SUP'COM

المدرسة العليا للمواصلات بتونس

ÉCOLE SUPÉRIEURE DES COMMUNICATIONS DE TUNIS

Cycle de formation des ingénieurs en Télécommunications

Option :

Management de l'Innovation Technologique

Rapport de Projet de fin d'études

Alignement d'entités dans les graphes RDF

Réalisé par :
Salma GASSAB

Supervisé par :
Mme. Malek BEN YOUSSEF

Encadrants :
Mr. Richard CHBEIR & Mme. Nadia YACOUBI

Travail proposé et réalisé en collaboration avec
LIUPPA

Année universitaire : 2020-2021

Ecole Supérieure des Communications de Tunis
SUP'COM
2083 Cité Technologique des Communications - Elghazala - Ariana - Tunisie
Tél. +216 70 240 900 – site web : www.supcom.mincom.tn

Remerciements

Tous d'abord, je tiens à remercier également mon encadrant **M.Richard Chbeir** pour son aide immense, la qualité de son suivi ainsi que pour tous les conseils et les informations qu'il m'a prodigués avec un degré de patience et de professionnalisme sans égal.

Ma reconnaissance va également à **Mme.Nadia Yacoubi** pour ses précieux conseils, son œil critique m'a été très précieux pour structurer le travail et pour améliorer la qualité des différentes sections.

Je tiens à remercier tout particulièrement mon encadrante **Mme. Malek Ben Youssef**, pour l'aide compétente qu'elle m'a apportée, pour sa patience et son encouragement non seulement dans le cadre de ce projet, mais aussi tout au long de mes études à Sup'Com.

Que les membres de jury trouvent, ici, l'expression de mes sincères remerciements pour l'honneur qu'ils me font en prenant le temps de lire et d'évaluer ce travail.

Je souhaite aussi remercier l'équipe pédagogique et administrative de Supcom pour leurs efforts dans le but de nous offrir une excellente formation.

Pour finir, je tiens à remercier toute personne ayant contribué de prés ou de loin à la réalisation de ce travail.

Résumé

Les données liées cherchent à interconnecter les données structurées sur le web d'une manière compréhensible par la machine dans le but d'avoir un seul espace de données qui englobe tous.

Avec la croissance des sources de données disponibles sur le web, le problème d'hétérogénéité de données dans ces ressources présente un défi, alors le besoin d'accès à toutes ces ressources a été le challenge de nombreuses recherches dans le domaine du web sémantique. Donc, c'est nécessaire d'offrir une vue unifiée aux ensembles des données liées par la génération des nouveaux sources des données tout en combinant les informations distinctes et hétérogènes.

Dans la littérature, plusieurs approches proposent différentes solutions qui visent à aligner les entités similaires dans les graphes RDF sans tenir compte des différents types d'hétérogénéités. Ces méthodes sont insatisfaisantes à cause de leur complexité quadratique. Donc, le développement d'une méthode performante qui minimise le nombre des comparaisons et résout les hétérogénéités des données est une exigence.

Alors, afin d'aligner les entités similaires, nous avons proposé une méthode qui vise à réduire le nombre de comparaisons lors de l'alignement d'entités et à résoudre le problème d'hétérogénéité de données. Notre approche utilise différentes mesures de similarité pour calculer la similarité entre les entités et elle a été évaluée en utilisant des ensembles de données réelles à partir des bases de connaissances DBpedia et Wikidata.

Mots clés : Données liées, alignement d'entités, Web sémantique, graphes de connaissances, mesures de similarité.

Abstract

Linked data seeks to interconnect structured data on the web in a machine-understandable way in order to have a single data space that encompasses all.

With the growth of data sources available on the web, the problem of data heterogeneity in these resources presents a challenge, so the need to access all these resources has been the challenge of many researches in the field of semantic web. Therefore, it is necessary to provide a unified view of the related data sets by generating new data sources while combining the separate and heterogeneous information.

In the literature, several approaches propose different solutions that aim at aligning similar entities in RDF graphs without considering different types of heterogeneities. These methods are unsatisfactory due to their quadratic complexity. Therefore, developing a powerful method that minimizes the number of comparisons and resolves the heterogeneities in the data is a requirement.

So, in order to align similar entities, we proposed a method that aims to reduce the number of comparisons when aligning entities and solve the data heterogeneity problem. Our approach uses different similarity measures to compute the similarity between entities and has been evaluated using real datasets from DBpedia and Wikidata knowledge bases.

Keywords : Linked Data, entity matching, Semantic web, knowledge graphs, similarity measures.

Table des matières

Table des figures

Liste des tableaux

Liste des algorithmes

Liste des acronymes

RDF	*Ressource Description Framework*
RDFS	*Ressource Description Framework Schema*
LOD	*Linked Open Data*
SPARQL	*Simple Protocol And RDF Query Language*
URI	*Uniform Ressource Identifier*
HTTP	*Hypertext Transfer Protocol*
DAWG	*RDF Data Access Working Group*
W3C	*Consortium World Wide Web*
API	*Application Programming Interface*
TF	*Term Frequency*
IDF	*Inverse Document Frequency*
HTTP	*Hypertext Transfer Protocol*
JSON	*JavaScript Object Notation*
XML	*Extensible Markup Language*
NLP	*Natural Language Processing*
PHP	*Hypertext Preprocessor*
BGP	*Basic Graph Pattern*
TP	*Triple Pattern*

Introduction générale

Le terme du terme ”Linked Data” (données liées) désigne un ensemble de bonnes pratiques pour la publication et l'interconnexion des données structurées sur le web. Dans ce contexte, le web de données permet de publier des données structurées et non structurées sur le web, non pas sous la forme de silos de données isolés les uns des autres, mais en les reliant pour constituer un réseau d'informations global.

Les données liées comprennent plusieurs bases de connaissances qui sont exprimées par diverses ontologies telles que la géographie et la biologie. Ces bases de connaissances dans le web sont souvent créees indépendamment les unes des autres. Ils peuvent contenir des ressources qui présentent les mêmes entités mais qui ne sont pas explicitement définies. Ces bases de connaissances sont reliées entre elles par des liens entre les ressources, identifiées par les URIs. Ils décrivent les relations entre ces ressources, permettant à des agents de naviguer entre les bases de connaissances comme s'ils exploitaient une base de données locale intégrée. En conséquence, une information plus riche et plus enrichie est fournie dans la réponse.

La majorité de ces liens sont des liens d'identité. Par convention, un lien d'identité est défini entre deux ressources par la propriété owl :sameAs. Il exprime que les deux ressources se réfèrent au même objet du monde réel. Alors, lier les entités avec les liens owl :sameAs permet aux bases de connaissances de s'auto-compléter. Ce problème est appelé ”alignement d'entités”.

Avec la croissance des sources de données disponibles dans le web, le problème d'hétérogénéité de données dans ces sources augmente et donc le besoin d'accès à toutes ces sources à travers une interface cohérente a été le défi de nombreuses recherches dans le domaine d'intégration des données liées. En général, plusieurs approches d'alignement d'entités génèrent des candidats potentiels sur la base de certains critères. Cela permet de réduire l'espace de recherche en éliminant les comparaisons inutiles entre les entités qui sont peu susceptibles d'être des entités similaires. Ainsi, le nombre de comparaisons par paires sera minimisé.

Dans ce rapport, nous présentons plusieurs approches d'alignement d'entités. Notre principale contribution est de catégoriser ces approches en utilisant plusieurs critères. Le premier critère étudie les différentes mesures de similarité ainsi que les sources de connaissances qui sont utilisées dans le processus de découverte des liens d'identités. Le deuxième critère permet de comparer le positionnement de ces approches par rapport aux types des hétérogénéités.

Ce rapport est organisé comme suit, le premier chapitre contient un aperçu du contexte général du projet. Il explique l'idée du projet, la problématique abordée et les objectifs. Il introduit aussi les concepts de bases de Linked Data et présente les différentes technologies sous-jacentes.

Le deuxième chapitre détaille le problème d'alignement d'entités, y compris son origine et l'état de l'art. Nous identifions les défis existants pour la tâche d'alignement d'entités. De plus nous comparons les approches de l'alignement d'entités en utilisant des tables avec des aspects importants pour discuter leurs limitations.

Le troisième chapitre est dédié à la présentation de notre approche d'alignement d'entités, nous proposons une architecture permettant le liage de données entre deux bases de connaissances présentant différentes types d'hétérogénéités des données entre eux. Une évaluation de notre approche et une présentation des résultats sont données dans le dernier chapitre.

Nous clôturons le rapport par une conclusion générale et des futures perspectives.

Chapitre 1

Contexte général du projet

1.1 Introduction

Dans ce chapitre, nous allons présenter un aperçu du contexte général du projet. Nous allons présenter l'organisme d'accueil. Ensuite, nous allons expliquer l'idée du projet, la problématique abordée et les objectifs. Enfin, nous allons présenter l'évolution du web, décrire les principes de base des données liées et présenter les différentes technologies sous-jacentes.

1.2 Etablissement d'accueil

Ce projet fait partie d'un projet de recherche mené par le Laboratoire d'Informatique de l'Université de Pau et des Pays de l'Adour ou LIUPPA, créé le 1er octobre 2000, a reçu le label d'EA (n° 3000) le 1er janvier 2001.

Il adresse au sens large les sciences du Numérique avec une activité de recherche appliquée ancrée dans une société de plus en plus numérique et qui s'articule autour : de la gestion des données (dont la masse ne cesse de croître) hétérogènes, des traitements distribués, multimédias, et fortement délocalisées (cloud), et des usages différents (machines, humains). La recherche menée est de nature appliquée avec des domaines de prédilection tels que [17] :

- Génie logiciel
- Agents et composants logiciels
- Sécurité informatique
- Systèmes d'information
- Réseaux et protocoles
- Traitement des documents électroniques

Le LIUPPA assume une vision principalement appliquée de sa recherche qu'il mène depuis sa création et positionne son projet scientifique dans un champ applicatif précis : La gestion des systèmes d'information et des architectures des Systèmes Cyber-Physiques. En effet, ces systèmes sont des systèmes qui relient le monde physique (par exemple, via des capteurs ou des actionneurs) au monde numérique du traitement de l'information. Ils sont composés de divers éléments qui collaborent pour créer un comportement global.

1.3 Préliminaires sur les données liées

Dans cette section, nous allons présenter les différentes évolution du web, puis nous allons décrire les modèles de méta-données RDF, RDFS et les ontologies. Nous allons après présenter une description détaillée du problème d'alignement d'entités.

1.3.1 Evolution du web

Tel qu'illustré par la figure 1.1, le web a connu plusieurs évolutions en 30 ans, ces évolutions correspondent à quatre phases essentielles.

Le Web 1.0, appelé aussi le web de documents est un web statique basé sur un ensemble de normes : un mécanisme d'identification unique de ressource, un mécanisme d'accès universel (le protocole HTTP) et HTML comme format de contenu. Le langage HTML permet de de décrire des pages web riches en textes et créer des liens hypertextes entre les documents web pouvant exister sur différents serveurs web. Mais, il pose des problèmes aux développeurs à cause de sa tolérance aux erreurs. Il permet de transférer des informations d'une manière unidirectionnelle (des développeurs vers les internautes).

Le Web 2.0, appelé aussi le web social, permet le partage et l'échange de contenus (textes, images, vidéos, etc). Il offre aux utilisateurs qui ne connaissent aucun langage de programmation, de gérer les sites en insérant des modules dans les pages web (heures et dates, e-mailing-list, etc). Il a été utilisé pour le travail collaboratif entre les utilisateurs qui peuvent modifier ou créer un contenu comme l'encyclopédie Wikipédia.

Le Web 3.0, appelé aussi le web sémantique, son objectif est la compréhension machine des données, grâce à la mise en place des liaisons sémantiques entre les données qui se trouvent à l'intérieur des documents, dans le but de créer des sites plus intelligents, connectés et ouverts. Le web sémantique permet d'organiser les données en fonction du contexte et des besoins des utilisateurs et de donner un sens aux données. En outre, il contient des données cachées (métadonnées) consacrées à être utilisées par des moteurs de recherche et des applications. Le format RDF est le standard W3C pour la représentation des méta-données. Il permet de représenter ces informations et de structurer les données sous forme de triples <sujet, prédicat, objet>.

Le Web 4.0, appelé aussi le Web intelligent, vise à noyer l'utilisateur dans un espace web intelligent grâce à des agents intelligents.

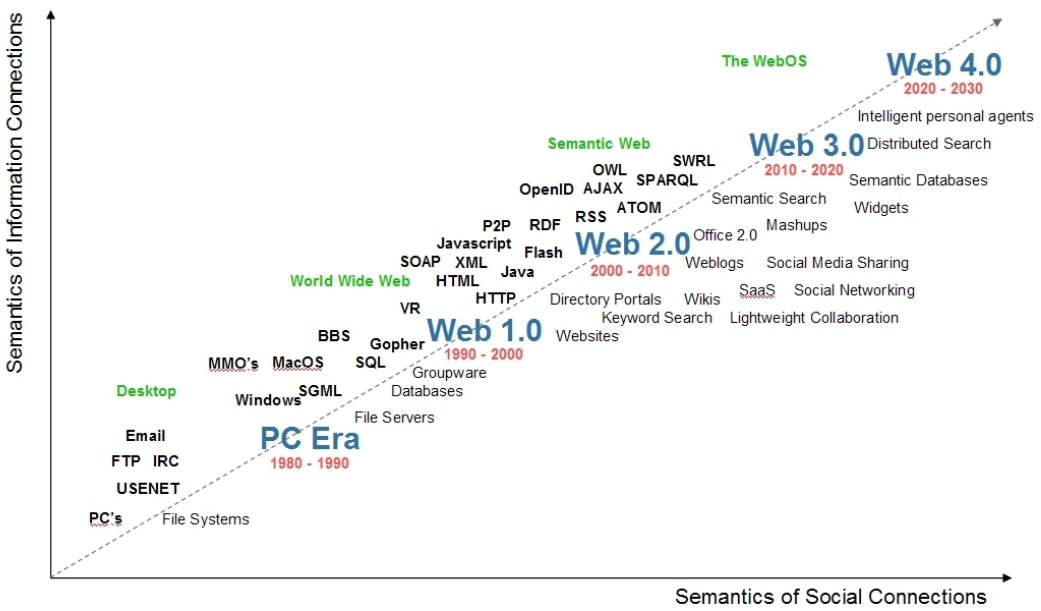

FIG. 1.1 : Les quatre phases d'évolution du web [15]

1.3.2 Ressources, URIs et espaces de noms

Une ressource est le composant de base du web et elle peut être : une page web, une vidéo, un fragment d'un document, un identificateur pour une entité, etc. Une URI est une chaîne de caractères qui identifie une ressource abstraite ou physique et pour exprimer tous les hyperliens du web sous forme d'URIs. Par exemple, la ressource "Richard Chbeir" sur le web est exprimée par l'URI : http ://spider.sigappfr.org/members/rchbeir/.

1.3.3 Ressource Description Framework (RDF)

RDF [11] est un modèle de méta-données basé sur le formalisme de graphes, visant à représenter les informations sur les ressources Web sous forme de graphes orientés avec des nœuds et des arcs étiquetés, ce modèle est destiné à faciliter l'intégration des données hétérogènes et permettre l'échange des données sur le Web de données. Une ressource RDF est représentée sous forme d'un ensemble de triplets RDF qui présente une phrase simple se composant de trois parties : un sujet, un prédicat et un objet. Par exemple :

Le Musée de Louvre est localisé à Paris

Sujet Prédicat Objet

Le sujet d'un triplet RDF est une ressource identifiée par une URI. L'objet d'un triplet RDF peut être soit un littéral (rdf :literal) qui présente une valeur littérale telle qu'une chaîne de caractères, un nombre ou une date ou l'URI d'une autre ressource qui est en relation avec le sujet. Le prédicat est le type de la relation entre le sujet et l'objet et il est identifié également par une URI.

Soit ns1 le préfixe définie par l'URI <http ://mydartaset.chapter1.example/>. Sachant qu'un préfixe est un espace de nom dans lequel un ensemble de ressources RDF sont

déclarées.

