



République Algérienne Démocratique et Populaire
Ministère de l'enseignement supérieur et de la
recherche scientifique

Université Larbi Tébessi - Tébessa

Faculté des Sciences Exactes et des Sciences de la Nature et de la Vie

Département : Mathématiques et Informatique



كلية العلوم المحيطة وعلوم الطبيعة و الحياة
FSES NV
E=MC
FACULTÉ DES SCIENCES EXACTES
ET DES SCIENCES DE LA NATURE ET DE LA VIE

Mémoire de fin d'étude
Pour l'obtention du diplôme de MASTER
Domaine : Mathématiques et Informatique
Filière : Informatique
Option : Systèmes d'information

Thème

Extraction des graphes de connaissances depuis des données non structurées

Présenté Par :

HELALI

Maroua

Devant le jury :

Mr. Bouroguaa. S	MCB	Université Larbi Tébessi	Président
Mr. Zagari. A	MAA	Université Larbi Tébessi	Examineur
Mr. Djeddai. A	MCB	Université Larbi Tébessi	Encadreur

Date de soutenance : 14 septembre 2020

Résumé

Le graphe de connaissances (GC) joue un rôle crucial dans de nombreuses applications modernes. En effet, l'extraction automatisée d'informations à partir du texte et sa transformation en une description formelle est un objectif important à la fois dans la recherche sur le Web sémantique et la linguistique informatique. Les informations extraites peuvent être utilisées pour une variété de tâches telles que la génération du graphe de connaissance. DBpedia extrait des informations structurées de Wikipédia, les relie à d'autres bases de connaissances et publie librement les résultats sur le Web à l'aide de Linked Data et de SPARQL. Dans ce contexte, nous présentons dans ce mémoire une étude pour l'enrichissement des résultats de l'extracteur de Dbpedia. Dans ce travail, nous avons concentré sur l'enrichissement de l'extraction des connaissances depuis les infobox de Wikipédia. Dans ce cadre, nous avons présenté principalement une contribution pour l'utilisation de la mesure de similarité pour l'enrichissement des résultats de l'extracteur de Dbpedia. L'approche proposée comporte deux étapes: le prétraitement des triples RDF et ensuite le calcul de similarité. L'approche est implémentée en utilisant Eclipse Java et plusieurs API comme JENA pour manipuler les graphes RDF.

Les Mots clés : Graphe de connaissances, Extraction des connaissances, Extracteur de DBpedia, Similarité, WordNet.

Abstract

The knowledge graph (KG) plays a crucial role in many modern applications. Indeed, the automated extraction of information from text and its transformation into a formal description is an important goal in both Semantic Web research and computational linguistics. The extracted information can be used for a variety of tasks such as generating the knowledge graph. DBpedia extracts structured information from Wikipedia, links it to other knowledge bases, and freely publishes the results to the web using Linked Data and SPARQL. In this context, we present in this thesis a study for the enrichment of the results of the Dbpedia extractor. In this work, we focused on enriching knowledge extraction from Wikipedia infoboxes. In this context, we mainly presented a contribution for the use of the similarity measure for the enrichment of the results of the Dbpedia extractor. The proposed approach involves two steps: preprocessing the RDF triples and then calculating it for similarity. The approach is implemented using Eclipse Java and several APIs like JENA to manipulate RDF graphs.

Keywords: *Knowledge graph, Knowledge extraction, DBpedia extractor, Similarity, WordNet.*

ملخص

يلعب الرسم البياني المعرفي (KM) دورًا مهمًا في العديد من التطبيقات الحديثة. في الواقع ، يعد الاستخراج الآلي للمعلومات من النص وتحويلها إلى وصف رسمي هدفًا مهمًا في كل من أبحاث الويب الدلالي واللغويات الحاسوبية. يمكن استخدام المعلومات المستخرجة في مجموعة متنوعة من المهام مثل إنشاء الرسم البياني للمعرفة. تستخرج *DBpedia* المعلومات المنظمة من ويكيبيديا ، وتربطها بقواعد المعرفة الأخرى ، وتنشر النتائج بحرية على الويب باستخدام البيانات المرتبطة و *SPARQL*. في هذا السياق نقدم في هذه الرسالة دراسة لإثراء نتائج مستخرج *Dbpedia*. في هذا العمل ، ركزنا على إثراء استخراج المعرفة من صناديق معلومات ويكيبيديا. في هذا السياق ، قدمنا بشكل أساسي مساهمة لاستخدام مقياس التشابه لإثراء نتائج مستخرج *Dbpedia*. يتضمن النهج المقترح خطوتين: المعالجة المسبقة لثلاث مرات *RDF* ثم حساب التشابه. يتم تنفيذ هذا النهج باستخدام *Eclipse Java* والعديد من واجهات برمجة التطبيقات مثل *JENA* لمعالجة الرسوم البيانية *RDF*.

الكلمات المفتاحية الرسم البياني للمعرفة ، استخراج المعرفة ، مستخرج *DBpedia* ، التشابه ، *WordNet*.

Remerciements

Tout d'abord, longue vie à « *Allah* » qui nous a guidé sur le droit chemin tout au long du travail et nous a inspiré les bons pas et les justes réflexes. Sans sa miséricorde, ce travail .n'aurait pas abouti

L'encadrement scientifique de ce travail a été assuré par **Dr. Djedjai. A**, docteur à la faculté des Sciences Exactes et des Sciences de la Nature et de la Vie, Université Tébessa. Nous tenons vivement à lui exprimer nos profondes reconnaissances gratitude pour sa disponibilité, sa patience, sa compréhension, ses qualités humaines et ses intérêts portés pour notre sujet de travail. Nous le remercions de nous avoir fait confiance et d'avoir été présent aussi souvent que possible malgré ses tâches pédagogiques. Son soutien permanent et son dynamisme nous .ont permis d'avancer plus loin dans notre travail

Nos remerciements vont aussi à

Dr. Bourougaa. S, d'avoir ménagé son temps pour présider ce jury

Dr. Zagari. A, pour avoir bien voulu siéger dans ce jury afin d'examiner et critiquer ce .mémoire et nous éclairer par ces précieux conseils

Dr. Djedjai. A

Aucun remerciement ne saurait exprimer notre respect et considération pour les orientations .que vous avez consenties pour notre étude de l'Université

Tous nos amis pour leur solidarité

A nos parents

.Pour l'enfance merveilleuse qu'ils nous ont offerte ainsi que pour leurs encouragements

.Pour leurs soutiens et leurs aides

.Avec tout notre amour

A tous mes amis

.Pour leurs bonnes humeurs, leurs gentillesse et pour tous nos fous rires partagés

Pour tout ce qu'ils nous ont appris

Merci à tous ceux qui, d'une manière ou d'une autre qui ont contribué à la réalisation de ce .travail, et que nous ne pouvons citer individuellement

DEDICACE

Je dédie ce travail

A mes parents.

A ma sœurs : Narimen.

A mes frères : Takí Eddine, Ayoubé.

A toutes mes amies : Sarah, Chahra, Yousra, Dalal,

Amel, Hind, Bouchra, Hala.

A ma tante : Chaouia.

A mes oncles : Amor Messadia, Abd Eldjabar, Mourad.

A tous ceux qui m'ont aidé dans mes études.

A tous mes proches, et tous ceux qui m'aiment.

