDF.....	10
3. Tests et évaluation.....	11
3.1. Test 1.....	12
3.1.1. Graphe manuel.....	12
3.1.2. Graphe avec l'approche automatisé.....	12
3.2. Test 2.....	13
3.2.1. Graphe manuel.....	13
3.2.2. Graphe avec l'approche automatisé.....	13
3.3. Test 3.....	14
3.3.1. Graphe manuel.....	14
3.3.2. Graphe avec l'approche automatisé.....	14
3.4. Test 4.....	15
3.4.1. Graphe manuel.....	15
3.4.2. Graphe avec l'approche automatisé.....	15
3.5. Test 5.....	16
3.6. Conclusion évaluation.....	16
4. Etude des articles.....	17
4.1. D1-1 par Lynda.....	17
4.1.1. Résumé de l'article.....	17
4.1.2. Comparaison avec notre approche.....	17
4.2. D1-3 par Thivagini.....	18
4.2.1. Résumé de l'article.....	18
4.2.2. Comparaison avec notre approche.....	18
4.3. D2-1 par Nadine.....	19
4.3.1. Résumé de l'article.....	19
4.3.2. Comparaison avec notre approche.....	19
4.4. D3-2 par Mohamed.....	20
4.4.1. Résumé de l'article.....	20
4.4.2. Comparaison avec notre approche.....	21
5. Conclusion.....	22

1. Introduction

La montée en puissance des données non structurées sur le Web et dans divers domaines de l'information a suscité un besoin croissant de méthodes et d'outils permettant de traiter et d'exploiter ces données de manière efficace. Dans ce contexte, les technologies du Web sémantique et de l'analyse du langage naturel jouent un rôle crucial en permettant de donner un sens et une structure aux données non structurées.

Ce rapport se veut être une ressource précieuse pour les chercheurs, les praticiens et les passionnés de la science des données qui souhaitent approfondir leur compréhension de l'extraction de connaissances à partir de textes non structurés et exploiter pleinement le potentiel des données disponibles sur le Web et au-delà.

En effet, il se concentre sur le domaine fascinant de l'extraction de graphes de connaissances à partir de texte. Cette approche innovante vise à convertir des informations textuelles en connaissances exploitables, représentées sous forme de graphes orientés. Ce processus repose sur des techniques avancées de traitement du langage naturel et sur les principes fondamentaux du Web sémantique, notamment le Resource Description Framework (RDF) et les ontologies.

L'objectif principal de ce rapport est de présenter notre approche d'extraction de graphes de connaissances à partir de textes non structurés, ainsi que de discuter d'autres méthodes disponibles dans la littérature. Nous mettrons en avant les bases théoriques, les techniques d'extraction d'entités nommées, et la modélisation des relations entre celles-ci, tout en explorant les applications concrètes de cette approche dans divers domaines.

En plus de notre approche, nous analyserons également d'autres méthodes existantes pour l'extraction de graphes de connaissances à partir de textes. Cette analyse comparative permettra de mettre en lumière les avantages et les défis de chaque méthode, ainsi que les opportunités d'amélioration pour les futurs développements. Enfin, des études de cas seront présentées pour illustrer l'application pratique de ces techniques dans des contextes réels.

Dans le cadre du sujet 1, nous explorons l'extraction de graphes de connaissances à partir de textes non structurés. Notre objectif principal est la génération automatique d'un graphe RDF à partir de contenu textuel en langage naturel. Nous utilisons la librairie NLP SpaCy pour extraire des entités nommées, qui servent de sommets ou de ressources dans notre graphe. En outre, nous utilisons des ontologies et des techniques de traitement du langage naturel pour identifier et extraire les liens entre ces entités, qui sont représentés sous forme d'arcs ou de propriétés dans le graphe.

2. Description de la méthode

2.1. Schéma

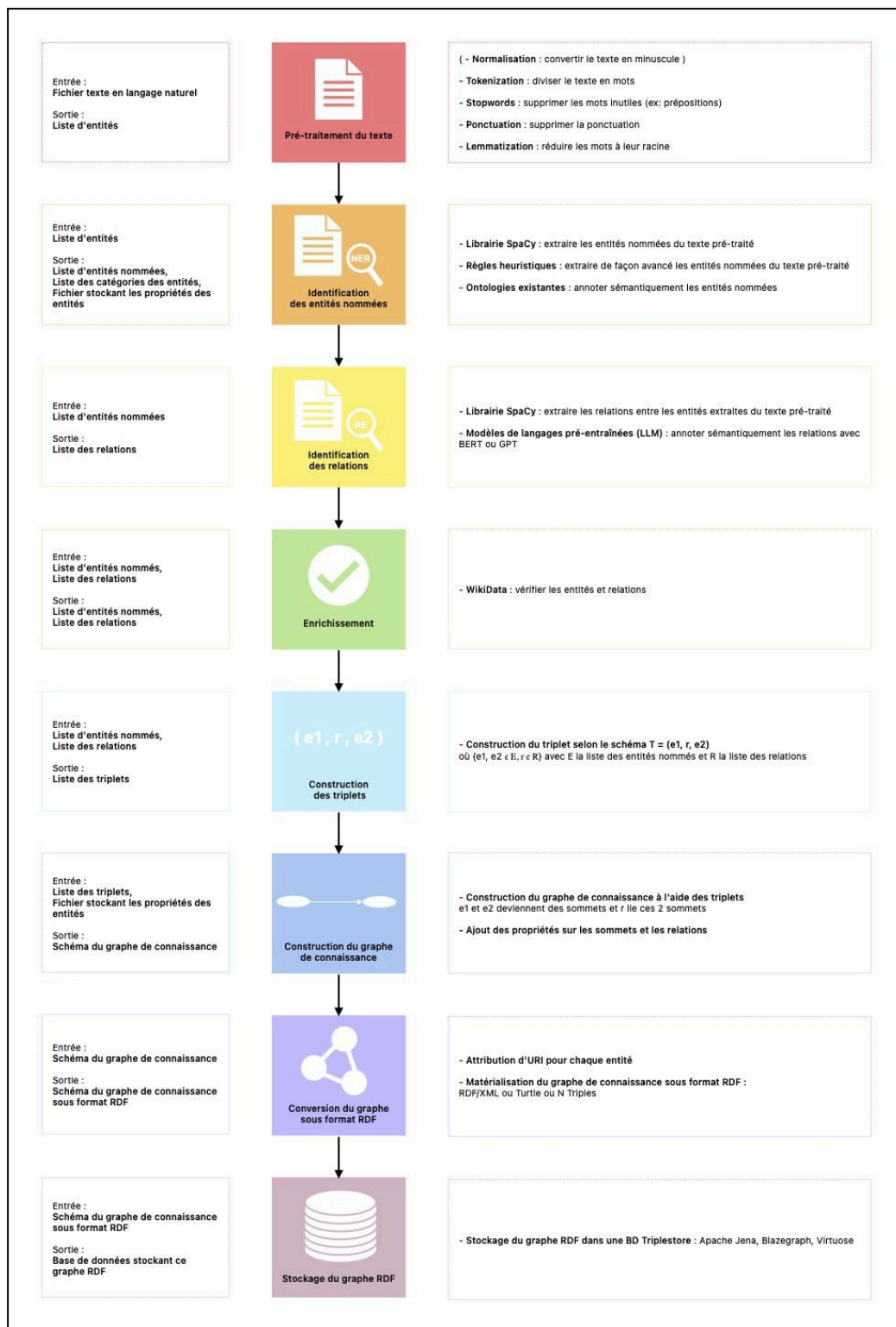


Image 1 : Schéma global de notre méthode

2.2. Explications

2.2.1. Prétraitement du texte

Le prétraitement de texte constitue une phase cruciale dans le traitement de données textuelles, jouant un rôle essentiel dans la préparation des données avant toute analyse ultérieure. Cette étape vise à nettoyer, normaliser et structurer les données textuelles afin de les rendre exploitables pour les algorithmes de traitement de langage naturel (NLP) et d'apprentissage automatique (LLM).