À titre d'exemple, un triplet RDF qui correspond à la phrase "Le musée du Louvre est localisé à Paris" utilisant un littéral comme objet pourrait être illustré par le graphe de la Figure 1.2. La même phrase peut être représentée en graphe RDF différement en déclarant "Paris" comme une ressource et donc en lui attribuant un URI tel qu'illustré par la figure 1.3.

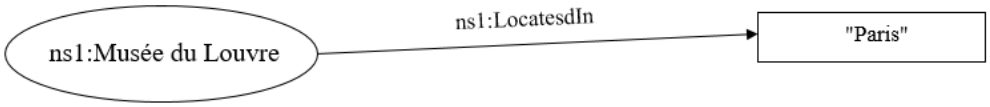

Fig. 1.2 : Exemple de triplet RDF ou l'objet est un littéral

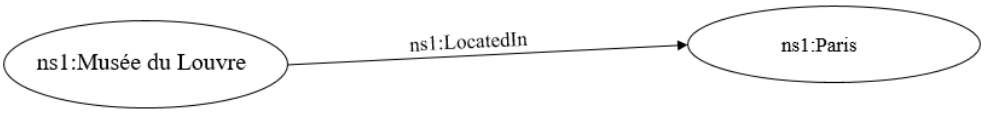

Fig. 1.3 : Exemple de triplet RDF ou l'objet est une URI

En effet, il existe deux visualisations possibles de triplets RDF, soit d'une façon tabulaire (sous forme d'une table de triplet) ou comme un graphe RDF.

Sujet	**Prédicat**	**Objet**
ns1:Musée du Louvre	ns1:LocatedIn	Paris
ns1:Musée du Louvre	ns1:country	ns1:France
ns1:Musée du Louvre	ns1:DateOuverture	10/08/1793
ns1:Musée du Louvre	ns1:région	ns1:Île-de-France

Tab. 1.1 : Visualisation des triplets d'une façon tabulaire

Tel qu'illustré par la figure 1.4, le Musée de Louvre présente une ressource et représentée par une URI dans la base de connaissances DBpedia.

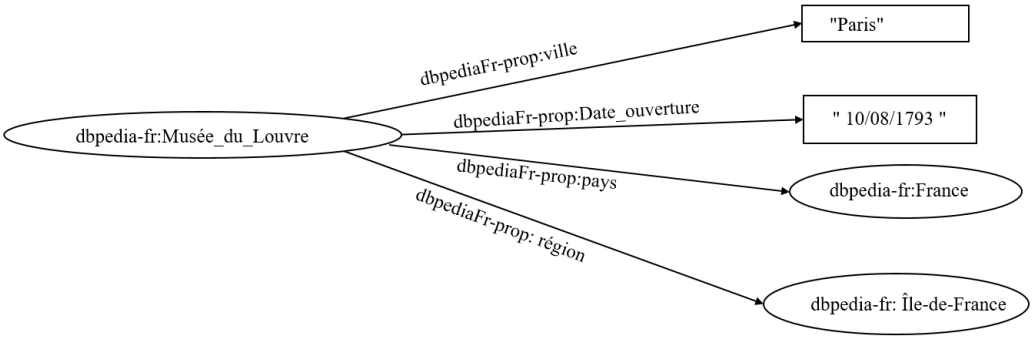

FIG. 1.4 : Représentation sous forme de graphe d'un ensemble de triplets

En RDF, nous pouvons différencier entre les individus (objets et sujets des triplets) et les propriétés (relation) en utilisant deux mots-clés pour les déclarer : rdf :type et rdf :Property. Le premier mot-clé permet de typer les individus avec des classes (qui sont définies avec RDF Schema) alors que le second mot clé rdf :Property permet de déclarer les propriétés comme par exemple :

ns1:localedIn rdf :type rdf :Property .

On considère dans un graphe RDF des nœuds vides qui sont utilisés pour capturer une certaine forme d'objets inconnus. Un nœud blanc (ou une ressource anonyme) est un sujet ou un objet qui n'est ni un littéral ni une URI.

Pour publier un graphe RDF sur le web, il faut qu'il soit sérialisé en utilisant une syntaxe RDF. La sérialisation est le processus d'encodage de l'information sous forme de bits ou d'octets pour assurer le stockage sur un disque et le transfert sur le réseau.

Parmi les syntaxes les plus utilisées pour publier les données RDF sur le web est la syntaxe RDF/XML [3] . Néanmoins, cette sérialisation est destinée à être lu par une machine mais très difficile à écrire ou lire pour les êtres humains puisqu'elle est appuyée sur le langage XML. C'est pourquoi, il est recommandé d'utiliser d'autres sérialisations telles que turtle ou N3.

Une autre manière de sérialisation des données RDF est le format Turtle [6] qui est un format de texte brut, elle est facile à lire, écrire est interpréter par un humain. La figure 1.5 illuste un exemple de sérialisation Turtle de trois triplets RDF.

```
@prefix ns0: <http://mydartaset.chapter1.example/> .

<https://dbpedia.org/page/Louvre>
  ns0:type ns0:Museum ;
  ns0:country ns0:France ;
  ns0:LocatedIn "Paris" ;
  ns0:DateOuverture "10/08/1793" ;
  ns0:région ns0:Île-de-France .
```

Fig. 1.5 : Un exemple de la sérialisation Turtle de trois triplets RDF.

N-Triple est un sous-ensemble de la syntaxe Turtle, mais avec moins de fonctionnalités telles que les préfixes des espaces de noms. Parce que tous les préfixes doivent être spécifiés en plein dans chaque triplet, les fichiers RDF en cette sérialisation ont beaucoup de redondance.

RDF/JSON est une autre manière de sérialisation des fichiers RDF, elle est hautement souhaitable, étant donnée le nombre croissant de langages de programmation fournissant des bibliothèques pour manipuler des données JSON tels que les langages de programmation Web JavaScript et PHP.

1.3.4 RDF Schema (RDFS)

RDF Schema [5] est un langage visant à représenter les ontologies er les vocabulaires sur le web de données liées. Les déclarations RDFS sont exprimées comme des triplets RDF en utilisant le mot clé RDFS. Ainsi, il permet de déclarer les objets et les sujets comme des membres de certaines classes.

D'autre part, RDFS déclare des relations d'héritage entre les classes (SubClasse Of) et les propriétés (SubProperty Of). C'est à dire, qu'il permet de déclarer qu'une classe est une sous-classe d'une autre classe en utilisant le prédicat rdfs :subClassOf. Les relations entre les classes peuvent être déclarées en tant qu'un ensemble de triplets de la forme : <classeA rdfs :subClassOf ClasseB>.

Tel qu'illustré par le script de la figure 1.6, la classe "Museum" est une sous classe de la classe "Building", Alors, si une entité a est de type A (classe A) et classe A et une sous classe de la classe B, alors x est de type B.

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

<http://dbpedia.org/ontology/Museum> rdfs:subClassOf <http://dbpedia.org/ontology/Building> .
```

FIG. 1.6 : Pseudo-code d'une relation d'héritage entre les classes

De même RDFS permet de déclarer les relations hiérarchiques entre les propriétés, en utilisant le prédicat rdfs :subPropertyOf. Par l'exmple dans le script de la figure 1.7 nous pouvons dire que la relation "location" est plus spécifique que la relation "hasLocation".

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

<http://dbpedia.org/ontology/location> rdfs:subPropertyOf <http://www.ontologydesignpatterns.org/ont/dul/DUL.owl#hasLocation> .
```

FIG. 1.7 : Pseudo-code d'une relation structurelle entre les propriétés

1.3.5 Ontologies et vocabulaires

Le terme ontologie [26] est utilisé dans le Web sémantique pour représenter un vocabulaire auquel on aura rajouté un certain nombre d'axiomes sur les classes et les propriétés. Une ontologie est exprimée en OWL qui est une adaptation de la logique de description pour le Web. Ainsi, OWL [12] est un langage d'ontologie plus riche que RDFS et largement utilisé pour ajouter une sémantique aux données.

On appelle axiome une construction logique qui relie souvent une propriété à une classe pour les définir, par exemple tous les livres ont une date d'ouverture, une localisation.

Un fait important est de bien savoir nommer les éléments d'un vocabulaire pour mieux décrire les données, par exemple si l'on veut décrire un musée, l'utilisateur cherchera les termes « œuvre », « localisation », « nombre de visiteurs », etc. d'où l'importance de donner de bons labels aux types et propriétés afin que le lecteur puisse déjà trouver le vocabulaire dont il a besoin et seulement ensuite, il vérifiera si le vocabulaire correspond bien à ses données.

On utilise donc le terme ontologie si la construction est faite en structurant les classes et les propriétés avec une sémantique formelle déclarée et dont la principale fonctionnalité est d'en faire du raisonnement logique.

1.3.6 Projet Linked Open Data

Un nombre important de personnes et d'organisations ont adopté les principes des données liées comme un moyen pour publier leurs données sur le web. Le Linked Open Data Cloud (nommé LOD Cloud) [30] est un projet qui permet de publier une quantité énorme de données structurées sur le web de données, selon les principes des données liées. Actuellement, il est constitué de billions de triplets RDF organisés en jeux de données

générés automatiquement à partir de nombreuses sources couvrant toutes sortes de sujets, à savoir : des emplacements géographiques, des personnes, des entreprises, des livres, des publications scientifiques, des films, de la musique, des programmes de télévision et de radio, des gênes, des protéines, des médicaments et des essais cliniques, des données statistiques, des communautés en ligne,etc.

Ce projet vise à établir des connexions entre ces jeux de données construits séparement afin d'obtenir un web de données global. En effet, les sources de données qui existent dans le LOD Cloud sont classées dans les domaines thématiques suivants : géographique, gouvernement, médias, bibliothèques, sciences de la vie, commerce, le contenu généré par l'utilisateur et des ensembles de données inter-domaines.

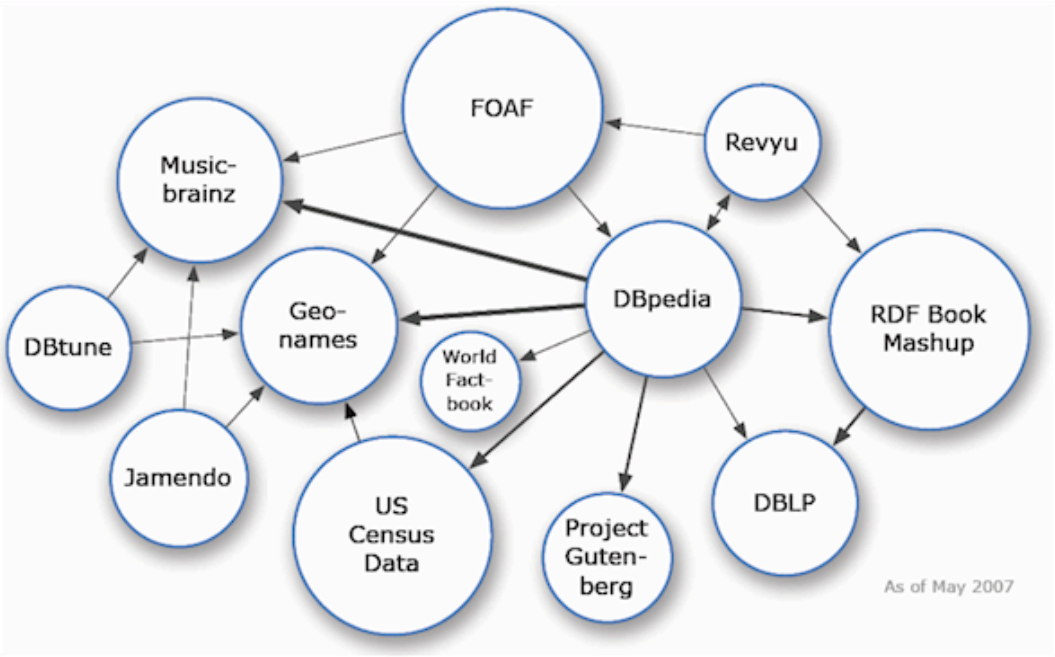

Fig. 1.8 : L'état du LOD Cloud en 2007 [31].

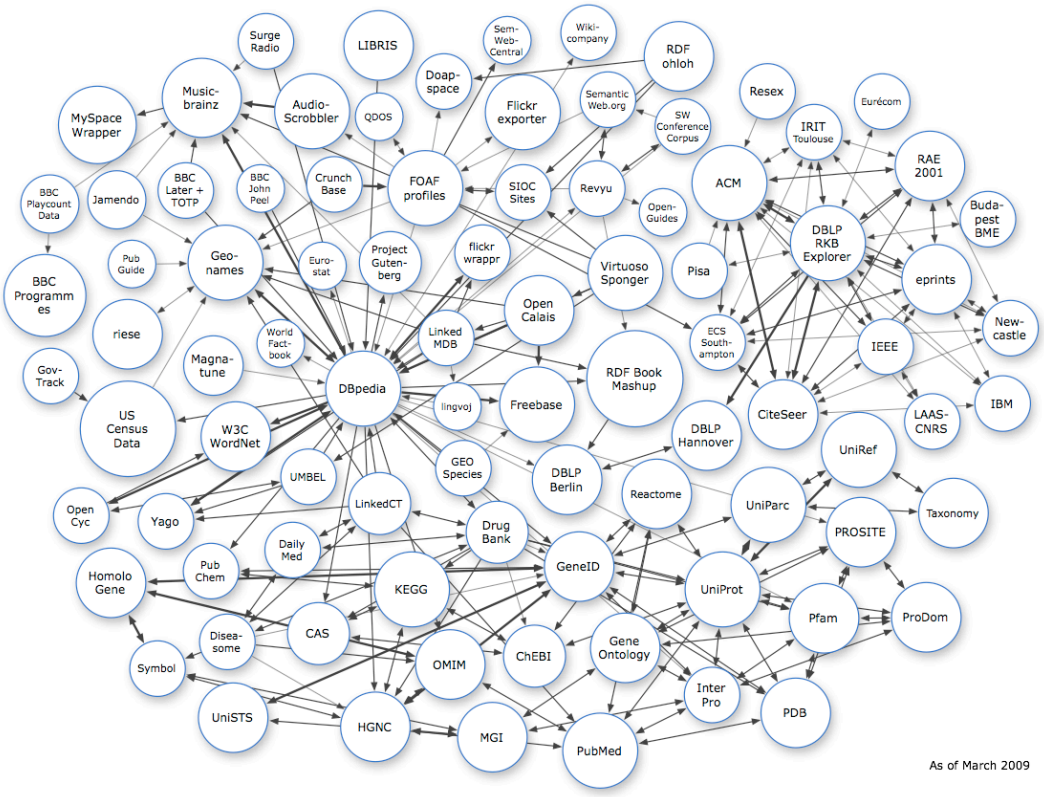

FIG. 1.9 : L'état du LOD Cloud en 2009 [32].

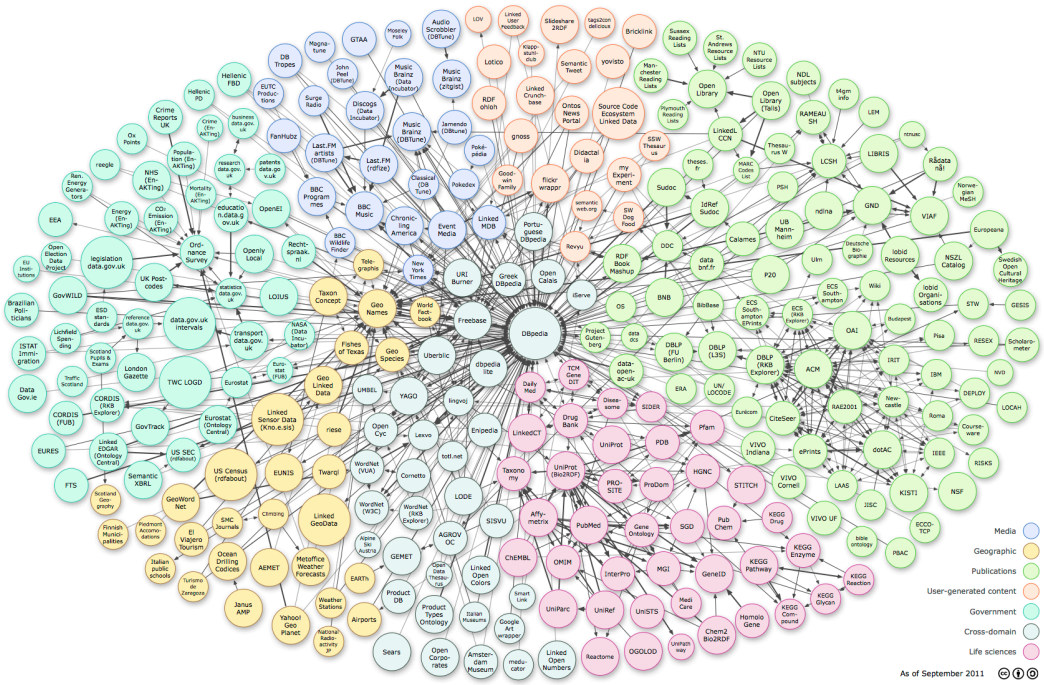

FIG. 1.10 : L'état du LOD Cloud en 2011 [33].