Maroua

Table des matières

Introduction générale	01
Chapitre 1. L'extraction des graphes de connaissances	
1 Introduction	20
2 Le graphe de connaissances	20
2.1 RDF	21
2.2 RDFs	22
2.3 Langage d'Ontologie web(OWL)	23
2.4 Ontologie	24
2.5 L'extraction des connaissances	25
2.6 Les types des graphes de connaissances	27
2.7 Graphes des connaissances d'entreprises	28
2.8 Les étapes d'extraction des connaissances	29
2.8.1 Mappage d'entité	29
2.8.2 Résolution de référence	30
2.8.3 Extraction des triples	30
2.8.4 Triple intégration	30
3 Conclusion du chapitre	31
Chapitre 2. Les travaux connexes	
1 Introduction	33
2 Les travaux connexes	33
2.1 L'approche [Morsey et al., 2012a]	33
2.1.1 Les points forts	33
2.1.2 Les points faibles	34
2.2 LODifier [Augenstein et al., 2012]	34
2.2.1 Les points forts	34
2.2.2 Les points faibles	34
2.3 Entity Extraction [Exner & Nugues, 2012]	35
2.3.1 Les points forts	35
2.3.2 Les points faibles	35
2.4 Le système T2KG	35
2.4.1 Les points forts	36
2.4.2 Les points faibles	36
5 Conclusion	36
Chapitre 3. L'approche proposée	

1	Introduction	39
2	Vue d'ensemble sur DBpedia	39
	2.1 Extraction d'Infobox générique ou basée sur la cartographie	39
	2.1.1 Extraction d'Infobox générique	39
	2.1.2 Extraction d'infobox basée sur la cartographie	40
	2.2 Extracteurs DBpedia	40
3	Architecture générale du système	43
4	Le prétraitement de résultat de l'extraction Framework DBpedia	44
	4.1 Le prétraitement	44
	4.1.1 Normalisation des triples RDF	44
5	Une nouvelle mesure de similarité	46
	5.1 Présentation du WordNet	46
	5.2 Un mesure de similarité basé sur les sens des concepts	46
	5.2 Etendre la mesure de similarité	48
6	Conclusion	51
Chapitre 4. Implémentation et Evaluation		
1	Introduction	53
2	Implémentation	53
3	Langage de développement	53
	3.1 Le langage Java	53
4	Les outils de développement	54
	4.1 Eclipse	54
	4.2 Jena	55
5	Les étapes d'exportation d'un page Wikipédia vers fichier XML	53
6	L'organigramme	56
7	L'évaluation	56
	7.1 L'évaluation de la mesure de similarité proposée	56
	7.2 L'évaluation d'enrichissement	57
9	Conclusion	61
Conclusion General		
1	Conclusion	63
2	Travaux future	63
Références bibliographiques		64

Liste de Figures

Figure 1.1	Exemple de RDF	22
Figure 1.2	Exemple de RDFs	23
Figure 1.3	Exemple d'OWL	24
Figure 1.4	Exemple De Format n-triples	25
Figure 1.5	Extrait de l'ontologie DBpedia	25
Figure 1.6	L'architecture de graphe de connaissances.	28
Figure 1.7	Exemple de graphe de connaissances d'entreprise.	29
Figure 3.8	Syntaxe de l'info-box MediaWiki pour l'Algarve (à gauche) et rendu de l'info-box (à droite).	41
Figure 3.9	L'architecture générale du système.	42
Figure 4.10	Fenêtre principal de l'éclipse	55
Figure 4.11	L'organigramme	56
Figure 4.12	Exemple de résultat d'extraction de l'extracteur DBpedia	58
Figure 4.13	La classe Java qui fait le prétraitement	58
Figure 4.14	La base de textes	59
Figure 4.15	La classe Java pour la mesure de similarité	59
Figure 4.16	La classe Java principale pour l'enrichissement	60
Figure 4.17	L'exécution de l'enrichissement	60
Figure 4.18	Le fichier RDF qui comporte le résultat de l'enrichissement	61

Liste de Tableaux

Tableau 4.1	Des valeurs de similarité calculées avec <i>sim1</i> , <i>sim2</i> , <i>sim_senses</i> et <i>Levenshtein</i> .	57
--------------------	--	-----------

Introduction Générale

Introduction Générale

Un graphe de connaissances (KG) est une base de connaissances structurée en graphes qui stocke les connaissances sous la forme de la relation entre les entités. Le KG joue un rôle important dans diverses applications, par exemple la réponse aux questions, la navigation dans les connaissances et la visualisation des données. Les données structurées désignent les données représentées selon un schéma défini, comme des tables, des taxonomies, des dictionnaires ou des ontologies. En général, les données structurées sont facilement interprétées par les machines informatiques et représentent une étape importante vers l'interopérabilité des données.

Entre autres, Les applications Web sémantiques s'appuient sur RDF et OWL pour représenter les connaissances utilisées pour décrire des contextes spécifiques, ces langages formels sont compréhensibles par la machine. En effet, ils permettent un espace où les machines informatiques sont les citoyens de première classe. Les humains, au contraire, utilisent un mode de communication déferent et son fondement repose sur l'utilisation du langage naturel (NL) pour exprimer la sémantique. Cet écart entre les deux langues déferentes a été comblé par les approches d'extraction des informations (IE) développées par la communauté de recherche Traitement du langage naturel (NLP).

Objectifs

Notre contribution dans ce manuscrit consiste à l'enrichissement des résultats de l'extracteur de Dbpedia et principalement l'extraction des connaissances depuis les Info-Boxes. Nous avons remarqué que les littéraux dans les triples RDF peuvent être enrichis par d'autres similaires. Dans ce contexte, notre objectif principal est l'amélioration des résultats de l'extraction de Dbpedia en utilisant une mesure de similarité proposée basée sur WordNet. On a utilisé aussi d'autre mesure syntaxiques dans le cas si le calcul de similarité est purement syntaxique. Nous avons aussi présenté une étape de prétraitement des fichiers RDF afin d'améliorer le calcul de similarité. Cette étape consiste à éliminer les caractères spéciaux et extraire les sous mots depuis les mots complexes.

Plan du mémoire

Ce mémoire comprend quatre chapitres présentés comme suit :

Le premier chapitre décrit les définitions des concepts clés de notre sujet et la présentation des relations entre eux.

Introduction Générale

Dans le deuxième chapitre, nous avons cité des travaux connexes en indiquant ses limites et ses avantages, et nous récapitulons par une catégorisation et une comparaison entre les différentes propositions.

Pour le troisième chapitre, nous abordons notre démarche proposée afin de répondre à la problématique du sujet qui est l'enrichissement des résultats de l'extracteur de Dbpedia.

Le dernier chapitre inclut des présentations sur les outils et les environnements utilisés pour le développement, et le déploiement de notre proposition en présentant les fonctionnalités de notre application munie par des interfaces.

Chapitre 1

L'extraction des graphes de connaissances

1. Introduction

Graphe de connaissance (GC) joue un rôle crucial dans de nombreuses applications modernes. Néanmoins, la construction de GC à partir de texte non structuré est un problème difficile en raison de sa nature. Par conséquent, de nombreuses approches proposent de transformer un texte non structuré en texte structuré afin de créer un GC.

Dans ce contexte, nous présentons dans ce chapitre le graphe de connaissance et comment créer le.