Cette étape de prétraitement est effectuée au commencement de chaque phase où l'on applique des techniques de traitement du langage naturel (NLP) telles que l'identification des entités nommées, l'identification des relations et la construction des triplets. Les techniques de prétraitement employées seront détaillées dans les prochaines étapes.

2.2.2. Identification des entités nommées

Après chargement du texte, on passe à l'étape d'identification des entités depuis le texte. D'après le texte, on va déduire les entités qui nous seront nécessaires pour la construction du graphe, ainsi que les labels associés à chacune. On prend en entrée le texte, on utilise la bibliothèque Spacy pour extraire les entités nommées et les retourner en sortie avec les labels associés à chacune.

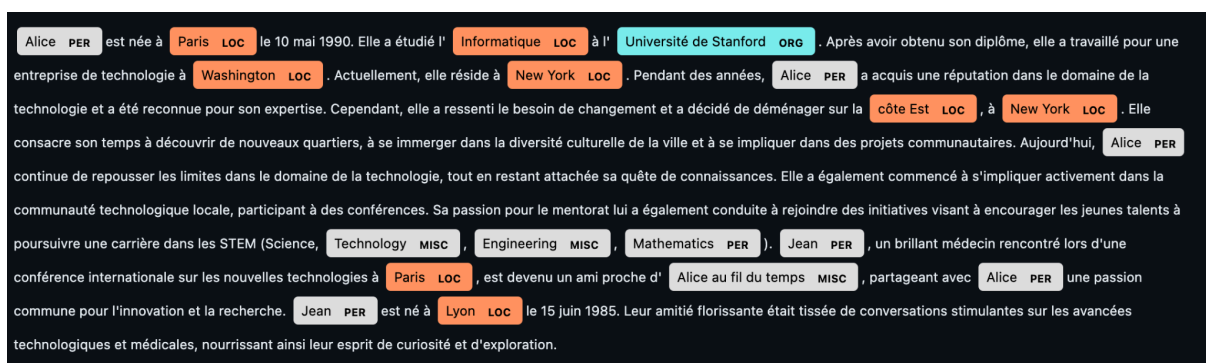


Image 2 : Identification des entités nommées avec SpaCy

Cependant, la bibliothèque SpaCy ne suffit pas à extraire toutes les entités du texte. C'est pour cela que l'on met en place des règles heuristiques basées sur la structure des phrases et la labellisation associée aux tokens afin d'extraire arbitrairement les notions dont on aura besoin pour construire nos graphes de propriété.

Tout d'abord, nous devons transformer le document de tokens extraits par SpaCy afin que la recherche d'entités soit plus simple et intuitive. En effet, nous lemmatisons le contenu du document afin d'avoir un seul référentiel pour chaque token à comparer, cela évite de devoir vérifier si toutes les variantes d'un mot sont présentes afin de vérifier son existence. On supprime également les stop words et la ponctuation.

Nous parcourons le document token par token pour les entités qui n'ont pas été extraites par SpaCy, au début nous assignons au label du token une valeur par défaut, nous allons essayer de lui assigner une valeur via les différents tests.

```
doc = nlp(" ".join([token.lemma_ for token in doc if not token.is_stop and not token.is_punct]))

# Ajouter les entités avec des règles heuristiques
for token in doc:
    ent_text = token.lemma_
    ent_label = None
```

Par exemple pour extraire une date, le test est effectué comme ceci :

```
# Ajouter la catégorie pour les jours
if token.like_num and token.nbor(1).text.lower() in ["janvier", "février", "mars", "avril", "mai", "juin", "juillet", "août", "septembre", "octobre"]:
    ent_label = 'DAY_OF_YEAR'

# Ajouter la catégorie pour les mois
if token.text.lower() in ["janvier", "février", "mars", "avril", "mai", "juin", "juillet", "août", "septembre", "octobre", "novembre", "décembre"]:
    ent_label = 'MONTH_OF_YEAR'

# Ajouter la catégorie pour les années
if token.like_num and len(token.text) == 4:
    ent_label = 'YEAR'
```

Pour le test d'assignation au label "DAY_OF_YEAR", on vérifie si le token suivant est un mois de l'année et si le token courant est un chiffre. Si c'est le cas, on attribue au token le label et on l'extrait dans le résultat. Seuls les tokens avec des labels assignés autrement que par None seront renvoyés en sortie. Ainsi pour le label "MONTH_OF_YEAR", on vérifie si la valeur du token est incluse dans la liste de mois fournie. Pour le label "YEAR", on vérifie si le token est un chiffre de taille 4.

Voici un autre exemple de test d'heuristique :

```
if token.i > 0 and token.nbor(-1).text.lower() in ["trouver"]:
    ent_label = 'GAIN'
    next_token_index = token.i + 1
    while next_token_index < len(doc) and doc[next_token_index].pos_ != "ADJ":
        ent_text += " " + doc[next_token_index].text
        next_token_index += 1
```

Si le token n'est pas le dernier token du document et que le token précédent possède la valeur "trouver", alors on assigne au token courant le label "GAIN" en vérifiant que les tokens suivants ne sont pas des adjectifs, si c'est le cas on les concatène au résultat afin de produire une heuristique de plusieurs mots.

Ainsi pour chaque label testé nous avons ce genre de vérification, si aucune vérification ne passe alors le token est laissé en suspens car il est pas reconnu. Ainsi on ne le retournera pas dans le résultat des entités trouvées.

```
*** Il y a 34 entités nommées. ***
Alice : PER
Paris : LOC
Informatique : LOC
Université de Stanford : ORG
Washington : LOC
New York : LOC
côte Est : LOC
Technology : MISC
Engineering : MISC
Mathematics : PER
Jean : PER
Alice au fil du temps : MISC
Lyon : LOC
10 : DAY_OF_YEAR
mai : MONTH_OF_YEAR
1990 : YEAR
diplôme : GAIN
entreprise technologie : NON_REAL_ENTITY
réputation domaine : GAIN
technologie : DOMAIN
expertise : QUALITY
besoin changement : FEELINGS
nouveau : RESEARCH
projet communautaire aujourd'hui Alice : OCCUPATION
activement communauté technologique local : OCCUPATION
15 : DAY_OF_YEAR
juin : MONTH_OF_YEAR
1985 : YEAR
```

Image 3 : Identification des entités nommées avec les règles heuristiques

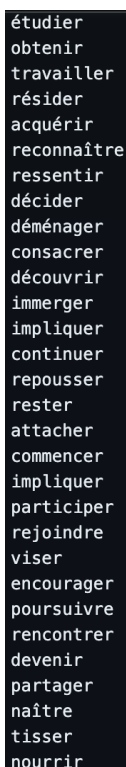
En plus de l'identification des entités, on a ajouté à chacune des entités une explication qui donne une idée générale, c'est l'étape d'annotation. Pour cela, on a utilisé les API Wikidata et GeoNames. GeoNames et Wikidata sont deux ressources riches en informations géographiques et générales sur divers sujets. On va utiliser GeoNames pour les entités de type Localisation et Wikidata pour les autres entités, puis on va stocker le tout dans un fichier JSON qui va contenir une description de chaque entité. Par exemple: pour les entités de type Localisation, on va avoir le nom du pays, la longitude et la latitude de celle-ci; et pour les autres types d'entités, on ajoute une description généralisée.

```
{
  "Alice": "prenom feminin",
  "Paris": {
    "description": "Paris (France)",
    "latitude": "48.85341",
    "longitude": "2.3488"
  },
}
```

Image 4 : Exemple d'annotations des entités dans le fichier JSON

2.2.3. Identification des relations

Concernant les relations, on a supposé que les verbes sont des relations. Avant d'identifier les relations, nous avons appliqué une étape de prétraitement. On a utilisé SpaCy pour pouvoir identifier les verbes du texte et les classer comme des relations. Ces verbes vont nous servir pour la construction des triplets ainsi que la construction du graphe.