Par exemple, tous les jeux de données qui contiennent des médias peuvent être regroupées en un cluster propre à ce thème, tous les jeux de données qui contiennent des informations géographiques peuvent être regroupés dans un cluster propre à ce thème,

etc. Dans la figure suivante, nous remarquons que les sources qui appartiennent au même cluster ont la même couleur du fond.

Le projet LOD Cloud a connu une croissance exponentielle du nombre de sources de données inter-connectées. Les figures 1.9, 1.10, 1.11 illustrent les sources de données dans le LOD Cloud en 2007, 2009 et 2011, respectivement. En 2007, le LOD Cloud conprenait 12 jeux de données, plus d'un milliard de triplets RDF et environ de 120.000 liens RDF entre les sources [14]. Ensuite, il a évolué à 295 ensembles de données, plus de 31 billions de triplets RDF et environ de 504 millions liens RDF entre les sources de données en 2011. Le nombre de jeux de données dans le LOD Cloud est devenu 9960 en 2016, plus de 176 milliards de triplets RDF et environ de 41 milliards liens RDF entre les sources de données.

1.3.7 Alignement d'entités dans les graphes RDF

Le problème d'alignement d'entités est apparu dans la littérature sous plusieurs terminologies. Ce problème est cité sous différents termes déduplication, matching d'entités, résolution d'entités . L'alignement est la tâche d'identifier les entités qui correspondent au même objet du monde réel. Elle permet ainsi de générer des liens d'identités au niveau des instances pour connecter les entités qui se réfèrent au même objet du monde réel.

La tâche d'alignement d'entités se présente sous plusieurs formes :

Le matching d'entités s'agit d'aligner des entités appartenant à différents ensemble de données. À titre d'exemple, considérons la tâche de fusionner les descriptions de l'entité Musée de Louvre dans deux bases de connaissances différentes Dbpedia et Wikidata, ayant ainsi des structures très hétérogènes et différentes.

La Déduplication s'agit d'aligner les entités qui correspondent au même objet du monde réel dans un même ensemble de données.

Dans le web des données liées, plusieurs fournisseurs de données utilisent leurs propres URIs pour désigner les mêmes entités. Ceci résulte que pour une même entité du monde réel plusieurs URIs existent à travers les sources du LOD. Ces URIs sont appelés "alias URIs" [14]. En effet, les éditeurs des données peuvent utiliser le lien " http ://www.w3.org/2002/07/owl#sameAs " pour déclarer que deux URIs distincts présentent la même entité.

Le script de la figure 1.11 présente deux URIs qui se réfèrent à la même ressource Musée de Louvre. Donc, c'est possible d'ajouter le lien " http ://www.w3.org/2002/07/owl#sameAs " à la page d'accueil dans l'ensemble de données DBpedia, en indiquant que les URIs http ://fr.dbpedia.org/resource/Musée_du_Louvre et http ://de.dbpedia.org/resource/Louvre qui sont utilisées pour identifier la ressource "Musée de Louvre" dans DBpedia se réfèrent au même objet du monde réel.

```
@prefix owl: <http://www.w3.org/2002/07/owl#> .

<http://fr.dbpedia.org/resource/Musée_du_Louvre> owl:sameAs <http://de.dbpedia.org/resource/Louvre> .
```

Fig. 1.11 : Exemple de deux URIs réfèrent à la même entité du monde réel

L'établissement des liens d'identité est indispensable pour la raison que l'utilisation de différentes URIs permet aux consommateurs des données sur le web de comprendre le sens exact qu'un éditeur veut dire à propos une ressource.

Le web de données liées repose sur la résolution de la déduplication des entités d'une façon évolutive (un nombre élevé de liens owl :sameAs est ajoutée au cours du temps) et distribuée (L'effort de la création des liens owl :sameAs peut être partagé entre les différentes fournisseurs de données).

En plus du lien owl :sameAs, il existe plusieurs prédicats qui permettent d'exprimer des liens d'identité entre les entités du web de données qui se réfèrent au même objet du monde réel (skos :closeMatch, skos :exactMatch, skos :relatedMatch) [10]. Aujourd'hui, owl :sameAs est largement utilisé dans le contexte de données liées et des centaines de millions de liens owl :sameAs sont publiés sur le web.

1.4 Contexte du projet

Le web de données, appelé aussi le web sémantique couvre de nombreux domaines, tels que des personnes, des entreprises, des films, de la musique, des lieux, des livres, des publications et des données scientifiques. Avec la croissance des sources de données disponible sur le web, le problème d'hétérogénéité de données dans ces ressources augmente, alors le besoin d'accès à toutes ces ressources a été le challenge de nombreuses recherches dans le domaine de web sémantique. Donc, c'est nécessaire d'offrir une vue unifiée aux ensembles des données liées par la génération des nouveaux sources des données tout en combinant les informations distinctes et hétérogènes.

1.4.1 Problématiques abordées

La tâche d'alignement d'entités est définie comme le processus de connecter des entités représentant le même objet du monde réel mais représentées différemment dans différentes sources. Ces alignements sont exploités pour interroger plusieurs sources de données simultanément, pour générer des nouvelles sources de données plus riches cohérentes et propres. La fusion de ces sources permet d'accroître la complétude et la concision des sources publiées sur le web.

La tâche d'alignement d'entités est importante. Ainsi, trois défis majeurs se posent lors de l'alignement d'entités :

- Le premier défi réside dans le volume élevé des entités stockées dans les sources RDF. Ainsi, plusieurs approches d'alignement d'entités ont une complexité quadratique

(Nécessite O (S*T) comparaisons) parce qu'elles comparent chaque entité de la base de connaissances source S avec toutes les entités avec de la base de connaissances target T. Alors, la réduction de la complexité temporelle est une exigence nécessaire.

- Le deuxième défi réside dans l'hétérogénéité des données liées. Dans le contexte de web sémantique, nous ferons référence à l'hétérogénéité des données par la différence entre la description d'une entité dans deux bases de connaissances différentes. Ces hétérogénéités de donnés sont vaguement liées à une riche diversité de schémas, allant des noms d'attributs aux annotations de style-tag, types des valeurs et au multilinguisme.

- Le troisième défi réside dans le bruit des données publiées sur le web, ces données sont publiées via un logiciel gratuit qui ne peut pas filtrer les informations de mauvaise qualité. En conséquent, les données présentent des fautes d'orthographes, des informations redondantes, des valeurs incohérentes…

1.4.2 Objectifs

L'objectif du travail est de proposer une approche d'alignement d'entités entre les ensembles de données liées qui couvre un nombre important d'hétérogénéités. Par conséquent, les objectifs de ce projet sont énumérés dans ce qui suit :

Objectif 1: Nous cherchons à résoudre le problème d'hétérogénéité sémantique qui existe dans les ensembles de données liées en proposant une classification des différents types d'hétérogénéités. Ainsi, notre objectif est d'explorer ces problèmes d'hétérogénéité et de les classifier.

Objectif 2: Notre approche utilise des mesures de similarité qui sont à base de chaînes de caractères (par exemple Levenshtein, Jaccard, TF/IDF …), des mesures numériques, des mesures de dates ou des coordonnées géographiques pour calculer la similarité entre les entités.

Objectif 3: Le nombre de sources des données disponibles sur le web augmente rapidement. Donc, ça nécessite des méthodes pour réduire le nombre de comparaisons lors de l'alignement d'entités et par conséquent optimiser la complexité temporelle et éviter le temps d'exécution quadratique.

1.5 Conclusion

Dans ce chapitre, nous avons mis en évidence le contexte de notre projet, les problématiques abordées et les objectifs de ce projet. En outre, nous avons présenté les notions de base sur les données liées. D'autre part, nous avons énuméré les différents éléments qui constituent les données liées. Pratiquement, l'alignement d'entités est un processus qui permet de réduire l'hétérogénéité des données et de fournir une vue unifiée des différentes sources de données. Il existe, dans la littérature, plusieurs approches qui visent à aligner les entités dans les graphes RDF. Le chapitre suivant introduit en détail différentes techniques d'alignement d'entités.

Chapitre 2

Etat de l'art : Alignement d'entités dans les graphes RDF

2.1 Introduction

Le présent chapitre est organisé comme suit : Dans la première section, nous présentons la structure d'une base de connaissances RDF et une définition formelle d'une base de connaissances RDF. Dans la section 2, nous commençons par l'explication de la problématique principale de la tâche d'alignement d'entités par des exemples. Dans la section 3, nous présentons les différentes approches d'alignement d'entités. En outre, nous proposons des tableaux comparatifs de ces approches. Section 4 présente des mesures de similarité que nous allons utilisé dans notre travail. Section 5 présente des problèmes non traités dans les approches connexes qui peuvent être résolus par notre approche et nous concluons après ce deuxième chapitre.

2.2 Structure d'une base de connaissances RDF

Nous proposons l'utilisation d'un modèle de données basé sur le modèle de données du Web sémantique standard, Resource Description Framework. RDF est un modèle de données simple composé d'instructions de triples sous la forme de triplets sujet-prédicat-objet. Par exemple, " L'acteur Jean Dujardin joue le rôle de George Valentin" est une déclaration (ou triple) où "L'acteur Jean Dujardin" est le sujet, "George Valentin" est l'objet et "joue" est le prédicat. De cette façon, tous les relations compliquées peuvent être représentées dans le modèle RDF par des objets et des relations binaires comme dans le modèle orienté objet.

RDF peut être représenté sous la forme d'un graphe avec des « nœuds », également appelés ressources (c'est-à-dire des instances et des concepts), et « liens ou relations dirigés » entre les nœuds, aussi appelé propriétés ou prédicats.

On utilise le mot base de connaissances, knowledge base ou RDF comme étant la même chose. La définition suivante décrit formellement c'est quoi une base de connaissances RDF :

Définition 1 (Base de connaissances) Une base de connaissances KB est un ensemble de faits sous la forme (V,P,E) où :

– V est un ensemble fini de sommets. V est l'union disjointe $C \cup I \cup L$ où : C est l'ensemble de toutes les classes, I est l'ensemble de toutes les instances, et L est l'ensemble des valeurs de littéraux.

– P est un ensemble fini de prédicats.

– E est un ensemble fini d'arcs de la forme p(v1,v2) où : $p \in P$ et $v1, v2 \in V$.

La spécification d'un domaine à travers une ontologie et un ensemble de données relatives à cette ontologie constitue ce qu'on appelle une connaissance dirigée par une ontologie. Une base de connaissances cible généralement un domaine donné. Il est formé d'une ontologie qui représente la connaissance axiomatique du domaine, et un ensemble d'informations établissant les faits relatifs à ce domaine [27].

La figure 2.1 montre un sous-ensemble d'une base de connaissances sur le domaine du cinéma. La partie supérieure du schéma montre les connaissances ontologiques. Elle exprime notamment qu'un long métrage est un type de film, qu'un acteur est un type de personne et qu'un rôle d'acteur est un type de rôle. Ce fragment d'ontologie déclare également trois prédicats exprimables entre les instances de ces classes. Ces prédicats sont d'ailleurs exploitées dans la partie inférieure du schéma, relative aux assertions, qui exprime que l'acteur Jean Dujardin joue le rôle de George Valentin dans le film The Artist.

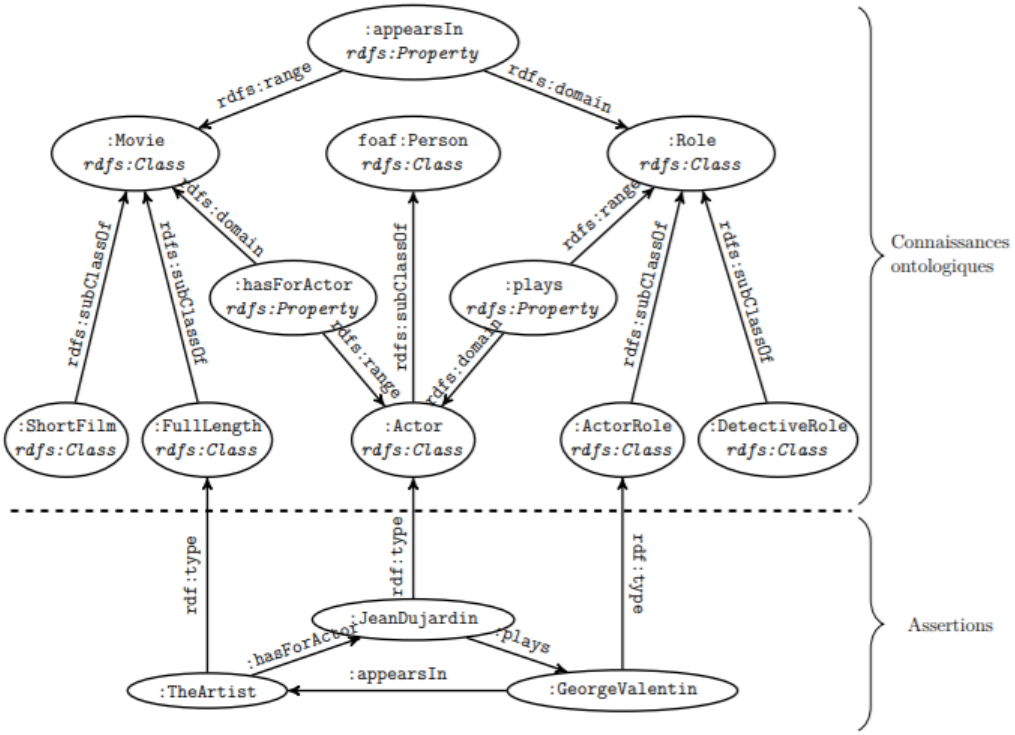

FIG. 2.1 : Sous-ensemble d'une base de connaissances sur le domaine du cinéma [27]

Par raison de clarté, on remarque que les prédicats présentent comme des nœuds dans la partie ontologique alors qu'elles sont utilisées sur les arcs dans la partie relative aux assertions.

2.3 Hétérogénéité des données RDF

La détection des liens d'identités entre les ensembles de données RDF est une tâche cruciale et difficile. Le principal défi rencontré par cette tâche est que les données fournies par différentes sources sont très hétérogènes, ainsi, deux entités faisant référence au même objet du monde réel sont décrites et structurées d'une manière très différente.

Ainsi, pour résoudre ce problème, il faut se situer par rapport aux différents types d'hétérogénéité rencontrés au niveau des données RDF[1]. La figure 2.2 présente un exemple.