2. Le graphe de connaissances (GC)

Un graphe de connaissances (GC) est une représentation graphique-théorique des connaissances humaines telles qu'elles peuvent être ingérées avec de la sémantique par une machine. En d'autres termes, c'est un moyen d'exprimer des " connaissances " à l'aide de graphiques, de manière à ce qu'une machine soit en mesure de raisonner et d'inférer sur ce graphique pour répondre de manière significative aux requêtes (" questions "). Cependant, cette définition n'est pas très opérationnelle. La définition fonctionnelle la plus simple d'un graphe de connaissances est qu'il s'agit d'un ensemble de triplets, chaque triple représentant intuitivement une «assertion». Si le GC a été construit correctement (avec une précision de 100%) sur une source de données fiable, nous pourrions également considérer les assertions comme des faits. Formellement, un triple est un triple (h, r, t) où h représente une entité tête, t représente une entité queue et r exprime une relation entre les deux entités. De nombreuses déclarations, mais pas toutes, en langage naturel (par exemple, l'anglais) peuvent être exprimées de manière pratique sous cette forme. Considérons, par exemple, la phrase Fido le chien a volé un os dans la cour de Mary, qui peut être exprimée comme un ensemble de triplets {(Fido, est-un, Chien), (Fido, a volé, os_1), (os_1, est-un, Os), (os_1, localisé, cour_1), (cour_1, is-a, Cour), (cour_1, appartient-à, Mary), (Mary, est-un, Personne)}. Pourquoi est-il judicieux d'appeler un tel ensemble un «graphe»? Pendant longtemps, en fait, il n'était pas conventionnel de le faire et ce que nous appelons ici un graphe de connaissances était connu (et est toujours connu, dans de nombreux articles) comme une base de connaissances. L'une des principales raisons pour lesquelles les bases de connaissances se transforment lentement en graphiques de connaissances peut être attribuée à l'influence et au succès du Graphe de Connaissance Google. Cependant, il y avait également une influence omniprésente à la fois de

la découverte des connaissances et du Web sémantique, les communautés, qui ont toujours été étroitement associées aux innovations de la théorie des graphes. Pendant une grande partie de ce millénaire, la communauté des bases de données a également étudié en détail les bases de données graphiques, les algorithmes et les structures de données.[Kejriwal, 2019]

1. RDF

RDF perçoit le monde comme un ensemble de ressources. Une ressource peut être n'importe quoi (une page Web, un fragment de page Web, une personne, un objet, etc.) et est référencée dans le Web sémantique avec un identificateur de ressource uniforme (URI). RDF est construit sur 3 concepts: ressources, propriétés (relations) et instructions. Comme mentionné précédemment, une ressource peut être tout ce qui est référencé par un URI. Une propriété ou une relation est une ressource qui donne des informations sur un aspect d'une ressource. Puisqu'une propriété est une ressource, toutes les propriétés ont un URI unique. Une déclaration est un triple de la forme <Sujet, Propriété, Objet> qui met une Ressource (le sujet de la déclaration) en relation avec une autre Ressource (l'objet de la déclaration). La propriété (relation) d'une déclaration indique les aspects sur lesquels la déclaration donne des informations. Un ensemble d'instructions forme un graphe RDF. RDF définit un petit ensemble de ressources standard. La plus importante est la propriété type (relation) qui exprime une relation "est une". Un triple qui utilise la relation de type telle que <Sujet, type, Objet> indique généralement que le sujet du triplet est une conceptualisation et l'objet est une instance de la conceptualisation. La figure 1.1 est un graphe RDF simple qui décrit certaines conceptualisations matérielles courantes et des instances de ces conceptualisations qui se rapportent au processeur de réseau Intel IXP45X. Chaque ovale représente une ressource RDF qui représente une conceptualisation. Les flèches représentent les ressources de la propriété. Toutes les ressources sont identifiées par un URI (le préfixe URI est utilisé pour la concision). Le modèle indique qu'un IXP45X est un processeur réseau (un type spécial de processeur) fabriqué par un fabricant spécifique appelé Intel et prenant en charge un bus de communication spécifique appelé PCI v2.2. La spécification RDF définit un format de sérialisation standard appelé RDF / XML pour sa syntaxe abstraite.[Aboulhamid & Rousseau, 2018]

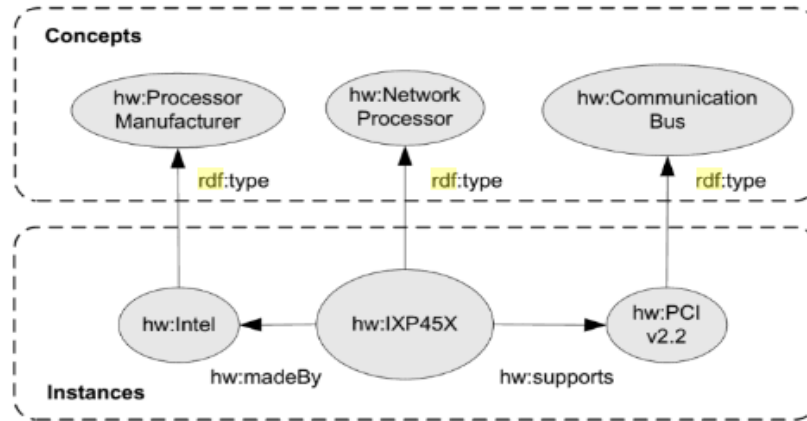


FIGURE 1.1 : EXEMPLE DE RDF. [ABOULHAMID & ROUSSEAU, 2018]

2. RDFs

Le schéma RDF (RDFS) étend RDF sur 2 axes: il définit une liste précise de ressources (sens et URIS) et un ensemble de règles d'implication qui permettent d'inférer de nouveaux triplets à partir de graphes RDFS. RDFS permet de définir des classes de ressources ainsi que leur organisation. Le concept d'une classe dans RDFS doit être compris comme un ensemble. La figure 1.2 est une extension de la figure 1.1 qui définit les relations entre les conceptualisations. Par exemple, la figure indique que l'ensemble de tous les processeurs de réseau est un sous-ensemble des processeurs définis. Nous nous référons souvent aux réseaux de conceptualisations comme schéma d'où le nom de schéma RDF. Sur la base de la sémantique de RDFS et des règles d'implication sous-jacentes, 2 triplets implicites doivent être compris: $\langle \text{hw: Intel}, \text{rdf: type}, \text{hw: Manufacturer} \rangle$ et $\langle \text{hw: IXP45X}, \text{rdf: type}, \text{hw: Processor} \rangle$. Les triplets implicites précédents peuvent être déduits en raison de la sémantique de la propriété `rdfs: subClassOf`. Puisque le `rdfs: subClassOf` exprime la relation de l'ensemble parent et du sous-ensemble entre 2 ensembles, tous les individus du sous-ensemble sont des individus nécessaires de l'ensemble parent. Puisque RDFS et OWL sont des langages définis avec RDF, les deux peuvent être sérialisés en RDF / XML ou N3.[Aboulhamid & Rousseau, 2018]

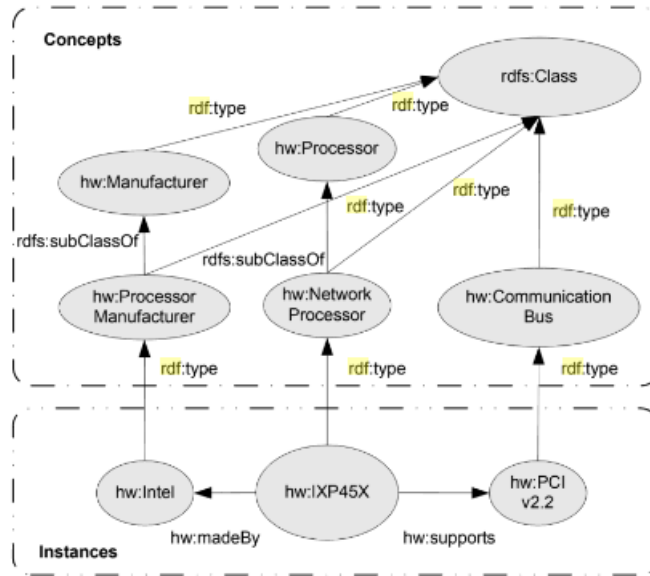


FIGURE 1.2 : EXEMPLE DE RDFS. [ABOULHAMID & ROUSSEAU, 2018]