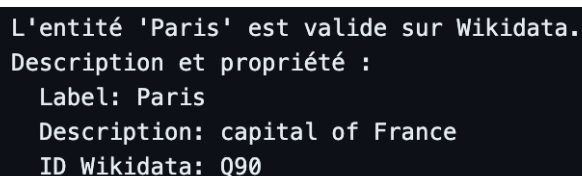


étudier
obtenir
travailler
résider
acquérir
reconnaître
ressentir
décider
déménager
consacrer
découvrir
immerger
impliquer
continuer
repousser
rester
attacher
commencer
impliquer
participer
rejoindre
viser
encourager
poursuivre
rencontrer
devenir
partager
naître
tisser
nourrir

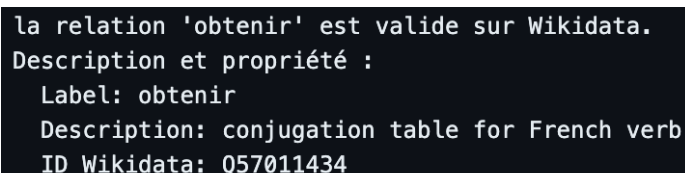
Image 5 : Identification des relations

2.2.4. Enrichissement

Dans la partie enrichissement, nous avons adopté une approche basée sur le principe suivant: si une entité ou une relation existe dans la base de données de Wikidata, alors elle est considérée comme valide. Nous utilisons donc la présence de ces entités et relations dans Wikidata comme critère pour valider et enrichir notre propre graphe de connaissances extrait à partir de textes non structurés. Cette approche nous permet d'assurer la qualité et la fiabilité des entités et des relations incluses dans notre graphe, en les comparant à une source de données largement reconnue et maintenue comme Wikidata.



L'entité 'Paris' est valide sur Wikidata.
Description et propriété :
Label: Paris
Description: capital of France
ID Wikidata: Q90



la relation 'obtenir' est valide sur Wikidata.
Description et propriété :
Label: obtenir
Description: conjugation table for French verb
ID Wikidata: Q57011434

Image 6 : Enrichissement

2.2.5. Construction des triplets

La construction des triplets est une étape fondamentale dans notre démarche d'extraction de graphe de connaissance à partir du texte. Cette étape permet de transformer le texte sous un format tel que triplet = (sujet, verbe, objet) offrant ainsi une structure solide pour former le graphe de connaissance. Dans cette étape, nous avons 3 phases : le prétraitement du texte, l'analyse grammaticale du texte et la construction des triplets par les règles grammaticales. On n'utilisera que la librairie SpaCy pour l'analyse et le traitement textuelle.

Un prétraitement sur le texte est nécessaire afin de pouvoir extraire des triplets corrects. Notamment, nous avons remarqué que lorsqu'un texte contient des pronoms personnels de type "elle", nous obtenons des triplets de type (elle, verbe, objet). Or, ce triplet n'apporte aucune information puisque le pronom n'est pas identifié par son sujet. C'est donc pour cela qu'intervient la gestion des coréférences qui va permettre de remplacer les pronoms personnels par le sujet principal afin d'avoir des triplets corrects (sujet, verbe, objet), et non (pronom, verbe, objet).

Avant la gestion de coréférences:

```
Alice est née à Paris le 10 mai 1990. Elle a étudié l'Informatique à l'Université de Stanford. Après avoir obtenu son diplôme, elle a travaillé pour une
```

Après la gestion de coréférences:

```
Alice est née à Paris le 10 mai 1990 . Alice a étudié l' Informatique à l' Université de Stanford . Après avoir obtenu son diplôme , Alice a travaillé pour une
```

Pour extraire les triplets, nous avons besoin d'analyser le texte grammaticalement à l'aide de SpaCy. Voici un exemple de la structure grammaticale d'un texte:

```
Alice a étudié l ' Informatique à l ' Université de Stanford .  
nsubj aux:tense ROOT det case obj case det det obl:mod case nmod punct  
NOUN AUX VERB DET DET PROPN ADP DET DET NOUN ADP PROPN PUNCT  
étudié étudié étudié Informatique Informatique étudié Université Université Université étudié Stanford Université étudié
```

Cette analyse va permettre de définir des règles pour extraire le sujet, les verbes ainsi que les objets. Nous procéderons à un nettoyage de triplets pour supprimer les doublons, ainsi nous obtenons les triplets suivants:

```
2 : ('Alice', 'reconnaître_1', 'expertise')  
3 : ('Alice', 'immerger_1', 'diversité de ville')  
4 : ('Alice', 'ressentir_1', 'besoin de changement')  
5 : ('Alice', 'décider_1', 'New York')  
6 : ('Alice', 'repousser_1', 'limites dans domaine de technologie')  
7 : ('Alice', 'étudier_2', 'Université de Stanford')  
8 : ('Alice', 'étudier_1', 'Informatique')  
9 : ('Alice', 'naître_2', '10 mai 1990')  
10 : ('Alice', 'obtenir_1', 'diplôme')  
11 : ('amitié', 'tisser_1', 'conversations')  
12 : ('Alice', 'acquérir_2', 'réputation dans domaine de technologie')  
13 : ('Alice', 'naître_1', 'Paris')  
14 : ('Jean', 'naître_3', '15 juin 1985')  
15 : ('Alice', 'attacher_1', 'quête de connaissances')  
16 : ('amitié', 'tisser_2', 'avancées')  
17 : ('Jean', 'partager_1', 'passion pour innovation')  
18 : ('Alice', 'consacrer_1', 'temps')  
19 : ('Alice', 'impliquer_1', 'projets')  
20 : ('Alice', 'participer_1', 'conférences')  
21 : ('Alice', 'résider_1', 'New York')  
22 : ('Alice', 'encourager_1', 'talents')  
23 : ('Alice', 'rejoindre_1', 'initiatives')  
24 : ('Alice', 'impliquer_2', 'projets')  
25 : ('Alice', 'déménager_1', 'côte Est')  
26 : ('Alice', 'acquérir_1', 'années')  
27 : ('Alice', 'poursuivre_2', 'STEM')  
28 : ('Alice', 'travailler_1', 'entreprise de technologie')  
29 : ('Alice', 'poursuivre_1', 'carrière')  
30 : ('amitié', 'nourrir_1', 'esprit de curiosité')  
31 : ('Jean', 'devenir_1', 'ami')
```

2.2.6. Construction du graphe de connaissance

Maintenant que nous avons nos triplets, nous devons construire un graphe de connaissance qui va permettre de comprendre la structure des données et d'identifier les informations pertinentes. Pour construire notre graphe de connaissance, nous allons représenter les triplets de la façon suivante : on forme un nœud pour les sujets et les objets, et nous créons un arc entre ces 2 nœuds qui contiendra le verbe. Voici la représentation d'un triplet (sujet,verbe,objet) :



Image 7 : Schéma d'un triplet en graphe

Voici un exemple de graphe de connaissance :