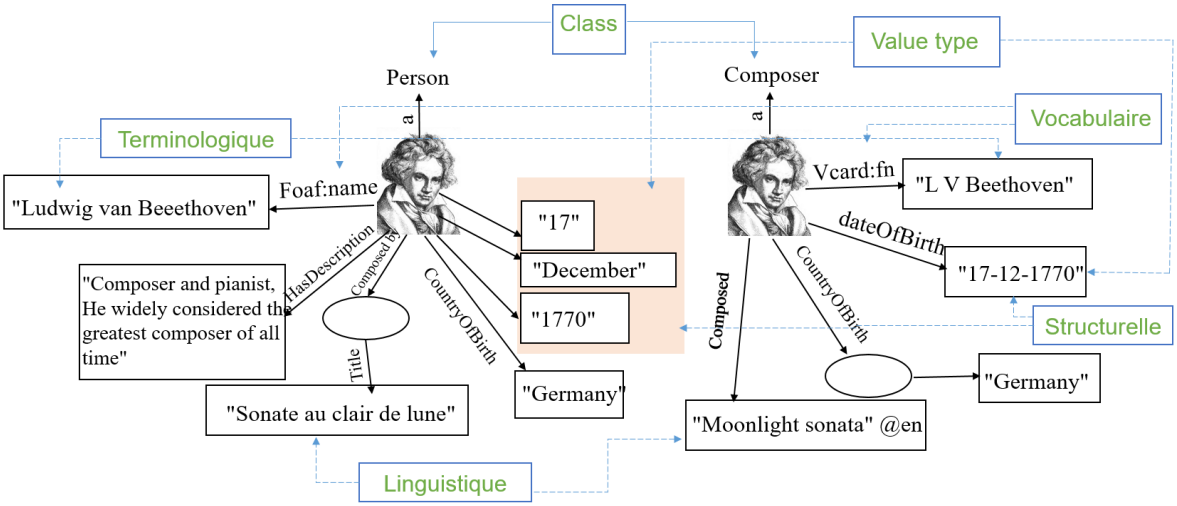

FIG. 2.2 : Description de la même entité Beethoven dans deux bases différentes [1]

Comprendre l'hétérogénéité des données permet d'identifier et d'analyser le problème d'alignement d'entités et donc de proposer des meilleures solutions. Dans le contexte du web sémantique, il est possible d'observer plusieurs types d'hétérogénéité et à différentes dimensions : hétérogénéité de schémas (classe et prédicats), hétérogénéité de valeurs ou de structuration. On considère ainsi trois types majeurs d'hétérogénéité des données (données, schéma et sémantique). Pour illuster les différents types d'hétérogénéités, nous utiliserons un exemple fictif, montrant la description d'une entité du monde réel (le compositeur Ludwing van Beethoven) dans deux bases de connaissances différentes.

2.3.1 Dimension données

Les valeurs des prédicats sont une source importante d'hétérogénéités.

- Hétérogénéité terminologique : Correspond à l'existance de plusieurs valeurs littérales utilisées pour désigner la même information. Ceci comprend les problèmes bien connus liés à la synonymie, à la polysémie ou aux variations d'orthographe (faute de frappe, acronymes, abréviations, etc..). Par exemple, le nom du compositeur "Ludwing van Beethoven" et "L V Beeth".

- Hétérogénéité linguistique : Correspond à l'utilisation de différentes langues pour désigner la même information. Le multilinguisme est considéré comme un défi majeur pour résoudre le problème d'alignement d'entités, plusieurs études proposent des solutions basées sur la traduction automatique en s'appuyant sur la source lexico-sémantique telle que BabelNet.
- Hétérogénéité de type des valeurs (value type heterogeneity) : Ce type d'hétérogénéité concerne les différences dans le codage des données, on trouve par exemple une valeur d'âge présente sous forme d'une chaîne de caractères ou d'un nombre. Par exemple, une date peut être représentée soit au format standard d'une date soit au format chaîne.

2.3.2 Dimension du schéma

Nous allons discuter les différences liées aux schémas dans les graphes RDF.

- Hétérogénéité de vocabulaire : L'utilisation de différents ontologies et vocabulaires est un problème difficile dans le contexte de Web de données. Exemple : le vocabulaire vCard et le vocabulaire FOAF sont deux vocabulaires RDF utilisées pour décrire une entité de la classe personne. Chaque vocabulaire a ses propres prédicats pour décrire une entité. Par exemple : foaf :name et vcard :fn.
- Hétérogénéité structurelle : La description d'une entité peut se faire à différents niveaux de granularité, c'est le cas de la date de naissance de Beethoven, cette information soit donnée dans un seul champ d'information soit répartie sur plusieurs prédicats.

2.3.3 Dimension sémantique

Nous décrivons deux principaux problèmes d'hétérogénéité dans ce groupe.

- Hétérogénéité de classe : C'est le cas ou deux entités appartenant à des classes différentes. En se référanat à l'exemple de la figure 2.1, deux instances représentant le même objet du monde réel mais appartenant à deux classes différentes (« Person » et « Composer »).
- Hétérogénéité de propriétés : Dans le cas des différents prédicats utilisés pour présenter les mêmes informations. Par exemple, les deux prédicats "composedBy" et "composed".

2.4 Approches existantes pour l'alignement d'entités

Ces approches sont conçues pour effectuer de l'alignement entre deux bases de connaissances source et cible.

2.4.1 Approche Legato

Legato [1] est une approche permettant de couvrir différents types d'hétérogénéités et vise à réduire l'intervention humaine. C'est une approche contenant quatre étapes : le filtrage des prédicats pour déterminer l'ensemble de prédicats discriminants pour comparer les entités. Le module d'instance profiling représente chaque ressource par un sous-graphe pertinent pour la tâche de comparaison, ainsi chaque ressource est représentée par un sac de mots qui contient toutes les informations.

Legato utilise la mesure TF-IDF pour comparer la similarité entre les descriptions des entités. Cette approche permet de donner une représentation vectorielle de l'entité et de calculer la similarité entre les vecteurs par la mesure du cosine similarity.

Le problème avec l'approche Legato qu'elle ne résout pas le problème de l'hétérogénéité du schéma qui est le type d'hétérogénéité le plus pertinent. c'est une approche qui est testée seulement avec des entités de la même classe. Aussi, Legato se réduit à l'ensemble de prédicats discriminants pour comparer les entités fournies par l'utilisateur mais faute de connaissance de la nature des données, ne permet pas de rendre compte automatiquement d'un grand nombre d'hétérogénéités structurelles.

2.4.2 Approche RDF-AI

RDF-AI [28] est un framework qui vise à aligner les entités entre deux bases de connaissances source et cible. Il permet de couvrir deux types d'hétérogénéités. L'hétérogénéité terminologique liée aux fautes de frappe, aux abréviations et aux synonymes en utilisant le Wordnet et des mesures de similarité syntaxiques pour calculer la similarité entre deux entités. Le deuxième type d'hétérogénéité est le multilinguisme, il traduit les informations à la même langue en utilisant l'API Google translate.

RDF-AI nécessite de l'intervention humaine, il utilise un fichier de configuration qui permet à l'utilisateur de sélectionner les prédicats les plus pertinents pour comparer les entités. L'utilisation de ce framework est très couteuse en termes du temps et ne permet pas de rendre compte des hétérogénéités de dimension ontologique.

2.4.3 Approche SILK

SILK [36] est une autre approche d'alignement d'entités. Elle combine des mesures de similarité pour calculer la similarité totale entre une paire d'entités. Elle soutient des mesures numériques, de date, ou des coordonnées géographiques pour calculer la similarité entre les entités. Pour agréger les différents mesures de similarité, il utilise l'opérateurs agrégé telles que min, max, moyenne (pondérée), etc.

Afin d'atteindre une complexité temporelle quasi-linéaire, SILK met en œuvre une technique de blocage efficace qui permet de grouper les entités potentiellement similaires dans un seul bloc. Cette technique de blocage est appelé MultiBlock. Elle utilise un indice multidimensionnel dans lequel les objets similaires sont situés à proximité les uns des

autres. SILK nécessite l'intervention d'un expert pour définir les prédicats qui doivent être utilisés dans l'étape de matching.

Par exemple, pour déterminer les entités qui désignent les mêmes villes dans le monde réel, l'expert précise dans un fichier script que des paires de villes sont comparées en fonction de leurs noms et leurs prédicats de population. Pour comparer les noms, la métrique de similarité de chaîne de Levenshtein est utilisée, tandis que pour la population, une mesure de similarité proposé par Silk pour les valeurs numériques est appliquée. Deux villes sont considérées comme co-référents si la similarité moyenne de ces deux prédicats est plus grande que 0,95.

2.4.4 Techniques de blocking

Dans [25], l'auteur met en œuvre des techniques de blocage pour atteindre une complexité temporelle acceptable lors de l'alignement d'entités dans de grands jeux de données.

Le volume des données ainsi que le nombre des entités dans le web sont très grands. Les algorithmes itératifs qui comparent chaque entité avec toutes les autres entités sont très couteux et ont une complexité quadratique : O (n2). Afin de réduire le nombre de comparaisons inutiles, les approches de blocking groupent les entités potentiellement similaires. Ainsi, ils sont généralement utilisées comme une étape de prétraitement pour la tâche d'alignement d'entités pour réduire le nombre de comparaisons inutiles.

Cette technique de blocage est basée sur l'idée suivante pour chaque token ti (un mot) crée un bloc bi qui contient toutes les entités ayant ti dans leurs descriptions, autrement dit, dans les valeurs des attributs, quel que soit l'attribut. De cette façon, les blocs sont construits de manière indépendante des attributs, ce qui permet d'éviter les hétérogénéités de dimension du schéma comme l'hétérogénéité de vocabulaire guidé par les valeurs littérales.

Cette technique permet d'optimiser le temps d'exécution, elle calcule la similarité entre les paires d'entités dans un même bloc et retourne le résultat de matching à l'utilisateur.

2.5 Mesures de similarité

Dans cette section, nous allons présenter des méthodes de calcul de similarités qui visent à trouver des relations de similarités entre les entités. Ainsi, une mesure de similarité permet de mesurer le degré de ressemblance entre une paire d'entités. Les différentes mesures de similarité utilisées dans la littérature compris des mesures de similarité à base de chaînes de caractères, des mesures de similarité numérique, de date et des mesures de similarité à base du Wordnet. Les différentes mesures de similarité visent à calculer le score de la similarité entre deux chaînes de caractères. Une mesure de similarité est une valeur entre [0,1], une valeur proche de 1 indique une grande similarité entre une paire de chaînes.

2.5.1 Les mesures de similarité à base de chaînes caractères

Une mesure de similarité de chaînes de caractères calcule le rapport de la partie commune entre deux chaînes de caractères [4].

- La distance de Levenshtein : appelée aussi Edit Distance, est utilisée pour mesurer la similarité entre des chaînes de caractères qui peuvent contenir des fautes d'orthographe. Plus cette distance est petite, plus les deux chaînes sont similaires. Cette distance Levenshtein permet de calculer le coût minimal de transformation d'une chaîne x en une chaîne y. Cette transformation est effectuée soit par l'insertion d'un caractère, l'ajout d'un caractère ou la suppression d'un caractère. La fonction de similarité associée à la distance de Levenshtein d(x,y) est la suivante :

$$Leven(x,y) = 1 - \frac{d(x,y)}{max(length(x), length(y))} \tag{2.1}$$

- La distance de Jaro est utilisée dans le cas de deux chaînes courtes, comme les noms et les prénoms. Elle considère le nombre et la proximité des caractères communs entre les deux chaînes. La fonction de similarité Jaro est définie comme suit [7] :

$$Jaro(x,y) = \frac{1}{3} * [\frac{c}{|x|} + \frac{c}{|y|} + \frac{\frac{c-t}{2}}{|c|}] \tag{2.2}$$

  Avec :

  c est le nombre des caractères en communs entre les deux chaînes x et y.

- La distance q-gram est conçue pour mesurer la similarité entre deux chaînes longues telles que les commentaires de deux entités. Elle est basée sur le nombre de séquences qui se trouvent à la fois dans les deux chaînes de caractères et elle est calculée comme suit [34] :

$$q - gram(x,y) = \frac{|ngram(x,n) \cap ngram(y,n)|}{min(|x|,|y|) - n + 1} \tag{2.3}$$

  Avec : ngram(x,n) est l'ensemble de sous-chaînes de x de longueur n.

- TF-IDF avec cosine-similarity : Le choix de la mesure adéquat qui estime la similarité entre deux attributs est l'étape la plus critique. Elle repose sur deux éléments :

  -Le modèle qui représente collectivement l'ensemble des valeurs associées à chaque attribut

  -La métrique qui évalue la similarité entre la représentation entre deux valeurs.

  La combinaison d'un modèle de représentation et d'une métrique de similarité est appelée paramètres de clustering. Dans ce qui suit, nous détaillons trois de ces paramètres qui ont été établi dans plusieurs approches.

Le terme "modèle de représentation" transforme un ensemble de valeurs V en un espace cartésien ou chaque dimension correspond à un mot distinct contenu dans V. Ainsi, un attribut nk est représenté par un vecteur dans la i-ème coordonnée désigne le poids TF(ti)*IDF(ti) du terme correspondant ti. Avec TF(ti) représente la fréquence du terme ti, c'est-à-dire, le nombre de fois le terme ti est associé à l'attribut nk. Tandis que IDF(ti) est égal à log(|n|/|N(ti)), où N signifie l'ensemble des attributs d'entrée et N(ti) correspond au sous-ensemble d'attributs contenant le terme ti. Donc ce poids est une mesure statistique utilisée pour évaluer l'importance d'un terme dans un ensemble de valeurs V.

Par exemple : le pair d'attribut <nk, vk> =< name, "home phone"> est représenté par le vecteur TF (home)*IDF(home) , TF(phone)*IDF(phone) ,0,….,0.

Dans ce contexte, la pertinence entre deux attributs n1 et n2 est quantifiée par la fonction cosine-similarity des vecteurs correspondants :

$$Sim(n1, n2) = \frac{n1 * n2}{||n1|| * ||n2||} \tag{2.4}$$

Cette mesure de similarité prend une valeur entre [0,1], une valeur proche de 1 indique une grande similarité entre une paire de chaînes (attributs).

2.5.2 Les mesures de similarité numériques

Les attributs numériques, par exemple, la forme de date de naissance d'une personne existe sous différents formats telles que : "1955-02-24", "24/02/1955" ou "24th Feb. 1955". Pour faire face à ces problèmes, nous normalisons les valeurs au format les plus populaires, par exemple, "aaaa-mm-jj" pour la date. Puis, nous extrayons tous les nombres de la chaîne via une expression régulière et définissons la mesure de similarité suivante pour calculer la similarité de leurs valeurs à partir des valeurs normalisées par la fonction [13] :

$$Sim(num1, num2) = \frac{|num1 - num2|}{max(num1, num2) + 1} \tag{2.5}$$

Si la valeur de cette similarité est inférieure à 0.01 alors les deux valeurs sont considérées équivalentes. Cette mesure de similarité est utilisée dans le cas d'autres attributs comme la surface, la longueur, la hauteur et la population.

2.5.3 Les mesures de similarité pour les coordonnées géographiques

La distance euclidienne est conçue pour mesurer la similarité entre deux coordonnées géographiques. Soient deux points géographiques, (latitude1, longitude1) les coordonnées géographiques du premier point et (latitude2, longitude2) les coordonnées géographiques

du deuxième point. La distance euclidienne entre ces deux points est définie comme suit :

$$Eucld(point1, point2) = \sqrt{(latitude1 - latitude2)^2 - (longitude1 - longitude2)^2} \quad (2.6)$$

Cette distance euclidienne permet de calculer la similarité par la formule suivante :

$$Sim(point1, point2) = \frac{1}{1 + Eucld(point1, point2)} \quad (2.7)$$

2.5.4 Les mesures de similarité à base de Wordnet

Wordnet [20] est une base de données lexicale, permettant de regrouper les mots anglais dans des ensembles de mots équivalents appelés synsets. Un exemple d'un ensemble de mots équivalents pour le mot "car" est : auto, machine, car, automobile, motorcar. Cette base de données donne une brève définition des homonymes et des hyponymes de ces synonymes. Il existe des versions de Wordnet en plusieurs langues mais la version la plus complète est la version anglaise. En effet, Wordnet est considéré comme un mélange d'un dictionnaire et un thésaurus, puisqu'il contient des concepts qui sont interconnectés par des relations sémantiques. Alors, il est utilisé pour établir la distance sémantique entre deux concepts en apportant six mesures de similarité et trois mesures de connexité. Parmi les mesures de similarité à base de Wordnet :

La mesure **WUP (Wu and Palmer)** est basée sur la profondeur des LCS de concepts en utilisant la somme de profondeurs des concepts individuels [37] . Elle est définit comme suit :

$$simWUP(c1, c2) = \frac{2 * depth(lcs(c1, c2))}{len(c1, lcs(c1, c2) + len(c2, lcs(c1, c2) + 2 * depth(lcs(c1, c2))))} \quad (2.8)$$

Avec : depth(lcs(c1,c2)) est la profondeur globale de la hiérarchies.