3. Langage d'Ontologie Web (OWL)

OWL est un langage de représentation des connaissances qui peut être utilisé pour représenter les connaissances terminologiques d'un domaine d'une manière structurée et formellement bien comprise. Plus précisément, OWL est un langage logique de description. Les logiques de description expriment des descriptions conceptuelles avec une logique de prédicat de premier ordre. OWL est défini au-dessus de RDFS de la même manière que RDFS étend RDF. OWL définit une liste de ressources (sens et URIS) et un ensemble de règles d'implication qui permettent l'inférence des connaissances sous forme de nouveaux triplets. En particulier, OWL ajoute à RDFS la capacité d'exprimer les critères d'admissibilité d'une classe donnée (ensemble). Il est possible de définir une classe non seulement en définissant sa relation avec les autres classes mais aussi en définissant les critères qu'une ressource doit respecter pour être classée comme individu de la classe. Il existe de nombreux éléments sémantiquement riches dans la spécification OWL, mais pour le contexte de ce chapitre, nous nous concentrerons sur le concept de restriction. Le concept d'une restriction définit une ressource de type owl: Class. Cette classe définit l'ensemble de tous les individus qui expriment des spécifications de restriction (critères) qui sont utilisées pour définir d'autres individus: Classe. Ces critères portent généralement sur l'existence de relations (déclaration de propriété) dont une

ressource individuelle doit faire partie. La figure 1.3 définit plus en détail la conceptualisation d'un processeur et d'un processeur réseau. L'exemple définit la classe Processor comme toutes les ressources qui font partie d'exactly 1 relation madeOf ainsi qu'au moins 1 relation prend en charge. La classe Network est définie comme étant un sous-ensemble de tous les processeurs qui font exactement 1 relation madeFor et l'objet de cette relation doit être "networking". L'exemple reprend la ressource IXPA45X des exemples précédents mais n'exprime rien sur son association avec une classe. Basé sur la sémantique de OWL, le raisonnement sur l'exemple conclurait que la ressource hw:IXPA45X est un processeur et un processeur de réseau car elle remplit tous les critères pour les deux ensembles.[Aboulhamid & Rousseau, 2018]

```
hw:Processor rdf:type owl:Class.
hw:Processor rdfs:subClassOf _:processorRestriction1.
_:processorRestriction1 rdf:type owl:Restriction.
_:processorRestriction1 owl:onProperty hw:madeBy.
_:processorRestriction1 owl:Cardinality "1"^^xsd:int.

hw:Processor rdfs:subClassOf _:processorRestriction2.
_:processorRestriction2 rdf:type owl:Restriction.
_:processorRestriction2 owl:onProperty hw:supports.
_:processorRestriction2 owl:min "1"^^xsd:int.

hw:NetworkProcessor rdf:type owl:Class.
hw:NetworkProcessor rdfs:subClassOf hw:Processor
hw:Processor rdfs:subClassOf _:processorRestriction3. Network
_:processorRestriction3 rdf:type owl:Restriction.
_:processorRestriction3 owl:onProperty hw:madeFor.
_:processorRestriction3 owl:hasValue "networking"^^xsd:string.
_:processorRestriction3 owl:Cardinality "1"^^xsd:int.

hw:IXPA45X hw:madeBy hw:Intel.
hw:IXPA45X hw:madeFor "networking"^^xsd:string.
hw:IXPA45X hw:supports hw:PCiv2.2.
```

FIGURE 1.3 : EXEMPLE D'OWL. [Aboulhamid & Rousseau, 2018]

La spécification OWL définit 3 sous-ensembles du langage qui s'étendent les uns les autres: OWL Lite, OWL DL et OWL Full. Chaque sous-ensemble est équilibré entre l'expressivité et la complexité de calcul pour raisonner sur un modèle défini avec le sous-ensemble. Comme OWL s'appuie sur RDFS et RDF, il utilise les mêmes formats de sérialisation.[Aboulhamid & Rousseau, 2018]

4. Ontologie

En philosophie, l'ontologie est une discipline, un compte rendu systématique de l'existence / de l'être. Il concerne le monde tel qu'il est étudié par la science. En informatique, le concept d'ontologie est une représentation schématique de la réalité. La logique et la représentation formelle sont à la base d'une conceptualisation.

En outre, " **Un ensemble de connaissances formellement représentées est basé sur une conceptualisation. Concepts et autres entités qui sont présumés exister dans un domaine d'intérêt et les relations qui les détiennent | ... Une conceptualisation est une vue abstraite et simplifiée du monde que nous souhaitons représenter dans un certain but.** "[Gruber, 1993]

Le W3C définit une ontologie comme suit: "**Une ontologie définit les termes utilisés pour décrire et représenter un domaine de connaissance.**". La figure 1.5 montre un extrait de DBpedia Ontologie.

```
1 <http://dbpedia.org/resource/William_Shakespeare>  
  <http://dbpedia.org/property/dateOfBirth> "April 1564" .  
2 <http://dbpedia.org/resource/William_Shakespeare>  
  <http://dbpedia.org/ontology/child>  
  <http://dbpedia.org/resource/Judith_Quiney> .
```

FIGURE 1.4 : EXEMPLE DE FORMAT N-TRIPLES. [MORSEY ET AL., 2012A]

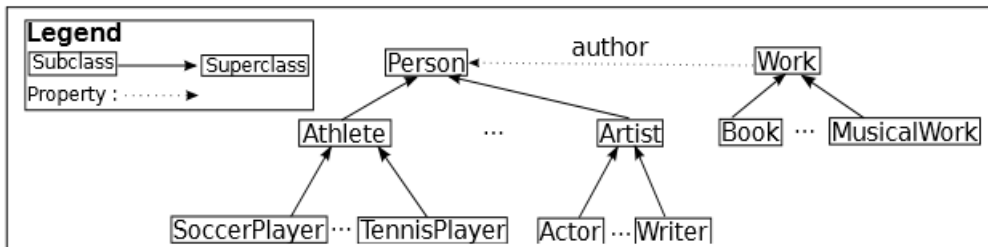


FIGURE 1.5 : EXTRAIT DE L'ONTOLOGIE DBPEDIA. [MORSEY ET AL., 2012A]

Cette ontologie dit qu'il existe une classe appelée «Writer» qui est une sous-classe de «Artist», qui est à son tour une sous-classe de «Person». William Shakespeare et Dan Brown sont considérés comme des instances de la classe «Writer». Notez qu'il existe une propriété reliant une instance de la classe "Work" à une instance de la classe "Person". Par exemple, le roman intitulé «Le symbole perdu» est une instance de la classe «Travail» et lié via la propriété «auteur» à son auteur «Dan Brown».[Morsey et al., 2012a]

5. L'extraction des connaissances

Les applications Web sémantiques s'appuient sur RDF et OWL pour représenter les connaissances utilisées pour décrire des contextes spécifiques. Comme nous l'avons décrit dans les sections précédentes, ces langages formels sont compréhensibles par la machine. En effet, ils permettent un espace où les machines informatiques sont les citoyens de première classe. Les humains, au contraire, utilisent un mode de communication déférent et son fondement repose sur l'utilisation du langage naturel (NL) pour exprimer la sémantique. Cet écart entre les deux langues déférentes a été comblé par l'Information Extraction (IE) approches développées par la communauté de recherche Natural Language Processing (NLP). Le but de l'extraction d'informations (IE) est de trouver des informations souhaitées, telles que des concepts (hiérarchie de termes utilisés pour pointer vers des définitions partagées), des entités (nom, expression numérique, date) dans des textes en langage naturel et les imprimer sous une forme adaptée à l'interrogation et au traitement automatiques. La plupart des informations stockées sous forme numérique sont cachées dans des textes en langage naturel pour les machines informatiques. Une approche prometteuse pour accéder par programmation à ces connaissances utilise des techniques IE pour réduire les textes en langage naturel à des structures tabulaires, à partir desquelles il est possible de récupérer des jetons de texte comme réponses à des requêtes. De nombreuses techniques IE reposent sur des modèles prédéfinis et des règles d'extraction basées sur des modèles ou des techniques d'apprentissage automatique pour identifier certaines entités dans des documents texte. Habituellement, les textes utilisent des vocabulaires, des structures et des styles de composition illimités pour définir approximativement le même contenu, ce qui rend difficile pour toute technique IE de couvrir toutes les variations du modèle d'écriture. D'autres approches ont été proposées et elles vont de l'approche par dictionnaire et de l'approche probabiliste. La première approche est souvent utilisée lorsqu'un petit ensemble de termes doit être identifié dans un texte, au lieu que ce dernier couvre une grande partie de ces techniques et de cet usage. Plus important encore, les techniques IE traditionnelles ne possèdent pas les connaissances de domaine requises pour créer des relations entre les entités extraites. Pour combler cette lacune, la communauté PNL a adopté des ontologies pour piloter les annotations. Jusqu'à présent, pour chaque jeton extrait dans le texte, il est classifié automatiquement selon une