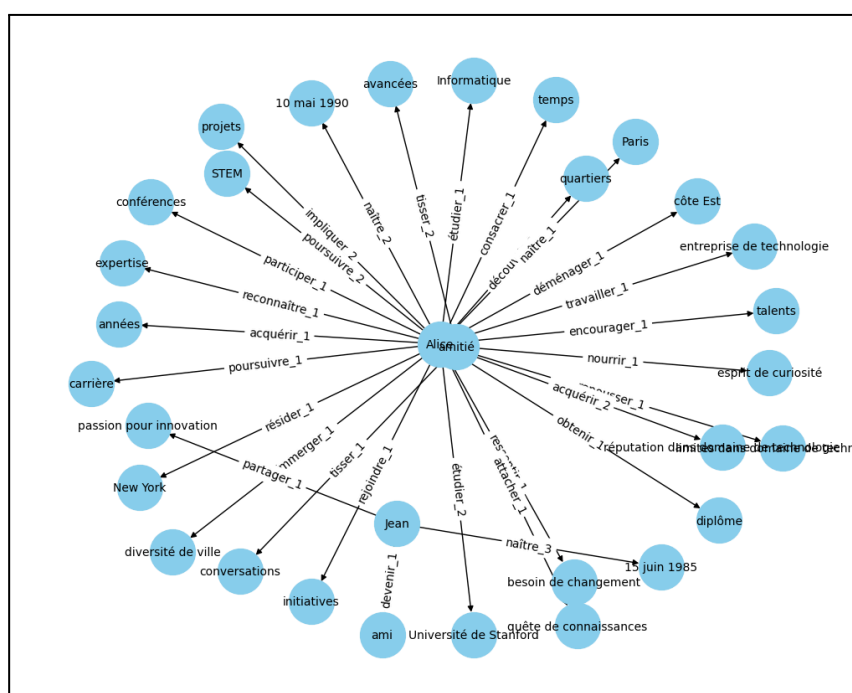


Image 8 : Graphe obtenu à partir des triplets extraits dans l'étape précédente

2.2.7. Conversion du graphe sous format RDF

On aimerait transformer ce graphe de connaissance sous le format RDF, un modèle standardisé pour représenter des données sur le web sémantique. Le passage sous RDF va permettre à d'autres systèmes d'accéder et de comprendre les données qu'on a structuré sous forme de graphe de connaissance tout en explorant à l'aide des requêtes SPARQL.

Nous avons opté pour l'enregistrement du graphe RDF sous format XML :

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
  xmlns:ns1="http://www.semanticweb.org/thivani/ontologies/2024/1/"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
>
  <rdf:Description rdf:about="http://www.semanticweb.org/thivani/ontologies/2024/1/untitled-ontology-4Alice">
    <ns1:untitled-ontology-4découvrir_1>quartiers</ns1:untitled-ontology-4découvrir_1>
    <ns1:untitled-ontology-4reconnaître_1>expertise</ns1:untitled-ontology-4reconnaître_1>
    <ns1:untitled-ontology-4immerger_1>diversité de ville</ns1:untitled-ontology-4immerger_1>
    <ns1:untitled-ontology-4ressentir_1>besoin de changement</ns1:untitled-ontology-4ressentir_1>
    <ns1:untitled-ontology-4décider_1>New York</ns1:untitled-ontology-4décider_1>
    <ns1:untitled-ontology-4repousser_1>limites dans domaine de technologie</ns1:untitled-ontology-4repousser_1>
    <ns1:untitled-ontology-4étudier_2>Université de Stanford</ns1:untitled-ontology-4étudier_2>
    <ns1:untitled-ontology-4étudier_1>Informatique</ns1:untitled-ontology-4étudier_1>
    <ns1:untitled-ontology-4naître_2>10 mai 1990</ns1:untitled-ontology-4naître_2>
    <ns1:untitled-ontology-4obtenir_1>diplôme</ns1:untitled-ontology-4obtenir_1>
    <ns1:untitled-ontology-4acquérir_2>réputation dans domaine de technologie</ns1:untitled-ontology-4acquérir_2>
    <ns1:untitled-ontology-4naître_1>Paris</ns1:untitled-ontology-4naître_1>
    <ns1:untitled-ontology-4attacher_1>quête de connaissances</ns1:untitled-ontology-4attacher_1>
    <ns1:untitled-ontology-4consacrer_1>temps</ns1:untitled-ontology-4consacrer_1>
    <ns1:untitled-ontology-4impliquer_1>projets</ns1:untitled-ontology-4impliquer_1>
    <ns1:untitled-ontology-4participer_1>conférences</ns1:untitled-ontology-4participer_1>
    <ns1:untitled-ontology-4résider_1>New York</ns1:untitled-ontology-4résider_1>
    <ns1:untitled-ontology-4encourager_1>talents</ns1:untitled-ontology-4encourager_1>
    <ns1:untitled-ontology-4rejoindre_1>initiatives</ns1:untitled-ontology-4rejoindre_1>
    <ns1:untitled-ontology-4impliquer_2>projets</ns1:untitled-ontology-4impliquer_2>
    <ns1:untitled-ontology-4déménager_1>côte Est</ns1:untitled-ontology-4déménager_1>
    <ns1:untitled-ontology-4acquérir_1>années</ns1:untitled-ontology-4acquérir_1>
    <ns1:untitled-ontology-4poursuivre_2>STEM</ns1:untitled-ontology-4poursuivre_2>
    <ns1:untitled-ontology-4travailler_1>entreprise de technologie</ns1:untitled-ontology-4travailler_1>
    <ns1:untitled-ontology-4poursuivre_1>carrière</ns1:untitled-ontology-4poursuivre_1>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.semanticweb.org/thivani/ontologies/2024/1/untitled-ontology-4Jean">
    <ns1:untitled-ontology-4naître_3>15 juin 1985</ns1:untitled-ontology-4naître_3>
    <ns1:untitled-ontology-4partager_1>passion pour innovation</ns1:untitled-ontology-4partager_1>
    <ns1:untitled-ontology-4devenir_1>ami</ns1:untitled-ontology-4devenir_1>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.semanticweb.org/thivani/ontologies/2024/1/untitled-ontology-4amitié">
    <ns1:untitled-ontology-4tisser_1>conversations</ns1:untitled-ontology-4tisser_1>
    <ns1:untitled-ontology-4tisser_2>avancées</ns1:untitled-ontology-4tisser_2>
    <ns1:untitled-ontology-4nourrir_1>esprit de curiosité</ns1:untitled-ontology-4nourrir_1>
  </rdf:Description>
</rdf:RDF>
```

Image 9 : Graphe sous format RDF/XML

2.2.8. Stockage du graphe RDF

Dans la phase de stockage du graphe RDF, nous avons choisi d'utiliser la base de données de graphes GraphDB pour stocker notre graphe RDF. Une fois le graphe stocké, nous avons utilisé des requêtes SPARQL pour interroger la base de données.

À chaque requête SPARQL exécutée, nous avons obtenu des résultats pertinents associés aux requêtes, ce qui nous a permis d'extraire des informations spécifiques du graphe RDF et d'effectuer des analyses plus approfondies. Cette approche de stockage et d'interrogation a facilité la manipulation et l'exploitation efficaces des données RDF extraites à partir des textes non structurés. Vous trouverez ci-dessous des exemples de requêtes SPARQL effectuées sur un graphe RDF stockées dans GraphDB.

Requête Sparql n°1 : Résidence d'Alice

```
1 SELECT ?residence
2 WHERE {
3   <http://www.semanticweb.org/thivani/ontologies/2024/1/untitled-ontology-4Alice>
4   <http://www.semanticweb.org/thivani/ontologies/2024/1/untitled-ontology-4résider> ?residence
5 }
```

Résultat :

	residence
1	New York

Requête Sparql n°2 : Etude d'Alice