La mesure **PATH** est une mesure de référence qui est définit par l'inverse du plus court chemin entre deux concepts. Et elle représentée par la formule suivante :

$$simPATH(c1, c2) = \frac{1}{len(c1, c2)} \quad (2.9)$$

Avec : len(c1,c2) est la longueur du plus court chemin entre c1 et c2.

La mesure **LCH (Leacock and Chodorow)** est une mesure de similarité sémantique permettant de trouver le plus court chemin entre deux concepts dans le Wordnet en se basant sur la longueur maximale trouvée dans une hiérarchie [16]. Elle est définit comme suit :

$$simLCH(c1, c2) = \log(\frac{len(c1, c2)}{2 * maxdepth(c)}) \quad (2.10)$$

Avec : len(c1,c2) est la longueur du plus court chemin entre c1 et c2. maxdepth(c) est la longueur maximale du chemin trouvée entre ls concepts c1 et c2 et la racine.

2.6 Synthèse

Le tableau 2.1 illustre une comparaison entre les approches d'alignement d'entités. Cette comparaison est faite en utilisant plusieurs critères. Le premier critère étudie les différentes mesures de similarité ainsi que les sources de connaissances qui sont utilisées dans le processus de découverte des liens d'identités. Le deuxième critère permet de comparer le positionnement de ces approches par rapport aux types des hétérogénéités.

Ces approches utilisent des techniques de blocage et des méthodes d'indexation pour réduire la complexité temporelle du processus d'alignement d'entités. Toutes ces approches utilisent des fonctions de similarité à base de chaînes de caractères telles que Levenshtein, Jaccard, Cosine-similarity, etc. Cependant, l'approche SILK utilise des mesures numériques, de date ou de coordonnées géographiques pour calculer la similarité entre les paires d'entités.

La plupart de ces approches n'utilisent pas des sources de connaissances externes. Alors que RDF-AI utilise le Wordnet comme une source de connaissance externe pour calculer la similarité sémantique entre les entités.

	Legato	**RDF-AI**	**SILK**
Types de mesures de similarité	Mesure TF-IDF avec cosine-similarity	Mesures de similarité syntaxique	Des similarités de chaînes de caractères, numérique, date et coordonnées géographique
Utilisation des ressources de connaissances externes	Non	Wordnet	Non
Hétérogénéités résolus	Hétérogénéité de vocabulaire (En négligeant les prédicats), l'hétérogénéité descriptive (Par filtrage des prédicats discriminantes)	Multilinguisme, hétérogénéité terminologique	Hétérogénéité de type de valeur. (DataType heterogeneity)

Tab. 2.1 : Comparaison des approches d'alignement d'entités

2.7 Discussion

Le problème d'alignement d'entités dans les graphes RDF est l'un des problèmes les plus difficiles qui affectent l'interopérabilité des données liées. Dans ce chapitre, nous avons présenté les différentes méthodes d'alignement d'entités, y compris l'alignement de schémas. Nous avons aussi comparé ces approches en tenant compte des différents critères.

Ces approches peuvent prendre en entrée soit des fichiers RDF soit des requêtes SPARQL. Les sorties de ces approches sont des liens de type owl :sameAs ou même d'autres liens précisés par l'utilisateur. De plus, les approches d'alignement d'entités utilisent plusieurs mesures de similarité pour comparer les entités similaires. Afin d'améliorer leurs complexités temporelles et être capable de générer des alignements de haute qualité, un outil d'alignement d'entités devrait également générer autant que possible des liens pour assurer la complétude. En outre, les outils d'alignement d'entités cherchent à améliorer les complexités temporelles. Ainsi, une approche non évolutive peut avoir une complexité quadratique parce qu'elle évalue toutes les combinaisons possibles entre ressources de deux ensembles de données liées. Donc, le développement d'une méthode performante qui minimise le nombre des comparaisons est une exigence dans le processus d'alignement d'entités.

Contrairement aux autres approches, notre approche cherche à améliorer le résultat d'alignement en utilisant une source de connaissances externe qui est le Wordnet. Elle améliore aussi la complexité temporelle par la génération des entités candidats à l'entité source pour éviter la complexité quadratique. Elle combine aussi des mesures de simila-

rité pour calculer la similarité totale entre une paire d'entités. Elle soutient des mesures numériques, de date, ou des coordonnées géographiques pour résoudre l'hétérogénéité liée aux types des valeurs.

2.8 Conclusion

La première partie de ce chapitre présente la structure d'une base de connaissances RDF. La deuxième partie concerne les différents niveaux d'hétérogénéités entre les entités décrivant le même objet du monde réel. La troisième partie concerne les approches qui sont proposées pour découvrir les liens d'identités entre les entités. Dans la partie synthèse, nous avons comparé ces approches en tenant compte des différents critères. Le chapitre suivant introduit en détail les différentes étapes de notre approche d'alignement d'entités.

Chapitre 3

Implémentation et méthodologie

3.1 Introduction

Ce troisième chapitre est organisé comme suit : dans la première section, nous présentons un exemple illustratif pour expliquer l'objectif de notre approche, la deuxième section décrit l'architecture système pour l'alignement d'entités dans les graphes RDF et la section 3 décrit en détails les étapes de notre approche. Dans la section 4, nous décrivons en détails notre approche d'alignement d'entités et nous concluons après ce troisième chapitre.

3.2 Exemple illustratif

Dans cette section, nous présentons un exemple illustratif afin d'expliquer notre contribution. Nous choisissons Wikidata comme l'ensemble de données source et DBpedia comme l'ensemble de données cible. Notre objectif est de découvrir les entités similaires existantes dans les deux ensembles de données. Nous avons utilisé des données qui sont publiées publiquement en tant que ressource dans [9].

Pour chaque entité dans wikidata, il existe plusieurs entités candidates qui partagent le même label avec l'entité source. Ces candidats sont utilisés pour le matching ce qui permet de réduire l'espace de recherche. Par exemple, la figure 3.1 représente l'entité wikidata ayant pour uri <https ://www.wikidata.org/wiki/Q774805> et ayant comme label la chaîne "The Joke" et les candidats de cette entité dans la base de connaissances cible DBpedia qui sont identifiés par les URIs :

<https ://dbpedia.org/ressource/The_Joke_(film)>

<https ://dbpedia.org/ressource/The_Joke_(novel)>

Ces entités sont décrites différement dans les deux bases de connaissances. Il y a différents types d'hétérogénéité tant au niveau du schémas. En effet, les noms des attributs sémantiquement équivalents sont déclarés différement (par exemple « country of origin » et « country »). De même pour les valeurs d'attributs présentent de l'hétérogénéité de

type des valeurs (par exemple une valeur numérique peut être définit soit sous le format numérique comme 1969 ou bien format string ”1969”). Un autre obstacle à la tâche d'alignement d'entité provient de l'hétérogénéité linguistique qui se présente sous la forme de tag de langues. En effet, la propriété ”titre” existe en deux langues différentes : The Joke (en) et Žert (Czech).

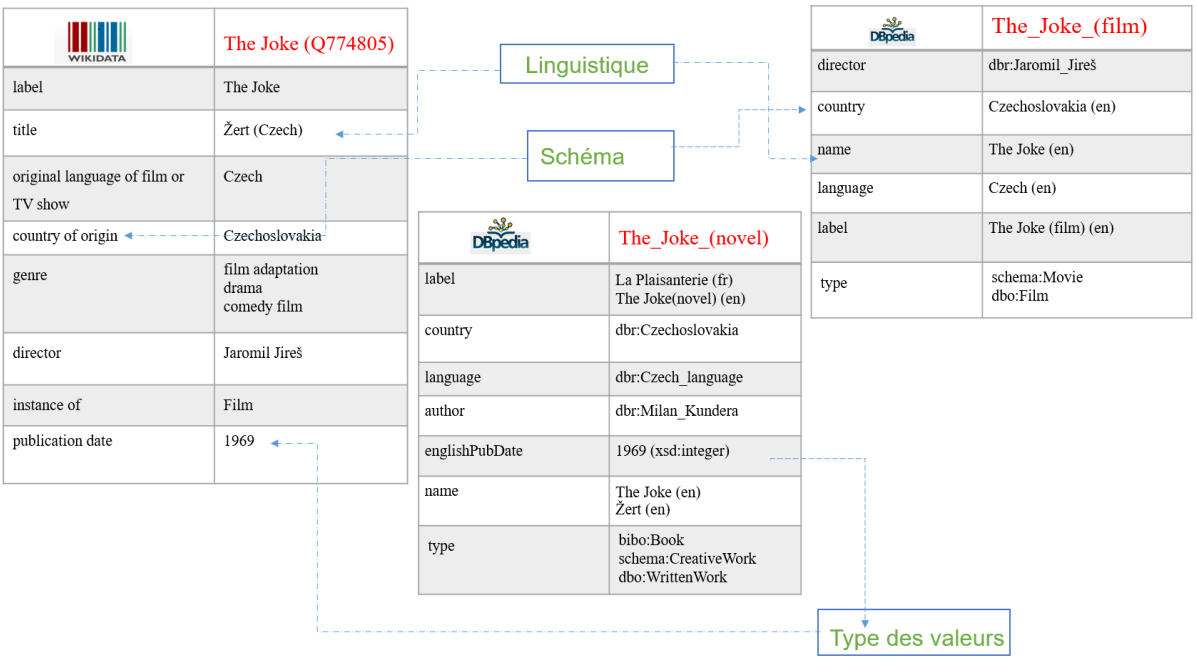

Fig. 3.1 : L'entité ”Q774805” dans Wikidata et ses deux candidats dans Dbpedia

Ensuite, par des mesures de similarité nous calculons la similarité des candidats avec l'entité source pour donner le résultat final de matching. Notre approche se base sur des mesures de similarité à base de chaînes de caractères, des mesures numériques, de dates, ou des coordonnées géographiques pour résoudre l'hétérogénéité liée aux types des valeurs. Le tableau 3.1 représente les candidats et le résultat de matching de l'entité ayant pour uri : <https ://www.wikidata.org/wiki/Q774805>.

L'entité	Les candiadts	Résultat de matching
Q774805	<https ://dbpedia.org/resource/The_Joke_(film)> <https ://dbpedia.org/ressource/The_Joke_(novel)>	<https ://dbpedia.org/resource /The_Joke_(film)>

Tab. 3.1 : Structure des données

3.3 Aperçu général de notre approche

Cette section couvre un aperçu général de notre système d'alignement d'entités et décrit les différents composants du système.

La figure 3.2 présente un aperçu général de notre approche. Ainsi, notre système se compose de deux blocs essentiels :

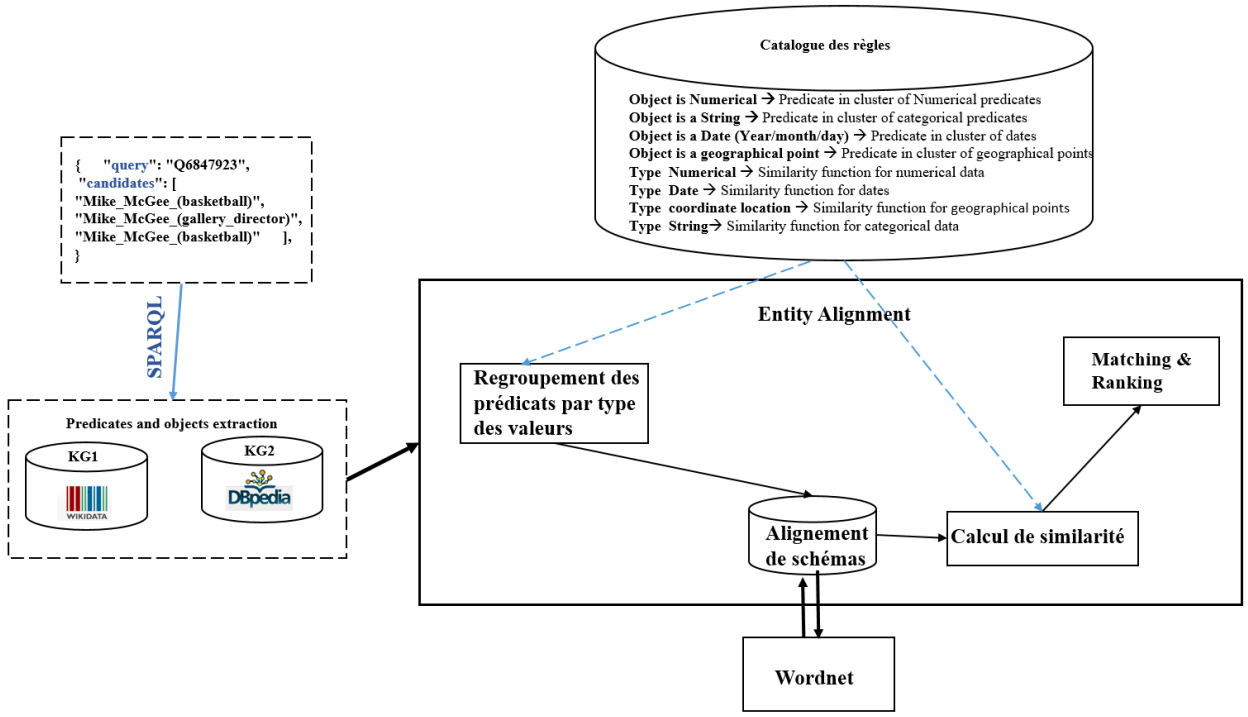

FIG. 3.2 : L'architecture système de l'alignement d'entités

Extraction des prédicats et leurs valeurs : La tâche principale de l'étape de collecte des données est l'extraction des prédicats et leurs valeurs décrivant une entité. Nous considérons chaque entité comme le sujet de notre triplet RDF. Dans cette étape, notre système collecte tous les prédicats et les valeurs de chaque entité donnée (source et candidat) en utilisant une requête SPARQL (par une exécution en ligne).

Tel qu'illustré par la figure 3.3, un exemple d'une requête pour collecter les prédicats et leurs valeurs décrivant l'entité identifiée par l'uri : <https ://www.wikidata.org/wiki/Q6847923>

Catalogue des règles : Notre approche est une approche basée sur des règles. Un catalogue de règles est prédéfini pour le regroupement des prédicats par types des valeurs et pour la mesure de similarité entre les entités. Les règles sont basées sur les expressions régulières permettant de regrouper les prédicats par types des valeurs.

Par exemple, le module de regroupement des prédicats par types des valeurs dans l'architecture recourt à la règle : Un prédicat est considéré comme date si la valeur qui lui est associé est de format date ('aaaa/mm/jj').

Une autre règle pour le module de mesure de similarité entre les entités : Si l'objet est considéré de format numérique alors la mesure de similarité utilisée est la mesure numérique.

Entity alignment : Cette phase se compose de quatre modules : regroupement des prédicats par type des valeurs, le nettoyage des prédicats en double, alignement de schémas, sélection de règles pertinentes et le module de matching et ranking après le calcul de similarité.

Le regroupement des prédicats par types des valeurs permet d'améliorer le temps

Fig. 3.3 : Requête SPARQL pour la collecte des prédicats et leurs valeurs décrivant l'entité identifiée par l'uri https ://www.wikidata.org/wiki/Q6847923

d'exécution et par conséquent réduire le nombre de comparaisons entre les prédicats de l'entité source et ses candidats.

Le nettoyage des prédicats en double permet de réduire l'espace de recherche et par conséquent réduire le nombre de comparaisons à faire entre les prédicats. En outre, cette étape permet de transformer et de nettoyer les données en entrée.