hiérarchie de termes (c'est-à-dire l'ontologie). Les jetons sont principalement des mots clés ou des entités nommées et ils diffèrent de l'objectif d'identification: un mot clé est l'un des mots trouvés dans un texte, au lieu de cela, une entité nommée est principalement un nom ou une expression du temps (nous pouvons définir qu'une entité nommée est une spécialisation d'un mot clé). Les tendances récentes de la recherche montrent comment la tâche de reconnaissance d'entité nommée (NER) a joué un rôle incroyable dans la communauté du Web sémantique. Principalement parce que la reconnaissance d'entités nommées dans un texte en langage naturel identifie de vrais objets verbaux prêts à être consommés dans le LOD. Ceci est crucial pour les données existantes qui hébergent une grande quantité d'informations lisibles par l'homme, mais elles sont inaccessibles en raison du temps nécessaire pour les extraire. Ainsi, si le processus de sortie de ces techniques produit des entités nommées, elles peuvent être ambiguës via des URI Web qui identifient des objets du monde réel. Il s'agit du dernier défi de ce processus et plusieurs efforts ont été consacrés à sa résolution.[Rizzo, 2012]

6. Les types des graphes de connaissances

- ✓ L'une des bases de connaissances sémantiques les plus connues est Google Knowledge Graph qui se compose de nœuds (sujets) connectés les uns aux autres par les bords (relations), bien sûr, sémantiquement.
- ✓ Freebase est un projet acheté par Google qui représente une base de données de données structurées du Web. Son service a été fermé par Google en août 2016.
- ✓ Le protocole Facebook Open Graph est un protocole créé par Facebook inspiré de Dublin Core, Microformats et RDFa. Selon le site officiel, "il permet à chaque page Web de devenir un objet enrichi d'un graphe social qui est" un graphe qui représente les relations personnelles des internautes. En bref, c'est un modèle ou une représentation d'un réseau social, où le mot graphe est tiré de la théorie des graphes. Le graphique social a été appelé «la cartographie globale de tout le monde et comment ils sont liés».
- ✓ Dbpedia est une infrastructure de données publiques Wikipédia pour un grand graphe de connaissances sémantique multilingue. Il s'agissait d'un projet dans lequel le contenu d'un article Wikipédia a été mis au format csv pour avoir un

meilleur accès aux connaissances et pouvoir l'intégrer dans un graphe de connaissances. (*Insight into Semantic Web and why is it important today*, s. d.)

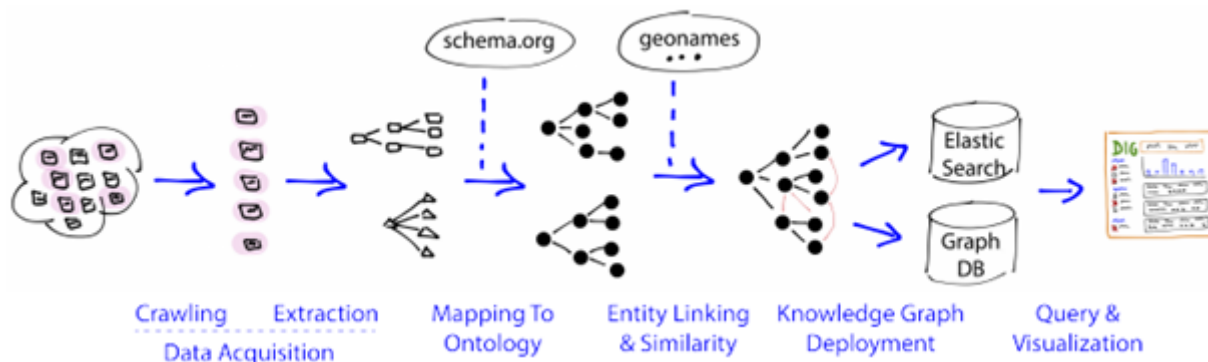


FIGURE 1.6 : L'ARCHITECTURE DU GRAPHE DES CONNAISSANCES. (*INSIGHT INTO SEMANTIC WEB AND WHY IS IT IMPORTANT TODAY*, S. D.)

7. Graphes des connaissances d'entreprise

Inspiré de Google et d'autres graphiques de connaissances, un graphique de connaissances d'entreprise (ECG) est un graphique intégré à l'échelle de l'entreprise créé par une organisation qui servira de source centralisée de connaissances et d'inférences intégrées.

Le passage aux électrocardiogrammes a été accéléré au cours des derniers mois par deux nouveaux développements. L'une est l'ajout de la base de données de graphes Neptune au portefeuille de bases de données d'Amazon. Le deuxième est le financement de systèmes de Cloud et de graphes sur site comme TigerGraph et d'autres startups de Bay Area. De nombreux architectes graphiques de Google, LinkedIn et Facebook (tous utilisant des bases de données graphiques) s'aventurent désormais seuls à développer des solutions pour l'entreprise.

Les électrocardiogrammes présentent toujours un ensemble intéressant de problèmes qu'ils doivent résoudre pour être vraiment considérés comme de classe entreprise.

L'ajout de transactions, le versionnage bitemporel, le contrôle d'accès basé sur les rôles, l'audit, l'évolutivité et la haute disponibilité sont tous des problèmes avec lesquels les systèmes traditionnels de preuve de concept open source ont du mal. Les universitaires

ne sont généralement pas intéressés par ces problèmes réels. (*Insight into Semantic Web and why is it important today*, s. d.)



FIGURE 1.7 : EXEMPLE DE GRAPHE DE CONNAISSANCES D'ENTREPRISE. (*INSIGHT INTO SEMANTIC WEB AND WHY IS IT IMPORTANT TODAY*, S. D.)

8. Les étapes d'extractions des connaissances

8.1. Mappage d'entité

Le composant Mappage d'entité a pour objectif de mapper une entité en texte non structuré à un identificateur de ressource uniforme (URI) en sortie. Dans le composant Mappage d'entités, les droits sont reconnus à partir du texte non structuré pour créer un ensemble d'entités extraites. Si une entité extraite peut être mappée à une entité identique dans n'importe quel KG, l'URI d'une telle entité dans ce KG doit être utilisé comme représentant de l'entité extraite. Sinon, un nouvel URI est donné à l'entité. Par exemple, considérons «dbpedia: United States» comme un URI dans le KG. Si l'entité «États-Unis» dans le texte non structuré est mappée sur «dbpedia: États-Unis», le même URI est utilisé. En revanche, si la même entité n'existe pas dans le KG, un

nouvel URI, par exemple «ex: États-Unis», est affecté à l'entité «États-Unis». Les résultats attendus du composant Mappage d'entités sont un ensemble d'entités de mappage pour chaque entité, par exemple, Barack Obama = {dbpedia: Barack Obama}[Kertkeidkachorn & Ichise, 2017]