```
1 SELECT ?studyField ?university
2 WHERE {
3   <http://www.semanticweb.org/thivani/ontologies/2024/1/untitled-ontology-4Alice>
4   <http://www.semanticweb.org/thivani/ontologies/2024/1/untitled-ontology-4étudier> ?studyField .
5   <http://www.semanticweb.org/thivani/ontologies/2024/1/untitled-ontology-4Alice>
6   <http://www.semanticweb.org/thivani/ontologies/2024/1/untitled-ontology-4étudier1> ?university
7 }
```

Résultat :

	studyField	university
1	Informatique	Université de Stanford

Requête Sparql n°3 : les propriétés de chaque ville:

```
1 PREFIX ns1: <http://www.semanticweb.org/thivani/ontologies/2024/1/>
2
3 SELECT ?ville ?propriété ?valeur
4 WHERE {
5   ?ville ns1:untitled-ontology-4description ?description .
6   BIND (str(?ville) AS ?propriété)
7   BIND (STRAFTER(?description, "'description': ") AS ?valeur)
8 }
```

Résultat:

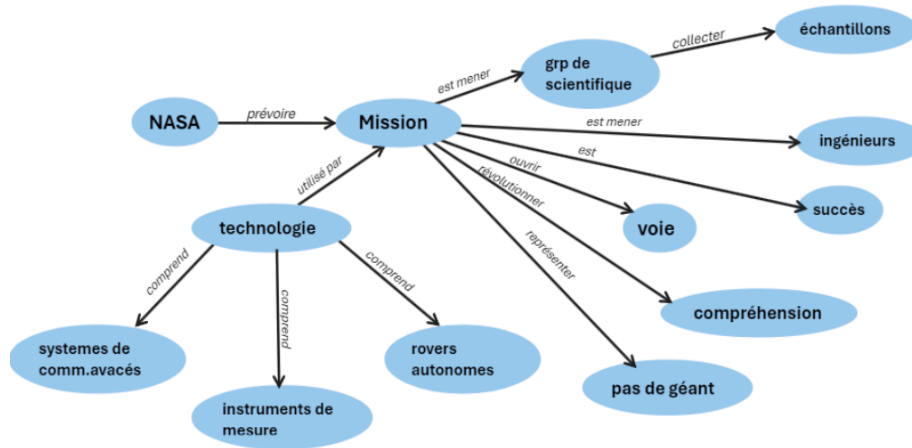
2	http://www.semanticweb.org/thivani/ontologies/2024/1/untitled-ontology-4Paris	http://www.semanticweb.org/thivani/ontologies/2024/1/untitled-ontology-4Paris	Paris (France); 'latitude': '48.85341', 'longitude': '2.3488'
3	http://www.semanticweb.org/thivani/ontologies/2024/1/untitled-ontology-4New York	http://www.semanticweb.org/thivani/ontologies/2024/1/untitled-ontology-4New York	New York (United States); 'latitude': '40.71427', 'longitude': '-74.00597'

3. Tests et évaluation

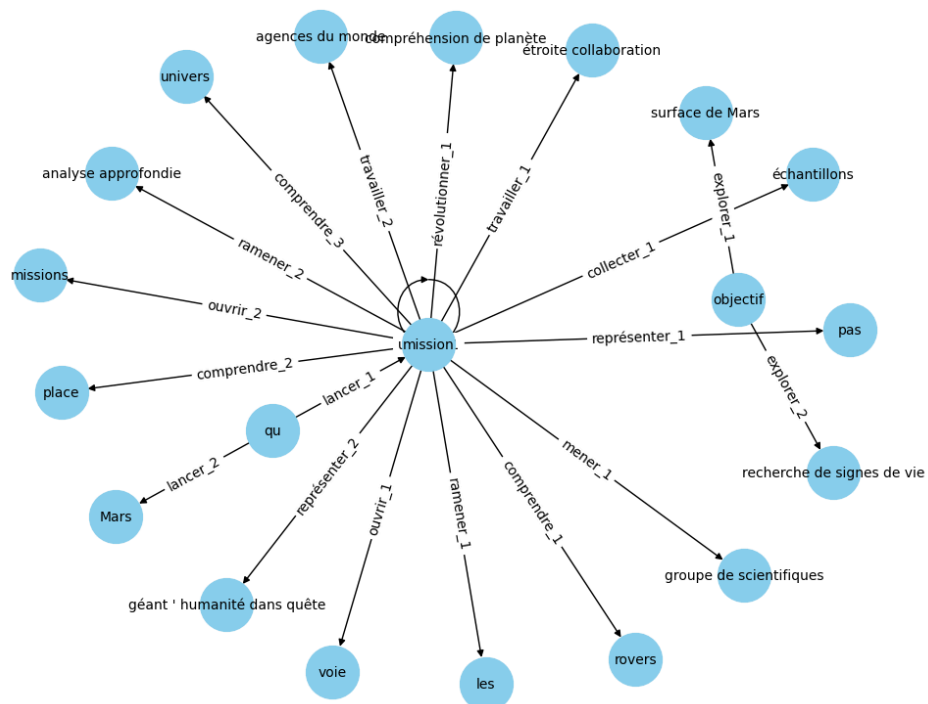
Pour évaluer l'efficacité de notre approche d'extraction de graphe de connaissances à partir du texte, nous avons mené des tests sur différents corpus de texte. Nous avons extrait manuellement des triplets à partir de ces textes, et nous avons dessiné les graphes. Cette démarche nous a permis d'évaluer la précision et la fiabilité de notre méthode d'extraction par rapport aux résultats obtenus par une extraction manuelle, ainsi que d'identifier les éventuelles lacunes ou améliorations nécessaires dans notre approche.

3.1. Test 1

3.1.1. Graphe manuel

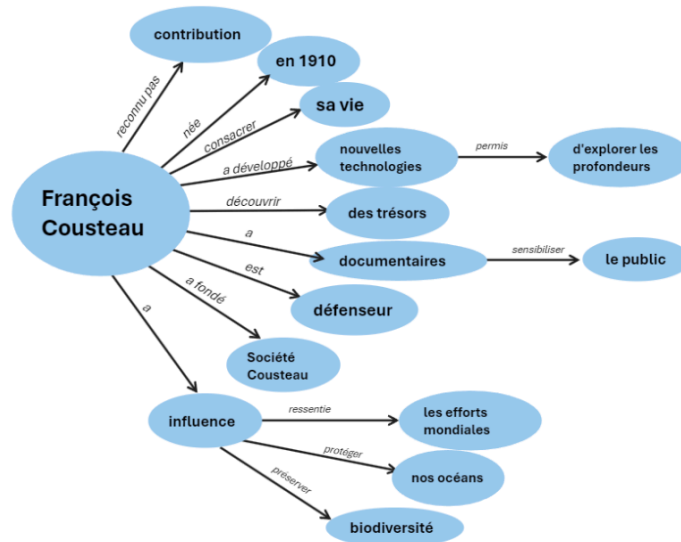


3.1.2. Graphe avec l'approche automatisé



3.2. Test 2

3.2.1. Graphe manuel

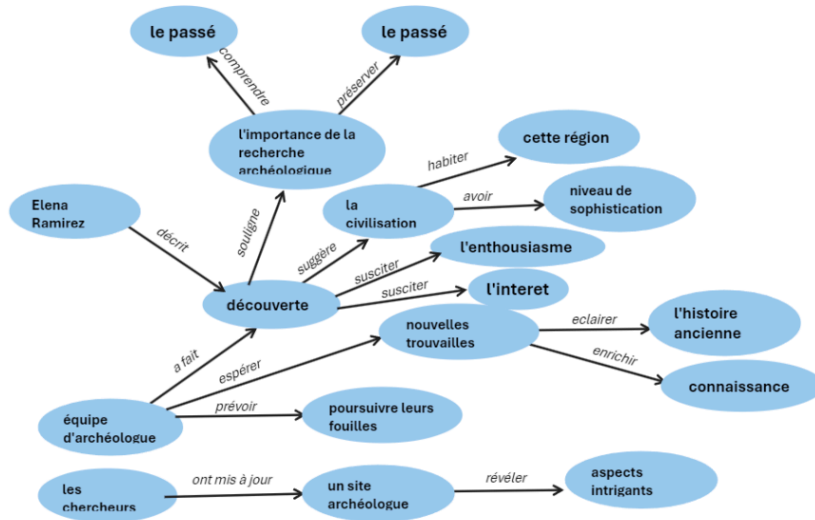


3.2.2. Graphe avec l'approche automatisé

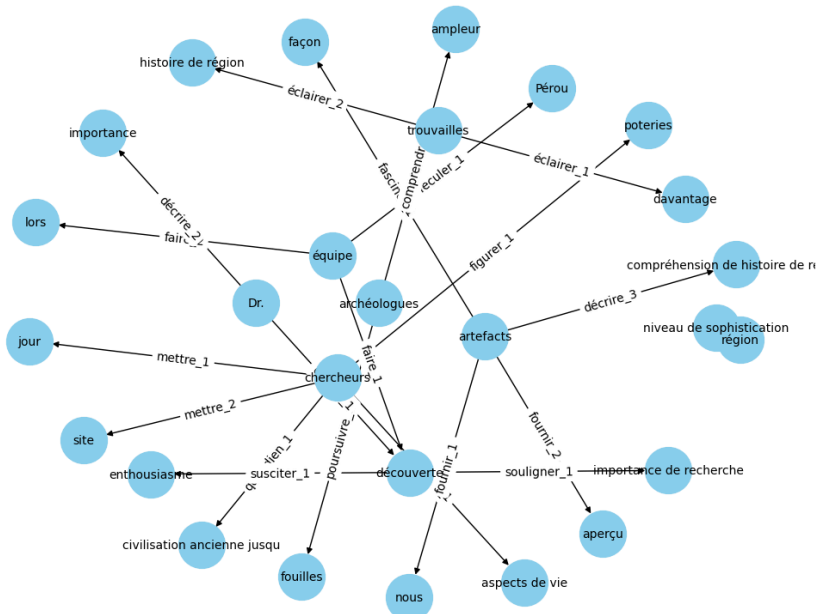


3.3. Test 3

3.3.1. Graphe manuel

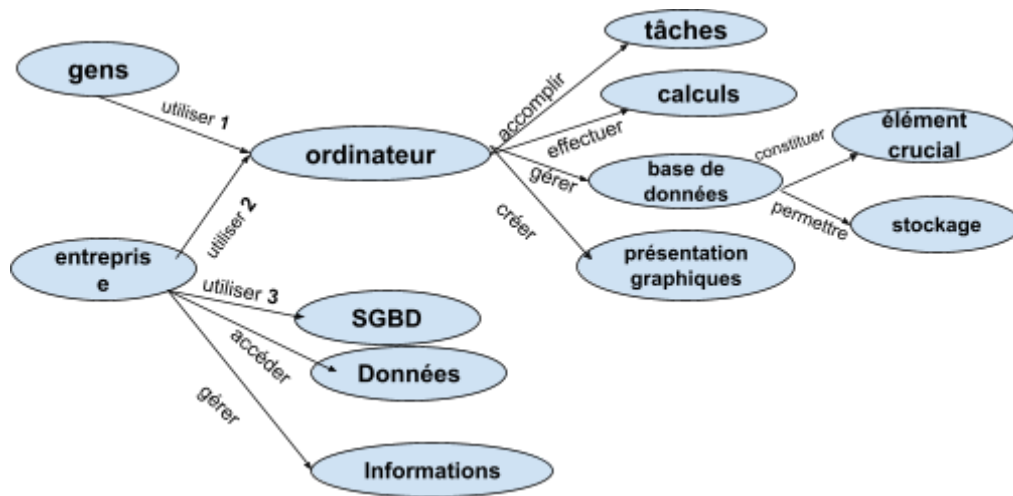


3.3.2. Graphe avec l'approche automatisé

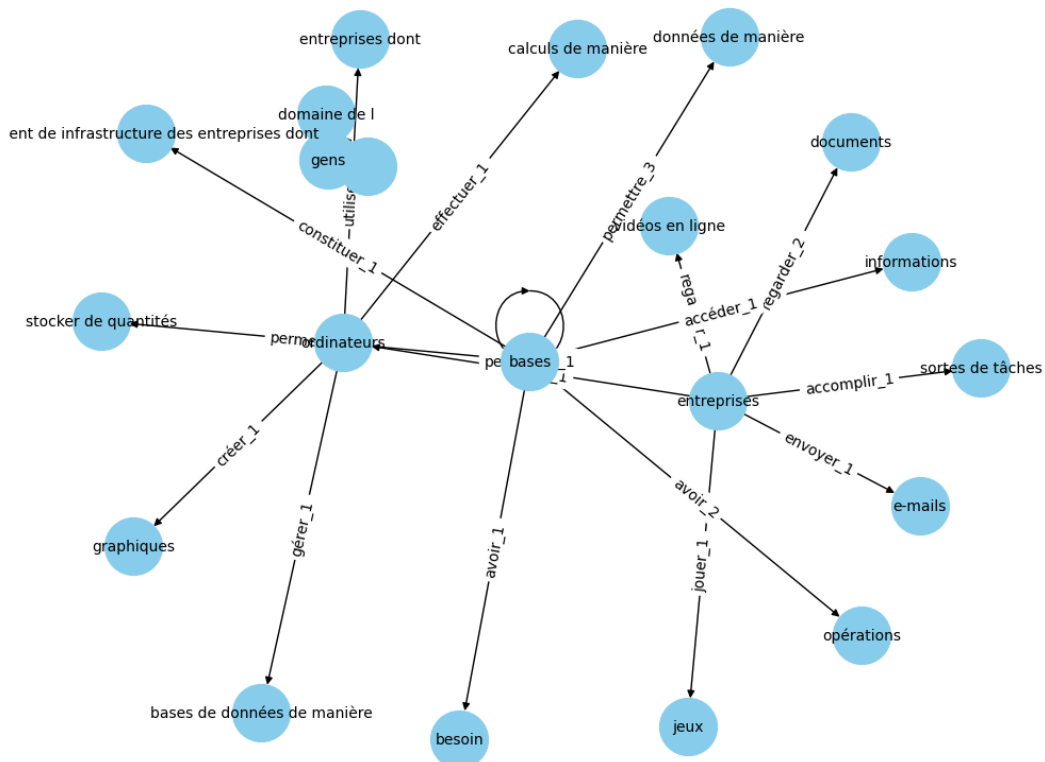


3.4. Test 4

3.4.1. Graphe manuel



3.4.2. Graphe avec l'approche automatisé



4. Etude des articles

4.1. D1-1 : Knowledge Graph Construction from Unstructured Text with Applications to Fact Verification and Beyond

4.1.1. Résumé de l'article

L'article présente une plateforme appelée dstlr qui vise à construire un graphe de connaissances à partir de texte non structuré en extrayant des mentions et des relations pour les intégrer dans une base de données Neo4j. Voici une analyse plus détaillée de l'article.

La plateforme vise à extraire des faits à partir de texte non structuré pour enrichir un graphe de connaissances. Elle utilise des technologies telles que Apache Solr, Stanford CoreNLP, Apache Spark et Neo4j pour effectuer l'extraction, l'enrichissement et le stockage des données. Les entités et relations sont identifiées dans le texte non structuré à l'aide d'outils comme Stanford CoreNLP. Ces faits extraits du texte sont ensuite intégrés dans le graphe de connaissances stocké dans Neo4j. En parallèle, des faits de haute qualité sont extraits de la base de connaissances Wikidata pour enrichir le graphe de connaissances. Ces faits provenant de Wikidata sont également ajoutés à Neo4j pour compléter et améliorer la base de connaissances.

La vérification des faits est réalisée en alignant les relations extraites avec les faits de Wikidata à l'aide de requêtes Cypher dans Neo4j. Cette approche permet de comparer les faits extraits avec une source de connaissances externe réputée pour sa qualité.

La plateforme peut être utilisée pour la vérification des faits, la localisation du support textuel pour les faits affirmés, la détection d'incohérences et de lacunes dans les faits, ainsi que l'extraction de données d'entraînement supervisées à distance.

La plateforme dstlr est présentée comme une solution complète et évolutive pour la construction de graphes de connaissances à partir de texte non structuré. L'article souligne l'importance de l'alignement des sous-graphes pour soutenir diverses tâches liées à la vérification des faits et à l'exploitation des graphes de connaissances.

4.1.2. Comparaison avec notre approche

Notre méthode d'extraction de graphe partage des points communs avec l'approche décrite dans l'article en ce qui concerne l'utilisation de bibliothèques de NLP, l'annotation des entités et la validation avec des sources externes comme Wikidata. Cependant, nous différons dans les détails de nos méthodes d'extraction, de validation et d'enrichissement du graphe.

Dans notre approche, nous privilégions l'utilisation de règles heuristiques pour extraire les entités manquantes et pour annoter les entités à l'aide d'API externes telles que Wikidata et GeoNames. En revanche, l'article se concentre sur l'extraction des entités et des relations à l'aide de CoreNLP.