La phase principale est le matching. En particulier, elle évalue des mesures de similarité spécifiées sur les paires d'entités. Les mesures de similarité utilisées sont à base des chaînes de caractères (par exemple Jaro Similarity, Jaccard, TF/IDF ...), des nombres ou des types de données spécifiques à un domaine (par exemple les dates ou les coordonnées géographiques). La dernière phase de l'approche de l'alignement d'entités permet de combiner les résultats du matching, classe la liste des candidats reçue par similarité par rapport à l'entité source. Il mesure la moyenne pondérée entre chaque paire d'entités et sélectionne le candidat qui maximise cette valeur moyenne avec l'entité source.

3.4 Nouvelle approche pour l'alignement d'entités

3.4.1 Collecte des données

Nous présentons, dans cette section, deux étapes pour collecter des données à partir de DBpedia et Wikidata. Le processus de la collecte de données liées prend en entrée une entité source de wikidata et des entités candidats de DBpedia et requête tous les prédicats et leurs valeurs décrivant une entité.

Nous avons utilisé des données qui sont publiées en tant que ressource dans [9]. Ces données présentent pour chaque entité sources de wikidata : une liste des candidats de dbpedia, et le résultat de matching.

Exemple :

”Query” : ”Q701169 ”,

”Candidates” : [

”Hintersee,_Austria”,

”Hintersee,_Mecklenburg-Vorpommern”

],

”Answer” : ”Hintersee,_Austria”

Par conséquent : L'entité ”Hintersee”, présentant une municipalité à Austria, est identifiée par l'uri <http ://www.wikidata.org/resource/Q701169> dans Wikidata et a deux candidats dans DBpedia qui sont identifiés respectivement par ces deux URIs : <https ://dbpedia.org/resource/ Hintersee,_Austria > et <https ://dbpedia.org/resource/ Hintersee,_Mecklenburg-Vorpommern>. Le résultat de matching est l'entité ayant pour uri <https ://dbpedia.org/resource/Hintersee,_Austria >.

Pour requêter tous les prédicats et leurs valeurs, nous considérons chaque entité comme le sujet de notre triplet RDF. Dans cette étape, notre système collecte tous les prédicats et leurs valeurs décrivant de chaque entité donnée (source et candidat) en utilisant une requête SPARQL tel qu'illstré dans la figure 3.3.

À titre d'exemple, les tableaux 3.2 et 3.3 présentent les prédicats et les valeurs qui sont reliés à l'entité source et aux candidats.

Entité source : <http ://www.wikidata.org/resource/Q311308> (tableau 3.2)

Prédicats	Valeurs
Label	Hintersee, Austria (en)
Description	municipality in the district of Salzburg-Umgebung in the state of Salzburg in Austria
instance of	rural municipality of Austria
country	Autria
located in the administrative territorial entity	Salzburg-Umgebung District
coordinate location	Point (13.266666666 47.716666666)
elevation above sea level	746
postal code	5324
local dialing code	06224
Commons category	Hintersee, Salzburg

Tab. 3.2 : Liste des prédicats et leurs valeurs décrivant l'entité source ”Q701169”

Candidate 1: <https ://dbpedia.org/resource/ Hintersee,_Austria > (tableau 3.3)

Prédicats	Valeurs
rdfs :label	Hintersee, Austria (en)
dbo :abstract	Hintersee is a municipality in the district of Salzburg-Umgebung in the state of Salzburg in Austria. (en)
dbo :areaCode	06224
dbo :country	dbr :Autria
dbo :elevation	746.000000 (xsd :double)
dbo :postalCode	5324
dbp :areaCode	6224 (xsd :integer)
dbp :elevationM	746 (xsd :integer)
dbp :name	Hintersee (en)
dbp :pushpinMap	Austria (en)
georss :point	47.71666666666667 13.266666666666667
geo :geometry	POINT(13.266666412354 47.716667175293)
geo :lat	47.716667 (xsd :float)
geo :long	13.266666 (xsd :float)

Tab. 3.3 : Liste des prédicats et leurs valeurs décrivant l'entité identifiée par l'URI " https ://dbpedia.org/resource/ Hintersee,_Austria "

3.4.2 Prétraitement et nettoyage des données

Après la collecte des prédicats et leurs valeurs décrivant les entités source et les candidats, notre approche applique des méthodes de prétraitement et nettoyage des données collectées. En effet, l'étape de prétraitement est très importante dans le processus d'alignement d'entités. Des données comportent des erreurs et du bruit risquent de donner des faux alignements. Ainsi, nous avons procédé à un nettoyage des données textuelles pour lemmatiser les valeurs et supprimer les symboles de ponctuations et les stop words.

Tokenization : Cette étape consiste à diviser le texte en petits morceaux qui sont plus faciles à traiter, généralement nous utilisons l'espace comme délimiteur de division pour la langue anglaise, ce qui ne sera pas le cas pour d'autres langues comme l'allemand.

Elimination de stop words : Cette étape consiste à supprimer les mots très communs et très utilisés dans tous les textes tels que : (Il, elle, est, dans…). Ainsi, leur présence peut affecter la performance du processus de comparaison entre les entités.

Stemming/ lemmatisation : Les deux termes sont utilisés pour générer la forme racine d'un mot spécifique. La différence entre eux est que la racine peut ne pas être un mot réel alors que le lemme est un mot dérivé du dictionnaire. Par exemple, la racine du mot "beautiful" et "beauty" est "beauti" qui n'a pas de sens le dictionnaire anglais contrairement au lemme qui serait "beautiful".

3.4.3 Regroupement des prédicats par types des valeurs

Après le prétraitement des données collectées, notre système classe les prédicats et les valeurs collectés en quatre classes : chaîne de caractères, date, coordonnées géographiques et valeur numérique. Nous notons que ce regroupement permet de simplifier la comparaison (Voir exemple : tableau 3.4).

Prédicats	Valeurs	Classe
Label	Hintersee	Chaîne de caractères
Description	municipality in the district of Salzburg-Umgebung in the state of Salzburg in Austria	Chaîne de caractères
instance of	rural municipality Austria	Chaîne de caractères
country	Autria	Chaîne de caractères
located in the administrative territorial entity	Salzburg Umgebung District	Chaîne de caractères
coordinate location	Point (13.266666666 47.716666666)	Coordonnées géographiques
elevation above sea level	746	Valeur numérique
postal code	5324	Valeur numérique
local dialing code	06224	Valeur numérique
Commons category	Hintersee, Salzburg	Chaîne de caractères

Tab. 3.4 : Classification des prédicats et des valeurs décrivant l'entité source "Q701169"

Ce regroupement est fondé sur les expressions régulières. Les expressions régulières permettent de reconnaître des segments dans les textes de type date, coordonnées géographiques ou bien un mot particulier comme la chaîne "Point" pour définir des coordonnées géographiques. Autrement dit, c'est un ensemble de formules permettant de caractériser un ensemble de chaînes de caractères. Une expression régulière est inscrite dans un ensemble d'alphabets de symboles Σ et de mots vides [29]. Dans ce qui suit, nous présentons dans le tableau 3.5 un ensemble de symboles utilisé pour gérer des opérations complexes sur les expressions régulières.

Symboles	Descriptions
.	Remplace n'importe quel symbole sauf le retour chariot \n
[]	Remplace l'un quelconque des symboles places entre les crochets
[^]	Remplace l'un quelconque des symboles qui ne sont pas entre les crochets
\w	Remplace n'importe quel caractère alphanumérique (lettre ou chiffre) plus le caractère
\d	Remplace n'importe quel chiffre
\D	Remplace n'importe quel symbole qui n'est pas un chiffre

Tab. 3.5 : Opérations complexes sur les expressions régulières.

Les expressions régulières ont un rôle important. Elles permettent ainsi de segmenter les textes en détectant les séparateurs.

Exemple : Trouver tous les dates dans une chaîne de caractères S.

S = ”14-2-1998 2020/05/30”

Expression : re.compile(”’([12][0-9]|3[0-1]|0?[1-9]) ([./-]) (1[02]|0?[1-9]|[1-9]) ([./-]) (2?1?[0-9][0-9][0-9])”’, re.VERBOSE)

-([12][0-9]|3[0-1]|0?[1-9]) # Pour détecter les jours de 1 à 31

-([./-]) # Pour détecter les séparateurs

-(1[0-2]|0?[1-9]|[1-9]) # Pour détecter le mois de 1 à 12

-([./-]) # Pour détecter les séparateurs

-(2?1?[0-9][0-9][0-9]) # Pour détecter l’année de 1000 à 2999

Résultat : [14, 02,1998], [30, 05,2020]

3.4.4 Regroupement des prédicats en double

Après la phase de prétraitement, notre système consiste à nettoyer les données en entrée dans la même base de connaissances et grouper les prédicats en double dans un même cluster. Cette étape est intéressante pour une entité de la base de connaissances cible DBpedia. Nous avons constaté la présence de plusieurs prédicat en double dans la base de connaissances DBpedia. Ceci est dû aux techniques d’extraction automatique qui ont lieu au graphe DBpedia. Ainsi, Dbpedia présente une source de redondance des prédicats.

Par exemple pour l’entité identifiée par l’URI : dbr :Hintersee,_Austria

Deux prédicats sont utilisés pour décrire l’indicatif régional : Le prédicat identifié par l’URI : dbp :areaCode et le prédicat identifié par l’URI : dbo :areaCode.

Deux prédicats sont utilisés pour définir l’altitude : Le prédicat identifié par l’URI : dbo :elevation et le prédicat identifié par l’URI : dbp :elevationM.

En conséquent, cette étape consiste à identifier et grouper ces prédicats en double.

Ce regroupement des prédicats en double permet de réduire l’espace de recherche et par conséquent réduire le nombre de comparaisons à faire entre les prédicats. En outre, le nettoyage des prédicats en double permet de transformer et de nettoyer les données en entrée et de réduire le nombre de comparaisons dans le processus d’alignement de schémas.

L’algorithme 1 décrit le processus de regroupement des prédicats en double dans la description d’une entité.

Les deux premières étapes de l’algorithme permettent d’initialiser un graphe G et d’ajouter tous les prédicats existants dans la description d’une entité dbpedia au graphe

comme des nœuds. La phase principale de la partie itérative consiste en le calcul de la similarité à base de Wordnet entre les prédicats. La fomule de calcul est la suivante :

$$WordnetSim(p1, p2) = \frac{simWUP(p1, p2) + simPATH(p1, p2)}{2} \tag{3.1}$$

Avec :

-p1 et p2 sont deux prédicats qui décrivent une entité de la base de connaissances DBpedia

-simWUP(p1,p2) est la mesure Wu et Palmer entre les prédicats p1 et p2.

-simPATH(p1,p2) est la mesure Path entre les prédicats p1 et p2.

L'étape de calcul de similarité compare les labels des prédicats entre eux et permet de lier les prédicats dont la similarité dépasse une seuil. Afin de calculer la similarité à base de Wordnet, nous considérons deux mesures de similarité importantes : La mesure Wu et Palmer et la mesure Path.

Algorithm 1 : Clustering of properties with wordnet

```
Input : A set of predicats P and a threshhold S
Output : A graph G
  G ← ∅
  for p in P do
    G.add_node(p)
  end for
  for (p1,p2) in P do
    if WordnetSim(p1,p2) > S then
      G.add_Edge(p1,p2)
    end if
  end for
  return G
```

3.4.5 Alignement de schémas

Cette étape est inévitable dans le cas où différents vocabulaires sont utilisés pour décrire le même type d'information. L'alignement de schémas est utilisé pour générer un ensemble d'équivalences entre les prédicats. En effet, les prédicats sémantiquement équivalentes apparaissent dans différentes formes d'hétérogénéité de vocabulaire (Citons l'exemple de l'entité ayant pour uri "http ://www.wikidata.org/resource/Q701169", l'altitude est décrite par le prédicat ayant pour uri < https ://www.wikidata.org/wiki /Property :P2044> qui a comme label "elevation above sea level" dans wikidata alors que dans DBpedia l'altitude est définie par le prédicat identifié par l'uri < https ://dbpedia.org/ontology/elevation >.

Par conséquent, cette étape permet d'ajouter des relations entre les prédicats qui sont sémantiquement équivalentes. Deux étapes sont nécessaires pour obtenir l'alignement de schémas, la première consiste à calculer les mesures de similarité à base de Wordnet pour

comparer les labels des prédicats, la deuxième étape est utilisée pour assurer l'exactitude de cet alignement et consiste à regarder les valeurs des entités.

3.4.5.1 Alignement de schémas en se basant sur les prédicats des entités

Dans cette étape, notre système consiste à aligner les schémas entre les prédicats dans DBpedia et les prédicats dans wikidata.

L'algorithme 2 représente les étapes d'alignement de schémas en se basant sur les prédicats.

Les trois premières étapes de l'algorithme permettent de sélectionner un prédicat par cluster pour réduire le nombre des comparaisons, permettent d'extraire les listes des prédicats à aligner pour DBpedia et Wikidata. La phase principale de la partie itérative est le calcul de la similarité à base de Wordnet entre les prédicats.

La fomule de calcul est la suivante :

$$WordnetSim(p1, p2) = \frac{simWUP(p1, p2) + simPATH(p1, p2)}{2} \tag{3.2}$$

Avec :

-p1 est un prédicat qui décrit une entité de la base de connaissances Wikidata et p2 décrit une entité de la base de connaissances DBpedia.

-simWUP(p1,p2) est la mesure Wu et Palmer entre les prédicats p1 et p2.

-simPATH(p1,p2) est la mesure Path entre les prédicats p1 et p2.

L'étape de calcul de similarité compare toutes les paires de prédicats entre eux et permet de lier les prédicats dont la similarité dépasse un seuil. Afin de calculer la similarité à base de Wordnet, nous considérons deux mesures de similarité importantes : La mesure Wu et Palmer et la mesure Path.

Algorithm 2 : Wodnet-based schema alignment

```
Input : A set of predicats S1 from DBpedia, a set of predicats from wikidata S2
        and a threshhold S
Output : A dictionnary D
  D ← ∅
  G ← DeduplicationPropertiesWordnet(S1)
  WikidataPreList ← getAllPredicatLabel(S2)
  DBpediaPreList ← getPredicatwithoutDuplication(G)
  for (p1,p2) in WikidataPreList X DBpediaPreList do
    if WordnetSim(p1,p2) > S then
      D ← AssertEquicalence(p1,p2)
    end if
  end for
  return D
```

3.4.5.2 Alignement de schémas en se basant sur les valeurs

Aprés l'alignement de schémas en se basant sur les prédicats, nous cherchons à améliorer l'exactitude et la complétude de cet alignement en regardant les valeurs décrivant les entités.

La complétude signifie qu'aucun alignement n'est oublié dans le résultat d'alignement de schémas. En outre, un outil d'alignement de schémas devrait également générer autant que possible des liens entre les prédicats pour assurer la complétude. Après l'alignement de schémas, nous visons à augmenter la complétude lors de l'alignement de schémas, c'est pourquoi, nous considérons les caleurs correspondantes aux prédicats. Nous considérons seulement les prédicats de wikidata qui n'ont pas encore des relations avec des prédicats dans DBpedia.

L'algorithme 3 représente notre algorithme d'alignement de schémas en se basant sur les valeurs.

Les deux premières étapes de l'algorithme permettent de grouper les prédicats en double pour la base DBpedia en faisant appel au premier algorithme et permet de sélectionner un représentant de chaque cluster dans le graphe pour réduire le nombre des comparaisons. Les étapes suivantes permettent d'extraire les listes des prédicats à aligner pour DBpedia et Wikidata qui ne sont pas aligné après l'exécution de l'algorithme 2. La partie itérative de l'algorithme est à la dernière étape. La première étape de la partie itérative est la récupération des valeurs associés aux prédicats. La phase principale de la partie itérative est le calcul de la similarité à base de type des valeurs. Dans le cas où le type est une chaîne de caractères, nous utilisons le jaccard_similarity pour calculer la similarité entre les deux valeurs. Dans le cas des dates et des coordonnées géographiques, nous considérons la distance euclidienne pour calculer la similarité entre les deux valeurs. Dans le cas des dates et des valeurs numériques, nous considérons la fonction de similarité pour les valeurs numériques pour calculer la similarité entre les deux valeurs. L'étape de calcul de similarité compare toutes les paires des prédicats de même type entre eux et permet de lier les prédicats dont la similarité dépasse un seuil.