8.2. Résolution de coréférence

L'objectif du composant Résolution de référence est de détecter les chaînes d'entités référentielles dans du texte non structuré et de regrouper ces entités. C'est également un élément essentiel car le texte non structuré contient généralement des abréviations, des pronoms et différentes expressions d'entités qui pointent vers les mêmes entités. Avec le composant Résolution de référence, une entité et ses différentes expressions peuvent être regroupées afin que les actions d'entités identiques dans différentes expressions puissent être capturées. Pour découvrir les chaînes d'entités référentielles, un résolveur de référence est utilisé.[Kertkeidkachorn & Ichise, 2017]

8.3. Extraction des triples

Le composant Triple Extraction a pour objectif d'extraire des triplets de relation à partir de texte non structuré. Il s'agit d'une étape clé pour acquérir des connaissances à partir d'un texte non structuré. Selon la théorie linguistique [Fillmore, 1976], le sens d'une phrase arbitraire peut être interprété en considérant un ensemble de relations et ses arguments associés. Par conséquent, un triple de relation est défini comme un triple décrivant une relation et ses arguments associés dans une phrase arbitraire. Dans notre scénario, la relation est un prédicat d'un triple et ses arguments associés sont un sujet d'un triple et un objet d'un triple. Pour extraire un triple de relation à partir de texte non structuré, toute technique d'extraction d'informations ouvertes peut être utilisée. Une technique d'extraction d'informations ouverte est utilisée pour extraire des informations dans une phrase arbitraire à l'aide de modèles de modèle, puis pour convertir ces informations en un triple de relation. Dans une technique d'extraction d'information ouverte, une relation et ses arguments associés dans une phrase sont identifiés sans utiliser ni connaissance préalable du domaine ni vocabulaire prédéfini.[Kertkeidkachorn & Ichise, 2017]

8.4. Triple intégration

L'objectif du composant Triple intégration est de générer des triplets de texte en utilisant les sorties du composant Mappage d'entité, du composant Résolution de Coréférence et du composant Triple Extraction. Dans le composant Triple Extraction, nous pouvons extraire des triplets de relation à partir de texte non structuré; cependant, le mappage d'entités et la résolution de coréférence entre les entités de ces triplets ne sont pas effectués. En conséquence, l'ambiguïté dans le triple se produit et l'interconnexion aux entités dans le KG n'est pas établie. Par conséquent, la transformation d'une relation triple conforme à la norme de KB est requise. Par conséquent, pour faire face à de tels problèmes, les résultats de trois composants sont intégrés et transformés par les processus suivants. Tout d'abord, des entités identiques sont regroupées à l'aide de chaînes de renvoi de cœur du composant Résolution de référence. Deuxièmement, un représentant du groupe d'entités référentielles est sélectionné par l'algorithme de vote. Étant donné que les entités d'un même groupe peuvent avoir différentes représentations, la majorité excluant les pronoms du groupe est choisie comme représentant du groupe. Troisièmement, toutes les entités appartenant au groupe dans la relation triple sont remplacées par le représentant de son groupe. Quatrièmement, la relation d'une relation triple est directement transformée en un prédicat en attribuant un nouvel URI. Enfin, si un objet d'une relation triple n'est pas une entité, il reste littéral. Après avoir effectué ces processus, les triplets de texte sont extraits du texte non structuré.[Kertkeidkachorn & Ichise, 2017]

9. Conclusion du chapitre

Dans ce chapitre, nous avons présenté les concepts et techniques liées au domaine du graphe de connaissance, En effet, nous avons présenté le graphe de connaissance.

Dans ce chapitre nous avons cité aussi l'extraction des connaissances et leurs étapes.

Chapitre 2

Les travaux Connexes

1. Introduction

Dans ce chapitre, nous présentons les travaux en relation de notre travail.

2. Les travaux connexes

Il existe plusieurs projets qui visent à l'extraction des graphes de connaissances depuis des données non structurés.

2.1 L'approche[Morsey et al., 2012a]

L'approche[Morsey et al., 2012a] est basée également sur Wikipédia, DBpedia a analysé les informations de Wikipedia en tant que données brutes, puis extrait les instructions RDF et les a stockées, dbpedia extrait des informations structurées de Wikipedia, les relie à d'autres bases de connaissances et publie librement les résultats sur le Web à l'aide de données liées et de SPARQL. Extraction les données RDF et les stocker dans un triplestore. DBpediaLive ajoute un certain nombre de fonctionnalités et résout en particulier les problèmes suivants:

- Les modèles MediaWiki dans les résumés d'articles sont maintenant rendus correctement.
- Les changements de mappages dans le wiki de mappages DBpedia.
- DBpedia-Live publie les triplets nouvellement ajoutés / supprimés dans les fichiers, afin de permettre la synchronisation entre le point de terminaison DBpedia et d'autres miroirs DBpedia. L'approche utilise OAI-PMH pour recevoir les événements de modification de Wikipedia et les traiter pour créer DBpedia Live, et affine le cadre d'extraction en ajoutant un outil pour importer en continu de nouveaux ensembles de modifications, des extracteurs supplémentaires. Le cœur de DBpedia consiste en un processus d'extraction d'infobox. . L'approche d'extraction d'infobox générique traite une infobox comme suit: Un URI DBpedia est créé à partir de l'URL de l'article Wikipedia l'infobox est contenu dans. Cet URI est ensuite utilisé comme sujet pour tous les triplets extraits. L'URI de prédicat est créé en concaténant l'espace de noms. Les objets sont créés à partir de l'attribut valeurs. Mapper les modèles Wikipédia à une ontologie. Les mappages de propriétés définissent également des règles précises sur la façon d'analyser les valeurs d'infobox et de définir les types de données cibles, qui aident les analyseurs à traiter les valeurs d'attribut.

2.2.1 Les points forts

- DBpedia live [Morsey et al., 2012a] fournit les modifications de Wikipedia sous forme de données structurées, conformes à l'ontologie DBpedia.
- DBpedia Live[Morsey et al., 2012a] est un graphe de connaissances reflète Wikipedia tout en l'affinant avec plus de connaissances ontologiques.

2.2.2 Les points faibles

- [Morsey et al., 2012a] DBpedia possède toujours une ontologie légère qui manque d'axiomes essentiels.

2.3 LODifier [Augenstein et al., 2012]

LODifier [Augenstein et al., 2012] est une stratégie qui vise à traduire l'entrée textuelle dans son intégralité en une représentation structurelle RDF. LODifier [Augenstein et al., 2012] est une implémentation de preuve de concept qui convertit un langage naturel non structuré en domaine ouvert en données liées, avec l'utilisation de techniques robustes de traitement du langage naturel PNL telles que la reconnaissance d'entité nommée (NER), la désambiguïsation de Word-Sense (WSD) et une analyse sémantique approfondie, incorporant la sortie RDF dans le cloud Linked Open Data (LOD) en utilisant le vocabulaire de DBpedia et WordNet 3.0. Le système utilise NER Wikifier pour reconnaître les mentions d'entrée et mappées sur des URI DBpedia, utilise C&C et Boxer pour extraire les relations sémantiques, qui génère des structures de représentation du discours (DRS), le texte est lemmatisé et les mots sont désambiguïsés avec l'outil WSD UKB pour obtenir les mappages WordNet, le graphe RDF est ensuite créé en traitant davantage la sortie du Boxer DRS, en la transformant en triplets. Enfin, il est enrichi des URI DBpedia et des URI sense WordNet.

2.3.1 Les points forts

- La sortie graphique de LODifier [Augenstein et al., 2012] est liée à d'autres sources de données.
- Cette approche[Augenstein et al., 2012] fonctionnent bien pour extraire des triplets (sujet, prédicat, objet) d'un texte non structuré.

2.3.2 Les points faibles

- LODifier[Augenstein et al., 2012] peut résoudre l'ambiguïté d'une entité extraite, mais tous les éléments d'un triple ne sont pas encore intégrés dans d'autres KG.