Par ailleurs, notre méthode semble se concentrer sur l'identification des relations à partir des verbes, ce qui la distingue de l'approche de l'article qui utilise des annotateurs spécifiques tels qu'OpenIE et KBP de CoreNLP pour cette tâche.

En ce qui concerne l'enrichissement du graphe, notre approche accorde une importance particulière à la validation manuelle des entités et des relations. En revanche, l'article décrit une approche plus automatisée pour l'enrichissement du graphe avec des faits provenant de Wikidata.

4.2. D1-3 : Constructing a Knowledge Graph from Indian Legal Domain Corpus

4.2.1. Résumé de l'article

La problématique de cet article est de trouver une approche qui permet de construire un graphe de connaissances à partir des documents non structurés dans le domaine juridique en Inde. L'extraction d'information (IE) dans les documents juridiques indiens pose des défis majeurs pour l'utilisation de modèles supervisés en raison du manque de données étiquetées, de la difficulté à obtenir des ensembles de données appropriés en raison de la numérisation en cours et de la mauvaise qualité des documents, notamment les fautes d'orthographe, la ponctuation incorrecte et la structure non uniforme.

Cet article présente une méthode basée sur des règles pour extraire, annoter et stocker systématiquement les entités clés juridiques dans une base de connaissances. Cette approche comprend l'extraction des entités nommées (NER) et des relations (RE) du corpus juridique, suivie du stockage et de la gestion des connaissances dans une base de données graphe sous forme de triplets. L'utilisation de l'ontologie NyOn (Nyaya Ontology) facilite l'identification des entités et des relations dans les documents juridiques. Ainsi le graphe de connaissances obtenu permettra de répondre à des questions spécifiques dans le domaine juridique.

La méthode comprend:

- la sélection du jeu de données provenant d'IndianKanoon.org,
- le prétraitement de données via le logiciel GATE,
- l'extraction des entités à l'aide de règles en JAPE et NyOn stockées en XML,
- l'extraction des relations entre les entités avec NyOn,
- la construction de triplets pour former un graphe de connaissances,
- et enfin, le chargement et l'interrogation de ces triplets dans un triplestore utilisant SPARQL, avec exposition en tant que Linked Open Data via des URI HTTP dé-référencables.

4.2.2. Comparaison avec notre approche

Cet article nous a permis de guider pour donner les étapes cruciales de notre approche: prétraitement, extraction des entités, extraction des relations, construction des triplets pour avoir le graphe de connaissances et le stockage dans un triplestore pour effectuer des requêtes SPARQL. Cependant, ils utilisent des outils différents des nôtres.

Pour le prétraitement de données, ils ont opté pour le logiciel GATE en Java alors que nous sommes contents de la librairie python SpaCy, qui est très simple à utiliser. La différence est que GATE offre une interface graphique pour gérer les pipelines de traitements de texte ainsi qu'une personnalisation approfondie grâce à son moteur de règles JAPE.

Avec l'utilisation de ce logiciel, ils viennent renforcer avec l'utilisation de NyOn qui est une ontologie qui va servir de schéma pour faire l'extraction des entités et des relations. Elle indique les types d'entités et de relations que l'on doit identifier dans les documents juridiques. Dans notre cas, nous avons utilisé des ontologies comme GeoNames seulement pour annoter nos entités et enrichir notre graphe de connaissance.

L'article nous a donné des choix sur la sérialisation en RDF (RDF/XML, Turtle et N Triples) ainsi que le stockage du graphe de connaissance dans un triplestore (Blazegraph, Apache Jena et Virtuoso). Nous avons pris la sérialisation en XML mais nous avons opté pour un autre triplestore : GraphDB.

4.3. D2-1 : Building Knowledge Graphs from Unstructured Texts : Applications and Impact Analyses in Cybersecurity Education

4.3.1. Résumé de l'article

Le document met en lumière l'importance des graphes de connaissances pour la représentation et la gestion des informations, soulignant le rôle clé de l'ontologie dans leur construction. Il aborde les défis liés à la construction de ces graphes à partir de textes non structurés, proposant une méthodologie combinant l'apprentissage automatique et l'expertise humaine. Cette méthodologie comprend plusieurs phases, notamment la construction des graphes, la collecte de données, l'intégration des connaissances et le développement de l'ontologie.

La phase de construction implique l'extraction des entités et des relations à partir des données brutes, suivie du développement d'un schéma spécifique. La collecte de données utilise des méthodes de reconnaissance d'entités nommées, tandis que l'intégration des connaissances vise à garantir une représentation précise et significative des entités et des relations. Le développement de l'ontologie catégorise les entités et les attributs pour organiser efficacement les informations.