Algorithm 3 : Schema Alignment based on objects

```
Input : A set of predicats S1 from DBpedia, a set of predicats from wikidata
        S2, a threshhold S and a type T
Output : A dictionnary D'
  G ← DeduplicationPropertiesWordnet(S1)
  ListPredicatDBpedia ← getPredicatWithoutDuplication(G)
  ListPredicatWikiadata ← getAllPredicatLabel(S2)
  D ← SchemaAlignementPredicatBased(S1,S2,S)
  ListPredicatWikiadata2 ← ∅
  for p in ListPredicatWikiadata do
    if p not in D.keys() then
      ListPredicatWikiadata2.addProperty(p)
    end if
  end for
  for (p1,p2) in ListPredicatWikiadata2 X ListPredicatDBpedia do
    O1 ← getObject(p1)
    O2 ← getObject(p2)
    if T== 'Text' then
      Sim=Jaccard_Similarity(O1,O2)
    else if T=='Date' then
      (Day1,Month1,Year1) ← ExtraireDayMonthYear(O1)
      (Day2,Month2,Year2) ← ExtraireDayMonthYear(O2)
      Sim= (NumericalSimilarity(Day1,Day2) +
      NumericalSimilarity(Month1,Month2)+ NumericalSimilarity(Year1,Year2)) /3
    else if T =='Numerical' then
      Sim= NumericalSimilarity(O1,O2)
    else if T =='GPS' then
      Sim=1/(1+distance_euclidienne(O1,O2))
    end if
    if Sim>S then
      D'[p1]=p2
    end if
  end for
  return D'
```

À titre d'exemple, le tableau 3.6 présente l'alignement entre les prédicats utilisés pour décrire l'entité "Hintersee_Austria" ayant pour URI dans wikidata <https ://www.wikidata. org/wiki/Q774805> et dans DBpedia <https ://dbpedia.org/resource/ Hintersee,_Austria>.

3.4.6 Matching d'entités

Aprés la détermination des propriétés équivalentes dans les descriptions des deux entités, nous utilisons les alignements pour comparer deux entités de ces deux bases de connaissances DBpedia et Wikidata.

Prédicats de l'entité source	Prédicats de l'entité cible
https ://www.wikidata.org/wiki/Property :P17 Label : Country	https ://dbpedia.org/ontology/country https ://dbpedia.org/property/pushpinMap
https ://www.wikidata.org/wiki/Property :P131 Label : located in the administrative territorial entity	https ://dbpedia.org/ontology/subdivision https ://dbpedia.org/ontology/ subdivisionName
https ://www.wikidata.org/wiki/Property :P473 Label : local dialing code	https ://dbpedia.org/property/areaCode https ://dbpedia.org/ontology/areaCode
https ://www.wikidata.org/wiki/Property :P2044 Label : elevation above sea level	https ://dbpedia.org/ontology/elevation https ://dbpedia.org/property/elevationM
https ://www.wikidata.org/wiki/Property :P281 Label : postal code	https ://dbpedia.org/ontology/postalCode https ://dbpedia.org/property/postalCode

Tab. 3.6 : Résultat de l'alignement de schémas pour l'entité "Hintersee,_Austria"

Le module de matching matche les ensembles de données liées pour produire des alignements entre les entités en utilisant différentes mesures de similarité. Il présente le cœur de notre approche et définit comment les métriques de similarité sont combinées afin de calculer une valeur de similarité totale entre une paire d'entités.

Pour comparer les valeurs de prédicats, nous utilisons un nombre de métriques de similarité. Le tableau 3.7 donne un aperçu de ces métriques. Les métriques implémentées incluent chaîne, valeur numérique, date et coordonnées géographiques. Chaque métrique est évaluée à une valeur de similarité entre 0 et 1, avec des valeurs plus élevées indiquant une similarité élevée.

Métrique	Description
JaroSimilarity	String similarity based on Jaro distance metrica
qGram Similarity	String similarity based on q-grams
Jaccard Similariy	String similarity based on Jaccard distance metric
Date Similariy	Similarity between two date values
Coordinate location Similarity	Similarity between two geographical points
Numerical Similarity	Percentual numeric similarity

Tab. 3.7 : Métrique de similarité

Ces mesures de similarité sont combinées à l'aide de la fonction d'aggrégation suivante :

- AVG – moyenne pondérée

$$AVG(E1, E2) = \frac{(NumSim(E1, E2) + GeoSim(E1, E2) + DateSim(E1, E2)) * 2 + StrSim(E1, E2)}{7} \tag{3.3}$$

Avec : NumSim(E1,E2),GeoSim(E1,E2),DateSim(E1,E2),StrSim(E1,E2) sont respectivement les valeurs de similarité pour les valeurs numériques, les coordonnées géographiques, les dates et les chaînes de caractères entre les entités E1 et E2 .

Dans ce module, nous calculons les valeurs de similarité pour les valeurs numériques, les coordonnées géographiques des villes, les dates et les chaînes de caractères. Comme l'entité source et les candidats partagent plusieurs données textuelles, nous avons pondéré les valeurs de similarité pour les valeurs numériques, les coordonnées géographiques et les dates par deux et la valeur de similarité pour les chaînes de caractères par un.

Dans notre cas, nous calculons les valeurs de similarité pour chaque candidat avec l'entité source. Nous sélectionnons le candidat qui maximise la similarité avec l'entité source. Après avoir détecté le candidat avec la valeur de similarité la plus élevée, les alignements doivent être écrits dans un fichier de sortie séparé pour évaluer le travail.

3.5 Conclusion

Notre approche est une approche efficace puisqu'elle permet de réduire le nombre de comparaisons qui est effectué pendant le processus de matching. Nous avons utilisé des entités candidates pour chaque entité source. Ces candidats sont utilisés pour le matching et par conséquent réduire l'espace de recherche. De plus, nous avons utilisé le regroupement des prédicats par types des valeurs pour résoudre le type d'hétérogénéité lié aux types des valeurs. Nous avons traité le cas de Dbpedia et wikidata où différents vocabulaires sont utilisés pour décrire le même type d'information en proposant un algorithme hybride (à base des prédicats et des valeurs) pour l'alignement de schémas. Suite à ce processus d'alignement d'entités, nous poursuivons dans le dernier chapitre l'évaluation de notre approche.

Chapitre 4

Résultat et évaluation

4.1 Introduction

Dans ce chapitre, l'accent est mis sur les expérimentations et l'évaluation de notre approche d'alignement d'entités en utilisant des datasets de données liées. Particulièrement, nous allons présenter les métriques que nous avons utilisées pour notre projet. Ensuite, Nous allons analyser les résultats et les interpréter.

4.2 Environnement de travail

Dans cette section, nous détaillons les différents logiciels et les technologies utilisées pour développer notre solution d'alignement d'entités.

4.2.1 Environnement matériel

Pour la réalisation ce projet, nous avons opté à utiliser l'environnement matériel présentant les caractéristiques suivantes :

- **Marque :** Lenovo
- **Processeur :** Intel Core i5 9th génération @4.1 GHz
- **Mémoire RAM :** 8 Go
- **Disque dur :** 512 Go
- **Système d'exploitation :** Windows 10

4.2.2 Environnement logiciel

Pendant le développement de notre approche, nous avons utilisé les outils logiciels suivants :

Anaconda : Anaconda [2] est une distribution libre et open source des langages de programmation Python et R appliqué au développement d'applications dédiées à la science des données et à l'apprentissage automatique comme le traitement de données, l'analyse prédictive et le calcul scientifique. Elle vise à simplifier la gestion des paquets et de déploiement. Elle intègre une interface graphique qui permet aux utilisateurs de lancer des applications et de gérer les librairies conda, les canaux et les environnements. Elle était utilisée dans le projet comme environnement de développement pour la plupart des étapes de l'approche.

Visual Studio Code : Visual Studio Code [35] est un éditeur de code source développé par Microsoft pour Windows, Linux et macOS. Les extensions de langue peuvent être trouvées et téléchargées gratuitement à partir de VS Code Marketplace. Nous avons utilisé Visual Studio Code pour l'écriture des scripts.

Overleaf : Overleaf [8] est un éditeur gratuit, moderne et en ligne pour développer des documents en LaTex. Il inclut le support unicode, la vérification orthographique, l'auto-complétion, le pliage de code et intègre le pdf. De plus, il est facile à utiliser et à configurer, tout comme Microsoft Word office. Et, il repère les erreurs et les avertissements en les enregistrant.

4.2.3 Frameworks et bibliothèques

Matplotlib : Matplotlib [19] est une bibliothèque du langage de programmation Python destinée à tracer et visualiser des données sous formes de graphiques. Elle peut être combinée avec les bibliothèques python de calcul scientifique NumPy et SciPy.

Pandas : Pandas [24] est une bibliothèque open source, sous licence BSD, qui fournit des structures de données et des outils d'analyse de données performants et faciles à utiliser pour le langage de programmation Python. C'est la bibliothèque la plus utilisée pour analyser les données.

NumPy : NumPy [23] est une bibliothèque pour le langage de programmation Python, ajoutant la prise en charge de grands tableaux et matrices multidimensionnels, ainsi qu'une vaste collection de fonctions mathématiques et mathématiques de haut niveau pour opérer sur ces tableaux. NumPy est un logiciel open source et a de nombreux contributeurs.

NetworkX : NetworkX [22] est une bibliothèque Python pour la création, la manipulation et l'étude de la structure, de la dynamique et des fonctions de réseaux complexes. NetworkX est un logiciel libre distribué sous la nouvelle licence BSD.

Nltk : NLTK [21] est une plateforme pour la création de programmes Python destinés à travailler avec des données sur le langage humain. Elle fournit des interfaces faciles à utiliser pour plus de 50 corpus et ressources lexicales telles que WordNet, ainsi qu'une suite de bibliothèques de traitement de texte pour la classification, la tokénisation, stemming, tagging, parsing, l'analyse syntaxique et le raisonnement sémantique, des wrappers pour des bibliothèques NLP industrielles, et un forum de discussion actif.

Sckilit learn : Sklearn [18] est une bibliothèque automatique pour python intégrant

des algorithmes classiques d'apprentissage automatique. Elle vise à fournir des solutions simples et efficaces aux problèmes d'apprentissage, accessibles à tous et réutilisables dans divers contextes.

4.3 Présentation du dataset

Nous évaluons notre approche sur les deux bases de connaissances DBpedia et Wikidata. Nous considérons Wikidata comme base de connaissances source et DBpedia comme base de connaissances cible. Le tableau 4.1 montre le nombre d'entités dans chaque ensemble de données.

Nous utilisons un jeu de données qui se concentre sur les entités ambiguës dans DBpedia et Wikidata. C'est-à-dire pour chaque entité dans l'ensemble de données source, il existe plus d'une entité dans la base cible qui a exactement la même valeur pour le prédicat "rdfs :label". Pour chaque entité source de wikidata, nous avons un candidat correct et plusieurs candidats incorrectes de la base de connaissances DBpedia.

Le tableau 4.1 résume les résultats de la collecte de données pour chaque base de connaissances.

Le nombre des entités sources	**Le nombre des candidats**
3056 entités de Wikidata	20375 candidats de DBpedia

Tab. 4.1 : Dataset Statistics

Le tableau 4.2 montre la répartition des entités sources pour différents nombres des candidats. 2587 entités wikidata possède entre 1 et 10 entités candidates possibles dans DBpedia. Notre approche permet d'identifier l'entité correct parmi cet ensemble.

Nombre des candidats	**Nombre des entités sources**
entre [1,10]	2587
entre [10,20]	287
entre [20,30]	93
entre [30,40]	46
entre [40,50]	14
entre [50,100]	21
entre [100,500]	8
Total	3056

Tab. 4.2 : Répartition des entités sources pour différents nombres des candidats

Le tableau 4.3 montre la répartition des entités sources pour différentes classes.

Classe des entités	Nombre des entités
Person	872
Settlement	605
Organization	148
TV_Program	112
Album	338
MusicalWork	200
Work	371
Animal	16
Others	394
Total	3056

Tab. 4.3 : Répartition des entités sources sur différentes classes

4.4 Métriques d'évaluation

Nous expliquons ici nos métriques pour évaluer notre travail. Nous avons examiné différentes métriques pour les algorithmes de classement.

- **Précision sur la première position :** La métrique la plus simple est la précision, elle est définie par le ratio entre le nombre de réponses correctes et le nombre total de reponses retournées. Dans le cas où le système peut donner plusieurs réponses par question on ne considère que la première position. En notant CRi le rang de la première réponse correcte pour la question i. Cette mesure donne directement la probabilité que le système soit correct quand il donne une réponse en considérant seulement la première position.

$$Precision = \frac{\#CRi = 1}{\#reponses} \tag{4.1}$$

- **Top-n accuracy (Précision sur les n premières positions) :** Dans de nombreux cas où nous utilisons des algorithmes de classement (par exemple, recherche Google, recommandation de produits Amazon), nous obtenons des centaines et des milliers de résultats. Il est intéressant de connaître la qualité de réponses correctes parmi les n premières réponses retournées et ne pas se limiter seulement à la première position.

$$Top\_nAccuracy = \frac{\#CRi \leq n}{\#reponses} \tag{4.2}$$

- **Mean Reciprocal Rank (MRR) :** Pour mesurer la qualité de ce classement, le Mean Reciprocal Rank est utilisé. Le rang de réponse correcte dans le classement est pondéré par l'inverse du rang. Cette mesure essaie de mesurer « Où est le premier élément pertinent ? ». Elle est étroitement liée à la famille de métriques de pertinence binaire.

Cette mesure est définie comme suit :

$$MRR = \frac{\sum \frac{1}{CRi}}{\#reponses} \tag{4.3}$$

Tel qu'illustré par la figure 4.1, supposons que nous ayons les trois listes de candidats suivantes pour trois requêtes. Nous pouvons calculer le rang réciproque de chaque requête en trouvant le rang du premier élément pertinent, par liste. Ensuite, nous faisons une moyenne simple sur tous les requêtes.

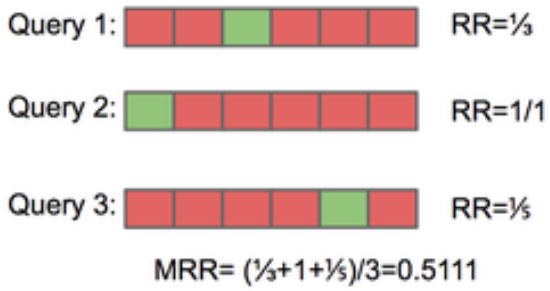

FIG. 4.1 : Exemple de calcul de la métrique MRR

4.5 Résultats et interprétations

Dans cette section, nous allons mettre l'accent sur les expérimentations menées pour évaluer notre approche d'alignement d'entités. Particulièrement, nous évaluons le MRR et la précision sur la première position et sur les trois premières positions de notre algorithme.

4.5.1 Résultat de l'alignement d'entités

Nous avons utilisé la mesure MRR, ainsi que la proportion des entités ayant une bonne réponse en première position ou dans les trois premières positions pour évaluer les résultats. Sur l'ensemble de données, notre approche atteint 0.85 MRR lors de l'alignement des entités Wikidata aux entités de DBpedia.

	Top1 Accuracy	**Top3 Accuracy**	**Mean Reciprocal Rank**
Wikidata-DBpedia	0.77	0.93	0.85

TAB. 4.4 : Performance de l'approche d'alignement d'entités

4.5.2 Effets du nombre des candidats

Nous étudions les effets d'augmentation du nombre des candidats sur l'efficacité. Nous mesurons le MRR sur l'ensemble de données pour différents nombres de candidats. Comme le montre la figure 4.2, le MRR diminue pour les requêtes qui ont un grand nombre de candidats. Ainsi, on peut voir que les meilleures valeurs de MRR sont atteintes pour le nombre des candidats moyen [50,100].