- LODifier[Augenstein et al., 2012] extrait des informations sémantiques et appliqué une résolution de coréférence. mais les entités extraites par ces systèmes n'ont pas été intégrées à une seule ontologie homogène.
- Cette approche[Augenstein et al., 2012] a une limitation concernant le mappage d'un prédicat.

2.4 Entity Extraction [Exner & Nugues, 2012]

Entity Extraction [Exner & Nugues, 2012] est un système proposé est basé sur un pipeline qui prend les articles Wikipédia en entrée et produit des entités sous la forme de triplets DBpedia RDF. L'approche[Exner & Nugues, 2012] examine tous les aspects de la création d'un GC. De plus, l'approche[Exner & Nugues, 2012] propose une méthode de cartographie d'ontologie qui amorce l'apprentissage à partir de triplets existants à partir du jeu de données DBpedia. Dans l'approche[Exner & Nugues, 2012], les auteurs exécutent l'extraction de relations ouvertes sur du texte Wikipedia et effectuent une cartographie a posteriori de l'ontologie DBpedia. DBpedia fait partie du cloud de données liées, de sorte que les résultats peuvent être exprimés sous forme de triplets RDF, ce qui les rend faciles à interroger et à réutiliser dans d'autres applications.

2.4.1 Les points forts

- Cette approche[Exner & Nugues, 2012] extrait toutes les relations d'entité du texte brut et tente de mapper les entités sur l'espace de noms DBpedia.
- Bien qu'ils ne traitent pas des nombres, leur approche peut extraire des littéraux de date et atteint une précision globale de 74,3%.

2.4.2 Les points faibles

- L'approche[Exner & Nugues, 2012] dépend fortement des règles générées, En tant que de la rareté du texte non structuré dans les domaines ouverts, les règles générées ne peuvent pas couvrir tous les modèles possibles, donc certaines règles manquent.

2.5 Le système T2KG

Le système T2KG [Kertkeidkachorn & Ichise, 2017] se concentre principalement sur le mappage d'un prédicat d'un triple extrait de texte non structuré. T2KG[Kertkeidkachorn & Ichise, 2017] est conçu pour prendre du texte non structuré en entrée et produire un KG en sortie. T2KG[Kertkeidkachorn & Ichise, 2017] a cinq composants:

Le composant Mappage d'entité mappe les entités de texte aux entités DBpedia. Le composant Résolution de coréférence qui recherche et remplace toutes les vocabulaires qui compense la même entité dans le texte. Le triple extracteur extrait une relation triple d'un texte non structuré utilise des techniques d'extraction d'informations ouverte. Le composant de traitement des métadonnées prépare et stocke l'ensemble des prédicats DBpedia candidats pour un prédicat de texte donné, qui pourrait éventuellement être le mappage. Le composant Mappage de prédicat utilise les prédicats du triple de texte pour rechercher à un prédicat prédéfini dans GC existant. Lors le prédicat n'existe pas dans le GC existant, le système utilise des nouveaux approches pour la recherché :

- Une approche basée sur des règles et
- Une approche basée sur la similitude

2.5.1 Les points forts

- Le système T2KG[Kertkeidkachorn & Ichise, 2017] peut générer avec succès un KG à partir de texte non structuré.

- Le système T2KG[Kertkeidkachorn & Ichise, 2017] traite des vocabulaires hétérogènes et atténuer la rareté du texte non structuré.

- Dans le système T2KG[Kertkeidkachorn & Ichise, 2017] les résultats expérimentaux indiquent que l'approche hybride améliore à la fois la précision et le rappel pour la cartographie d'un prédicat à un KG.

2.5.2 Les points faibles

- l'approche ne semble pas rendre les artefacts logiciels open source disponibles pour téléchargement.

- Le système T2KG [Kertkeidkachorn & Ichise, 2017] vérifier tous les triplets DBpedia pour mapper un seul prédicat de KG.

- il utilise un modèle basé sur word2vec qui n'est pas capable de capturer toutes les caractéristiques sémantiques car il utilise des réseaux neuronaux peu profonds.

3 Conclusion du chapitre

Les travaux Connexes

Dans ce chapitre, nous avons présenté les travaux en relation notre travail. On a montré aussi les points faibles et forts de ces travaux.

Chapitre 3

L'approche proposée

1. Introduction

Dans ce chapitre on va présenter notre approche proposée qui consiste à enrichir les résultats de l'extracteur Dbpedia qui extrait des triples RDF depuis les pages Wikipédia. Notre objectif principale est améliorer les fichiers RDF par des termes similaires afin faciliter la recherches des connaissances. Afin d'achever ce travail, on a proposé une mesure de similarité basée sur les sens depuis la base WordNet. On va présenter aussi un processus de prétraitement des fichiers RDF afin de faciliter et augmenter la précision de calcule de similarité. Premièrement, on va expliquer les étapes de l'extracteur de Dbpedia et ensuite on va compléter d'expliquer notre approche.

2. Vue d'ensemble sur DBpedia

Le cœur de DBpedia consiste en un processus d'extraction d'infobox. Les Infobox sont des modèles contenus dans de nombreux articles Wikipédia. Ils sont généralement affichés dans le coin supérieur droit des articles et contiennent des informations factuelles (Figure 3.8). Les infobox affichent les faits les plus pertinents d'un article Wikipédia dans un tableau souvent sous la forme de paires attribut-valeur.

Le système de modèles d'infobox de Wikipédia a évolué au fil du temps sans coordination centrale. En d'autres termes, différentes communautés utilisent différents modèles pour décrire le même type de choses (par exemple, `infobox_city_japan`, `infobox_swiss_town` et `infobox_town_de`). De plus, différents modèles d'infobox utilisent des noms différents pour désigner le même attribut (par exemple, `birthplace` et `placeofbirth`). Les valeurs d'attribut sont également exprimées à l'aide d'une large gamme de formats et d'unités de mesure différents (par exemple, `hauteur = {{ hauteur | ft = 5 | in = 7 }}` et `hauteur = {{ hauteur | m = 1,70 }}`). Pour surmonter cet obstacle il y a deux approches d'extraction différentes fonctionnant en parallèle: une approche générique qui vise une large couverture et une approche basée sur la cartographie qui vise une meilleure qualité des données.[Morsey et al., 2012b]

2.1 Extraction d'Infobox générique ou basée sur la cartographie

2.1.1 Extraction d'Infobox générique : L'approche générique d'extraction de l'infobox traite une infobox comme suit: Un URI DBpedia est créé à partir de l'URL de l'article Wikipédia l'infobox est contenu dans. Cet URI est ensuite utilisé comme sujet pour tous les triplets extraits. Pour exemple, l'URL de l'article Wikipédia

http://en.wikipedia.org/wiki/William_Shakespeare est utilisé pour créer la ressource http://dbpedia.org/resource/William_Shakespeare, qui est ensuite utilisé comme sujet. L'URI du prédicat est créé en concaténant l'espace de noms <http://dbpedia.org/property/> avec le nom d'attribut infobox. Par exemple, l'attribut Wikipédia lieu de naissance donne la propriété <http://dbpedia.org/property/birthplace>. Les objets sont créés à partir de l'attribut valeurs. Ces valeurs sont prétraitées et converties en RDF pour obtenir une valeur appropriée représentations. Par exemple, les liens internes Media Wiki sont convertis en URI DBpedia les références, les listes sont détectées et représentées en conséquence, les unités sont détectées et converties aux types de données standard. Le principal avantage de cette approche est qu'elle peut couvrir toutes les infobox types ainsi que leurs attributs. Son inconvénient est que synonyme, c'est-à-dire attribut équivalent les noms ne sont pas résolus. Cela rend l'écriture de requêtes sur des données d'infobox génériques plutôt complexe.(Morsey et al., 2012b)