L'étude identifie également des relations spécifiques pour l'éducation en cybersécurité et propose un modèle de validation de schéma pour maintenir l'intégrité des données. Un module d'Entity Matcher est conçu pour éviter les omissions d'entités, tandis que la phase de stockage de connaissances utilise Neo4j comme système de gestion de base de données. Enfin, le document présente une conclusion sur la méthodologie et son application dans le domaine du cloud computing, soulignant la perception positive des étudiants envers les graphes de connaissances comme outils informatifs et visuels.

4.3.2. Comparaison avec notre approche

La problématique centrale de cet article porte sur la construction de graphes de connaissances à partir de textes non structurés, avec un accent particulier sur le domaine de l'éducation en cybersécurité. Les auteurs mettent en lumière les défis majeurs associés à cette tâche, notamment l'absence d'une ontologie bien définie ou d'un ensemble d'entités-relation étiquetés pour ce nouveau domaine. Pour surmonter ces défis, les experts ont entrepris de

classifier minutieusement les entités et les relations identifiées dans les textes en différentes catégories.

Cette classification a permis une meilleure compréhension et organisation des entités et des relations, facilitant ainsi la construction de graphes de connaissances précis et pertinents pour le domaine de la cybersécurité éducative. Dans le cadre de notre méthode, nous avons adopté une approche similaire en utilisant des ressources externes telles que Wikidata et GeoNames pour enrichir davantage les descriptions associées à chaque entité.

En intégrant les informations provenant de ces sources supplémentaires, notre méthode vise à améliorer la richesse sémantique des entités et des relations dans le graphe de connaissances.

Ce processus renforce également la fiabilité et la précision des informations disponibles, ce qui est crucial dans un domaine aussi critique que la cybersécurité.

4.4. D3-2 : Domain Ontology : Automatically extracting and structuring community language from texts

4.4.1. Résumé de l'article

L'article présente une méthode automatique pour construire une hiérarchie de concepts de domaine à partir d'un corpus textuel. Cette hiérarchie de concepts vise à aider à la construction d'ontologies et à l'indexation sémantique des collections de documents. L'approche mise en œuvre est conçue pour être automatique, aussi indépendante que possible de la langue et peut être appliquée à n'importe quel domaine. L'article met en avant les avantages de l'utilisation de la C-Value ainsi que le poids "tf.idf" pour sélectionner les termes candidats dans le processus de construction d'ontologies de domaine.

Le principe est la sélection des termes candidats dans le processus d'extraction de termes. L'approche se base sur la détection des n-grammes, où les termes candidats idéaux doivent servir à discriminer entre les documents pertinents et non pertinents lors de la requête de la collection. De plus, les bons candidats pour une ontologie de domaine doivent également être des termes fréquents, car ces termes fréquents sont généralement représentatifs d'un domaine. Ainsi, l'approche prend en compte ces deux aspects pour sélectionner de bons candidats de termes.

Ce processus aborde l'importance des mots vides (stop words) dans le processus d'extraction de termes candidats. Les mots vides sont des mots courants qui sont souvent exclus lors de l'analyse de texte car ils ne portent pas de sens distinctif. Cependant, dans certains cas, les mots vides peuvent jouer un rôle crucial dans la compréhension du contexte et la sélection des termes pertinents. Cette section met en lumière l'impact des mots vides sur l'extraction de termes candidats et souligne l'importance de les prendre en compte de manière appropriée dans le processus d'analyse de texte.

Le processus utilise aussi la racinisation (stemming) qui est utilisé pour regrouper les termes en une forme commune. Par exemple, les mots "representations" et "representation" seront racinisés en "represent". Dans cette approche, chaque mot individuel est racinisé avant l'extraction des n-grammes à partir des textes. Cela conduit à des n-grammes racinisés et à

leurs termes équivalents. Les n-grammes racinisés sont considérés comme l'étiquette du concept dans l'ontologie, tandis que les variantes de ces n-grammes correspondent aux étiquettes associées.

Les candidats sont sélectionnés en utilisant deux poids complémentaires dans le processus d'extraction de termes. Tout d'abord, les candidats sont extraits de deux manières complémentaires : par comparaison avec un dictionnaire fourni par l'utilisateur et par une analyse statistique de la collection de textes. Extraction par dictionnaire : chaque n-gramme extrait est comparé aux entrées du dictionnaire, et seuls les candidats correspondant à une entrée du dictionnaire sont sélectionnés. Analyse statistique : cette méthode complète l'extraction par dictionnaire en extrayant des collocations qui correspondent à des candidats. Les collocations sont des séquences de mots fréquemment associées dans les textes. Les candidats idéaux sont ceux qui servent à discriminer entre les documents pertinents et non pertinents lors de la requête de la collection, tandis que les bons candidats pour une ontologie de domaine sont des termes fréquents, généralement représentatifs du domaine. Les deux poids utilisés pour sélectionner les candidats sont basés sur ces critères de pertinence et de fréquence.