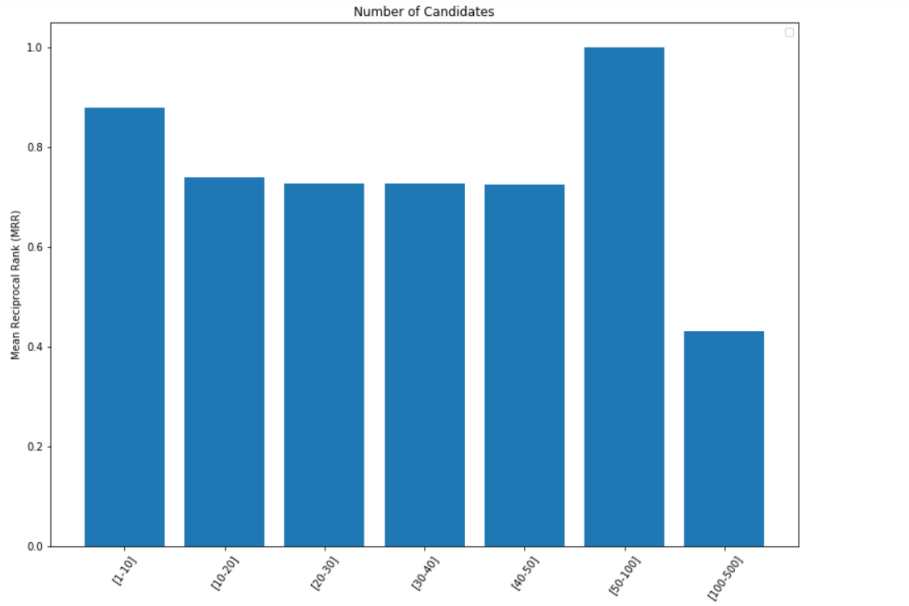

FIG. 4.2 : Effets d'augmentation du nombre des candidats

4.5.3 Effets de la classe des entités sur les performances de l'approche

La figure 4.3 illustre les performances de l'approche selon le type|classe des entités.

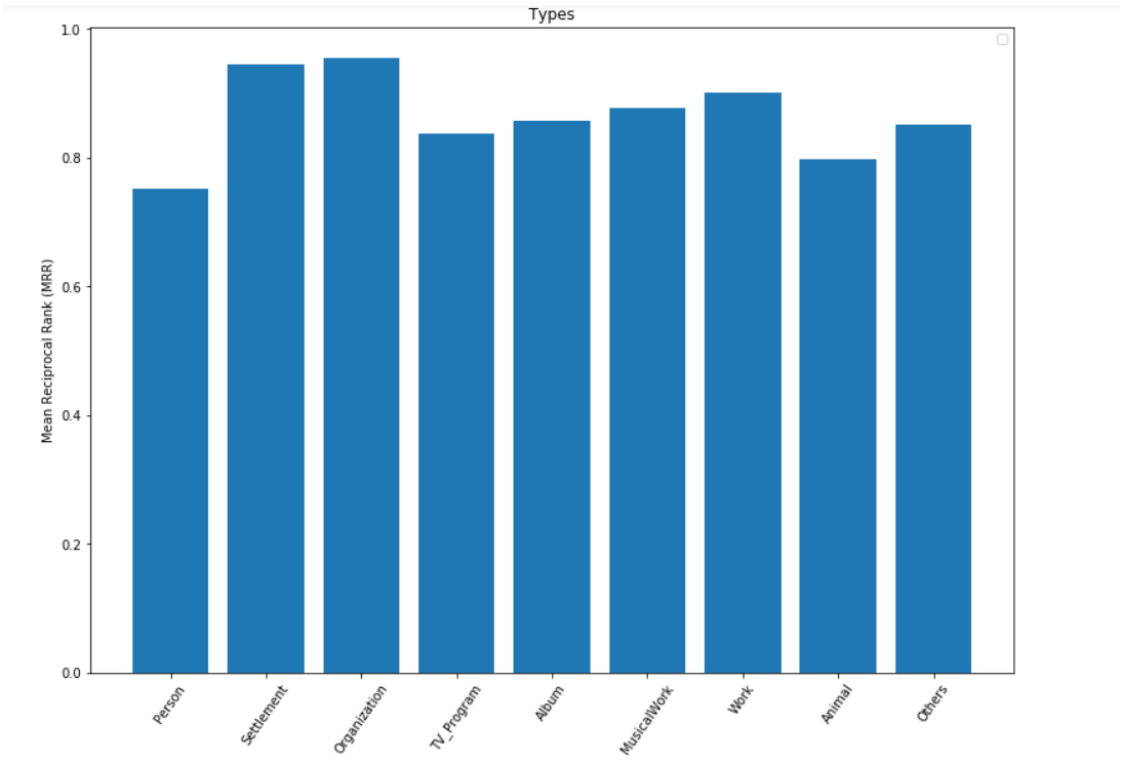

FIG. 4.3 : Effet de classes des entités sur le MRR

Le tableau 4.5 illustre les performances (MRR et Top1 Accuracy) de l'approche pour chaque classe.

Classe	Top1 Accuracy	Mean Reciprocal Rank	Nombre des entités
Person	0.63	0.75	872
Settlement	0.9	0.94	605
Organization	0.92	0.95	148
TV_Program	0.76	0.83	112
Album	0.75	0.85	338
MusicalWork	0.81	0.87	200
Work	0.82	0.9	371
Animal	0.62	0.79	16
Others	0.76	0.85	394
Total			3056

TAB. 4.5 : Effet de la classe sur le MRR et sur le Top1 Accuracy

En se basant sur l'analyse des erreurs, nous observons que notre système n'a pas la capacité de lever l'ambiguïté des entités dans le même type/domaine dans certains cas.

Par exemple, dans le domaine de la musique, étant donné une entité "Q7765805" de Wikidata, qui est un album musical de Alphabeat, le système a classé le single "The_Spell_(song)", en première position, et a classé l'entité correcte "The_Spell_(Alphabeat_album)", un album d'Alphabeat, en deuxième position.

Globalement, pour les cas où le système ne parvient pas à placer les entités correctes

au premier rang mais réussit au deuxième rang au lieu de cela, ces entités sont des entités de même type/domaine dans ces deux positions.

Notre système atteint des valeurs proches de 1 pour la classe Settlement et Organisation. Alors que les performances diminuent pour les deux classes Person et Animal. Cette diminution est dûe aux entités décrites d'une manière complémentaires et ont peu d'informations en commun. En outre, notre système n'arrive pas à résoudre le problème de la complémentarité des informations entre les ensembles de données. Donc, il classe incorrectement les candidats dans le cas où l'entité source et l'entité correcte ont peu d'informations en commun. Par conséquent, l'alignement du schémas donne résultat à un ensemble vide d'alignement entre les prédicats et donc de faibles valeurs de similarité entre l'entité source et l'entité correcte faibles.

4.6 Conclusion

Notre approche permet de réduire le nombre de comparaisons qui est effectué pendant le processus d'alignement d'entités. Pour cela, nous avons utilisé des entités candidats pour éviter la complexité quadratique. En outre, notre approche est une approche performante pour l'alignement des entités provenant de grands ensembles de données du LOD Cloud. Nous avons évalué notre approche en utilisant des données réelles et en calculant les valeurs de Mean reciprocal rank et de top-n accuracy pour montrer que l'approche est efficace parce qu'elle utilise des mesures de similarité sémantiques et syntaxiques.

Conclusion générale et perspectives

Dans ce travail, nous proposons, implémentons et évaluons une approche d'alignement d'entités dans les graphes RDF dans le contexte du web des données liées. Cette approche permet d'inclure des liens d'identité entre les différentes entités qui existent dans plusieurs ensembles de données liées et décrivent le même objet du monde réel. Cette approche permet d'identifier le principe des données liées afin de fournir une vue unifiée des ensembles de données liées pour que les applications puissent interroger ces données efficacement.

Nous fournissons un analyse approfondie des hétérogénéités au sein des datasets RDF, qui sont à l'origine du problème d'alignement d'entités.

Nous proposons une comparaison entre les approches d'alignement d'entités. Cette comparaison est faite en utilisant plusieurs critères. Le premier critère étudie les différentes mesures de similarité ainsi que les sources de connaissances qui sont utilisées dans le processus de découverte des liens d'identités. Le deuxième critère permet de comparer le positionnement de ces approches par rapport aux types des hétérogénéités. Ces approches utilisent des techniques de blocking et des méthodes d'indexation pour réduire la complexité temporelle du processus d'alignement d'entités. Toutes ces approches utilisent des fonctions de similarité à base de chaînes de caractères telles que Levenshtein, Jaccard, Cosine-similarity, etc.

Nous montrons que notre approche résout efficacement différents types d'hétérogénéités en adoptant une approche hybride qui combine le schéma et les valeurs associées. Nous avons utilisé différentes mesures de similarités pour calculer la similarité entre une paire d'entités. En outre, notre approche vise à utiliser le Wordnet et des mesures de similarités à base de chaînes, les mesures numériques, de date, ou des coordonnées géographiques pour résoudre l'hétérogénéité liée aux types des valeurs.

Afin de minimiser le temps d'exécution et réduire la complexité temporelle, notre approche améliore la complexité temporelle et réduit le nombre de comparaisons par la génération des entités candidats à l'entité source pour éviter la complexité quadratique O(N) et en traitant ces groupes séparément.

L'approche a été évaluée en utilisant des entités à partir des ensembles de données liées réels. Nous avons obtenu des résultats satisfaisants et encourageants grâce à l'utilisation du Wordnet et différentes mesures de similarités pour calculer la similarité.

Les futurs travaux pour l'amélioration de notre approche d'alignement d'entités comportent les tâches suivantes :

Fusion des données liées. Après la tâche d'alignement d'entités nous devons proposer une nouvelle approche qui vise à combiner, agréger et résoudre les valeurs conflictuelles provenant de différents ensembles de données pour obtenir une vue unifiée de ces données.

Viser d'autres types des liens. Afin d'améliorer et faciliter l'interrogation des données liées, nous devrons viser d'autres types des liens tels que l'inclusion, l'héritage et la méronymie.

Résoudre le problème de la complémentarité des informations entre les ensembles de données. Afin d'améliorer le résultat de notre approche, nous devrons gérer les entités décrites par des ensembles complémentaires de propriétés dans deux bases de connaissances différentes et ont donc peu d'informations en commun afin de les comparer.

Bibliographie

[1] Manel Achichi et al. “Linking and disambiguating entities across heterogeneous RDF graphs”. In : *Journal of Web Semantics* 55 (2019), p. 108-121.

[2] *Anaconda Documentation*. https://docs.anaconda.com/. [En ligne ; Septembre 2021].

[3] David Beckett. *RDF 1.1 XML Syntax*. en. Fév. 2014. url : https://www.w3.org/TR/rdf-syntax-grammar/.

[4] Khayra BENCHERIF et al. “Enrichissement et intégration des données liées”. Thèse de doct. 2017.

[5] Ramanathan V Guha Dan Brickley. *RDF Schema 1.1*. en. Fév. 2014. url : https://www.w3.org/TR/rdf-schema/.

[6] Tim Berners-Lee David Beckett. *Turtle - Terse RDF Triple Language*. en. Mar. 2011. url : http://www.w3.org/TeamSubmission/turtle/.

[7] AnHai Doan, Alon Halevy et Zachary Ives. *Principles of data integration*. Elsevier, 2012.

[8] *Documentation overleaf*. https://fr.overleaf.com/learn. [En ligne ; Septembre 2021].

[9] Michael Farag. “Entity Matching and Disambiguation Across Multiple Knowledge Graphs”. Mém. de mast. University of Waterloo, 2019.

[10] Fabien Gandon et al. “The semantic web : latest advances and new domains”. In : *ESWC 2015-European Semantic Web Conference*. T. 9088. Springer. 2015, p. 830.

[11] Jeremy J. Carroll Graham Klyne. *Resource Description Framework (RDF) :Concepts and Abstract Syntax*. en. Fév. 2004. url : https://www.w3.org/TR/rdf-concepts/.

[12] W3C OWL Working Group. *Owl 2 web ontology language document overview*. en. Déc. 2012. url : https://www.w3.org/TR/owl2-overview/.

[13] Fuzhen He et al. “Unsupervised entity alignment using attribute triples and relation triples”. In : *International Conference on Database Systems for Advanced Applications*. Springer. 2019, p. 367-382.

[14] Tom Heath et Christian Bizer. “Linked data : Evolving the web into a global data space”. In : *Synthesis lectures on the semantic web : theory and technology* 1.1 (2011), p. 1-136.

[15] *Le web sémantique, futures approches et visions du web pensant*. https://maveille.fr/comprendre-et-definition-web-semantique-et-visions-du-web-intelligent/. [En ligne ; Septembre 2021].

[16] Claudia LEACOCK, Martin CHODOROW et George A MILLER. "Using corpus statistics and WordNet relations for sense identification". In : *Computational Linguistics* 24.1 (1998), p. 147-165.

[17] *LIUPPA*. https://liuppa.univ-pau.fr/fr/index.html. [Publié le 4 juin 2020].

[18] *Machine learning module for Python*. https://www.kite.com/python/docs/sklearn. [En ligne ; Septembre 2021].

[19] *Matplotlib Documentation*. matplotlib.org/stable/contents.html. [En ligne ; Août 2021].

[20] George A MILLER. "WordNet : a lexical database for English". In : *Communications of the ACM* 38.11 (1995), p. 39-41.

[21] *Natural Language Toolkit*. https://www.nltk.org/. [En ligne ; Septembre 2021].

[22] *NetworkX Documentation*. [Online ; Juillet 2021]. URL : %5Curl%7Bhttps://networkx.org/documentation/stable/tutorial.htm%7D.

[23] *NumPy Documentation*. numpy.org/. [En ligne ; Septembre 2021].

[24] *Pandas Documentation*. https://pandas.pydata.org/docs/. [En ligne ; juillet 2021].

[25] Georgios PAPADAKIS. "Blocking Techniques for efficient Entity Resolution over large, highly heterogeneous Information Spaces". Thèse de doct. Hannover : Gottfried Wilhelm Leibniz Universität Hannover, 2013.

[26] Julien PLU. *Bien choisir ou créer un vocabulaire*. en. Fév. 2003. URL : https://jplu.developpez.com/tutoriels/web-semantique/choisir-son-vocabulaire/.

[27] Camille PRADEL. "D'un langage de haut niveau à des requêtes graphes permettant d'interroger le web sémantique". Thèse de doct. Université de Toulouse, Université Toulouse III-Paul Sabatier, 2013.

[28] François SCHARFFE, Yanbin LIU et Chuguang ZHOU. "Rdf-ai : an architecture for rdf datasets matching, fusion and interlink". In : *Proc. IJCAI 2009 workshop on Identity, reference, and knowledge representation (IR-KR), Pasadena (CA US)*. 2009, p. 23.

[29] *Syntaxe des expressions régulières*. http://blog.paumard.org/cours/java-api/chap03-expression-regulieres-syntaxe.html. [En ligne ; Septembre 2021].

[30] *The Linked Open Data Cloud*. https://lod-cloud.net/. [En ligne ; Septembre 2021].

[31] *The Linked Open Data Cloud en 2007*. https://lod-cloud.net/versions/2007-05-01/lod-cloud.png. [En ligne ; Septembre 2021].

[32] *The Linked Open Data Cloud en 2009*. https://lod-cloud.net/versions/2009-03-05/lod-cloud.png. [En ligne ; Septembre 2021].

[33] *The Linked Open Data Cloud en 2011*. https://lod-cloud.net/versions/2011-09-19/lod-cloud.png. [En ligne ; Septembre 2021].

[34] Esko Ukkonen. “Approximate string-matching with q-grams and maximal matches”. In : *Theoretical computer science* 92.1 (1992), p. 191-211.

[35] *visualstudio Documentation*. https://code.visualstudio.com/docs. [En ligne ; Août 2021].

[36] Julius Volz et al. “Silk-a link discovery framework for the web of data”. In : *Ldow*. 2009.

[37] Zhibiao Wu et Martha Palmer. “Verb semantics and lexical selection”. In : *arXiv preprint cmp-lg/9406033* (1994).