En conclusion, l'utilisation d'un schéma de pondération spécifique et la structuration des termes peuvent être appliqués indépendamment de la langue, ce qui rend la méthode adaptable à différents contextes linguistiques. L'article souligne l'importance de la construction d'ontologies pour diverses applications telles que la classification de textes, la recherche d'informations et le résumé de textes, en mettant en avant le potentiel de la méthode présentée pour faciliter ces tâches. Il est noté que la construction d'ontologies peut être une tâche chronophage, mais l'aide apportée par des méthodes automatiques peut être précieuse pour les concepteurs. L'exactitude de l'ontologie peut varier en fonction de l'application, mais même des représentations partielles peuvent être utiles pour enrichir les informations.

4.4.2. Comparaison avec notre approche

La méthode décrite dans l'article pour l'extraction d'entités se distingue de notre méthode plus classique basée sur des règles heuristiques par son approche statistique et automatique. Contrairement aux méthodes traditionnelles que l'on utilise qui se basent sur des règles prédéfinies pour extraire des entités, la méthode de l'article utilise des techniques d'analyse statistique et de pondération pour identifier et sélectionner les termes pertinents.

La méthode de l'article repose sur l'extraction de termes candidats à partir de n-grammes et sur la sélection de ces candidats en fonction de leur pertinence et de leur fréquence dans les documents. En utilisant des poids basés sur la fréquence et l'occurrence des termes, la méthode de l'article parvient à identifier des termes discriminants qui sont ensuite organisés hiérarchiquement pour construire une ontologie de domaine.

En revanche, notre méthode se base sur des règles heuristiques prédéfinies pour identifier des entités spécifiques en se reposant sur des modèles linguistiques ou des expressions régulières. Ces règles heuristiques peuvent être efficaces dans certains cas, mais elles peuvent être limitées en termes de couverture et de capacité à s'adapter à différents domaines ou langues.

En comparaison, la méthode de l'article offre une approche plus flexible et adaptable grâce à son caractère statistique et automatique. En utilisant des poids de pertinence et des mécanismes de structuration hiérarchique, cette méthode permet une extraction et une organisation efficace des entités à partir de textes, offrant ainsi une solution plus robuste et généralisable pour la construction d'ontologies de domaine.

5. Conclusion

Ce rapport sur l'extraction de graphes de connaissances à partir de textes non structurés a exploré les fondements théoriques, les méthodologies et les applications pratiques de cette approche fascinante dans le domaine du web sémantique. Nous avons étudié en détail les différentes étapes impliquées dans le processus d'extraction, de modélisation et d'enrichissement des connaissances à partir de textes non structurés, en mettant en évidence les défis, les techniques et les outils associés à chaque étape.

L'essor des données non structurées sur le Web a créé un besoin croissant de méthodes efficaces pour donner un sens à ces données et les exploiter de manière significative. Dans ce contexte, l'extraction de graphes de connaissances à partir de textes non structurés offre une approche prometteuse pour organiser, structurer et exploiter les informations dispersées sur le Web et dans d'autres sources de données.

Nous avons présenté notre approche, qui repose sur l'utilisation de techniques avancées de traitement du langage naturel, d'identification d'entités nommées et de modélisation des relations pour extraire des connaissances à partir de textes. Nous avons également comparé notre approche à d'autres méthodes existantes dans la littérature, mettant en lumière les avantages et les défis de chaque approche.

L'évaluation de notre méthode à travers des tests sur différents corpus de texte a permis de valider son efficacité et sa précision dans l'extraction de graphes de connaissances. Nous avons également examiné des études de cas et des articles connexes pour illustrer l'application pratique de ces techniques dans divers domaines tels que le juridique, l'éducation à la cybersécurité et la construction d'ontologies de domaine.

En conclusion, l'extraction de graphes de connaissances à partir de textes non structurés représente une discipline en plein essor avec un potentiel significatif pour transformer la manière dont nous organisons, naviguons et exploitons les vastes quantités de données disponibles. Ce rapport vise à contribuer à cette dynamique en fournissant un aperçu approfondi des principes, des méthodes et des applications de cette approche innovante dans le domaine du web sémantique. À mesure que la technologie et la recherche progressent, nous anticipons des développements encore plus passionnants et des applications encore plus étendues dans ce domaine prometteur.