2.1.2 Extraction d'infobox basée sur la cartographie : Afin de surmonter les problèmes de noms d'attributs et plusieurs modèles utilisés pour le même type de choses, nous avons mappé Modèles de Wikipédia vers une ontologie. L'ontologie a été créée manuellement en organisant le 341 modèles d'infobox les plus couramment utilisés dans l'édition anglaise de Wikipédia dans une hiérarchie de subsomption composée de 314 classes, puis mappant 2350 attributs de dans ces modèles à 1425 propriétés d'ontologie. Les mappages de propriétés définissent également des règles précises sur la façon d'analyser les valeurs de l'infobox et de définir les types de données cibles, qui aident le analyseurs pour traiter les valeurs d'attribut. Par exemple, si un mappage définit le type de données cible sur être une liste de liens, l'analyseur ignorera le texte supplémentaire qui pourrait être présent dans l'attribut valeur. L'ontologie utilise actuellement 376 types de données différents. Les unités de mesure déviantes sont normalisées à l'un de ces types de données. Les données d'instance dans l'ontologie infobox sont donc plus propres et mieux structurées que les données générées à l'aide de l'approche d'extraction générique.(Morsey et al., 2012b)

2.2 Extracteurs DBpedia

Un extracteur DBpedia est un module exécutable, chargé d'extraire un morceau spécifique de données d'un article de Wikipédia. Par exemple, l'extracteur de résumé extrait le résumé

d'un Article Wikipédia, c'est-à-dire le texte avant la table des matières de cet article.(Morsey et al., 2012b)

```

{{Infobox settlement
| official_name       = Algarve
| settlement_type    = Region
| image_map          = LocalRegiaoAlgarve.svg
| mapsize            = 180px
| map_caption        = Map showing Algarve
                    = Region in Portugal
| subdivision_type   = [[Countries of the
                    = world|Country]]
| subdivision_name   = {{POR}}
| subdivision_type3  = Capital city
| subdivision_name3  = [[Faro, Portugal|Faro]]
| area_total_km2     = 5412
| population_total    = 410000
| timezone           = [[Western European
                    = Time|WET]]
| utc_offset         = +0
| timezone_DST       = [[Western European
                    = Summer Time|WEST]]
| utc_offset_DST     = +1
| blank_name_sec1    = [[NUTS]] code
| blank_info_sec1    = PT15
| blank_name_sec2    = [[GDP]] per capita
| blank_info_sec2    = €19,200 (2006)
}}

```



Country	 Portugal
Capital city	Faro
Area	
- Total	5,412 km ² (2,089.6 sq mi)
Population	
- Total	410,000
Time zone	WET (UTC+0)
- Summer (DST)	WEST (UTC+1)
NUTS code	PT15
GDP per capita (PPS)	€ 19,200 (2006) ^[1]

FIGURE 3.8 : SYNTAXE DE L'INFOBOX MEDIAWIKI POUR L'ALGARVE (A GAUCHE) ET RENDU DE L'INFOBOX (A DROITE).

(MORSEY ET AL., 2012B)

Actuellement, le framework dispose de 19 extracteurs qui traitent les types de contenu Wikipédia suivants:

- **Étiquettes** : Tous les articles Wikipédia ont un titre, qui est utilisé comme **rdfs: etiquettes** pour la ressource DBpedia correspondante.
- **Résumés**: Nous extrayons un court résumé (premier paragraphe, représenté en utilisant **rdfs: comment**) et un long résumé (texte avant une table des matières, en utilisant la propriété **dbpedia: abstract**) de chaque article.
- **Liens inter-langues**: Nous extrayons des liens qui relient des articles sur le même sujet dans différentes éditions linguistiques de Wikipédia et les utilisons pour attribuer des étiquettes et des résumés dans différentes langues aux ressources DBpedia.
- **Images**: Les liens pointant vers les images Wikimedia Commons représentant une ressource sont extraits et représentés à l'aide de la propriété **foaf: depiction**.
- **Redirige**: Afin d'identifier des termes synonymes, les articles de Wikipédia peuvent rediriger vers d'autres articles. Nous extrayons ces redirections et les utilisons pour résoudre les références entre les ressources DBpedia.

- **Désambiguïsation:** Les pages d'homonymie de Wikipedia expliquent les différentes significations des homonymes. Nous extrayons et représentons les liens de désambiguïsation en utilisant le prédicat **dbpedia: wiki Page Disambiguation**.
- **Liens externes:** Les articles contenant des références à des ressources Web externes que nous représentons en utilisant la propriété DBpedia **dbpedia: wikiPageExternalLink**.
- **Liens de page:** Nous extrayons tous les liens entre les articles Wikipédia et les représentons en utilisant la propriété **dbpedia: wikiPageWikiLink**.
- **Page Wiki:** Lie un article Wikipedia à sa ressource DBpedia correspondante, par exemple:
(<http://en.wikipedia.org/wiki/Germany>
<http://xmlns.com/foaf/0.1/primaryTopic>
<http://dbpedia.org/resource/Germany>).
- **Page d'accueil:** Cet extracteur obtient des liens vers les pages d'accueil d'entités telles que des entreprises et des organisations en recherchant les termes page d'accueil ou site Web dans les liens d'articles (représentés par **foaf: homepage**).
- **Géo-coordonnées:** Le géo-extracteur exprime les coordonnées en utilisant le vocabulaire de base Geo (**WGS84** lat / long) et l'encodage **GeoRSS** simple du vocabulaire géospatial du **W3C**. Le premier exprime les composants de latitude et de longitude en tant que faits séparés, ce qui permet un filtrage de surface simple dans les requêtes **SPARQL**.
- **Données personnelles:** Il extrait des informations personnelles telles que le nom et la date de naissance. Ces informations sont représentées dans des prédicats tels que **foaf: nom de famille** et **dbpedia: date de naissance**.
- **PND.** Pour chaque personne, il existe un dossier contenant son nom, sa naissance et sa profession liés à un identifiant unique, qui est le numéro **PND** (Personennamendatei). Les PND sont publiés par la bibliothèque nationale allemande. Un **PND** est lié à sa ressource via **dbpedia: individualisedPnd**.
- **Catégories SKOS:** Il extrait des informations sur quel concept est une catégorie et comment les catégories sont liées à l'aide du vocabulaire **SKOS**.
- **ID de page:** Chaque article Wikipedia a un identifiant unique. Cet extracteur extrait cet ID et le représente à l'aide de **dbpedia: wiki PageID**.

- **ID de revision:** Chaque fois qu'un article Wikipedia est modifié, il obtient un nouvel ID de révision. Cet extracteur extrait cet ID et le représente à l'aide de **dbpedia:wikiPageRevisionID**.
- **Étiquette de catégorie:** Les articles Wikipédia sont organisés en catégories, et cet extracteur extrait les étiquettes de ces catégories.
- **Catégories d'articles:** Associe chaque ressource DBpedia à ses catégories correspondantes.
- **Boîte d'info:** Il extrait toutes les propriétés de toutes les infoboxes comme décrit. Les informations extraites sont représentées à l'aide des propriétés de l'espace de noms **http://dbpedia.org/property/**. Les noms de ces propriétés reflètent les noms des infoboxes attribuées de Wikipedia sans aucune modification (non mappées).
- **Mappages:** Il extrait des données structurées basées sur des mappages créés manuellement des infoboxes Wikipédia avec l'ontologie DBpedia. Tout d'abord, il charge tous les mappages d'infobox définis pour les langues requises, DBpedia-Live prend en charge la langue anglaise uniquement pour le moment, à partir du wiki des mappages. Le wiki mappings est disponible à **http://mappings.dbpedia.org**. Il extrait ensuite la valeur de chaque propriété Wikipedia définie pour ce type d'infobox et génère un triplet approprié pour celle-ci, en fonction des mappages.(Morsey et al., 2012b)

3. Architecture générale du système

L'architecture générale du système de DBpedia-Live est illustrée à la figure 3.9.

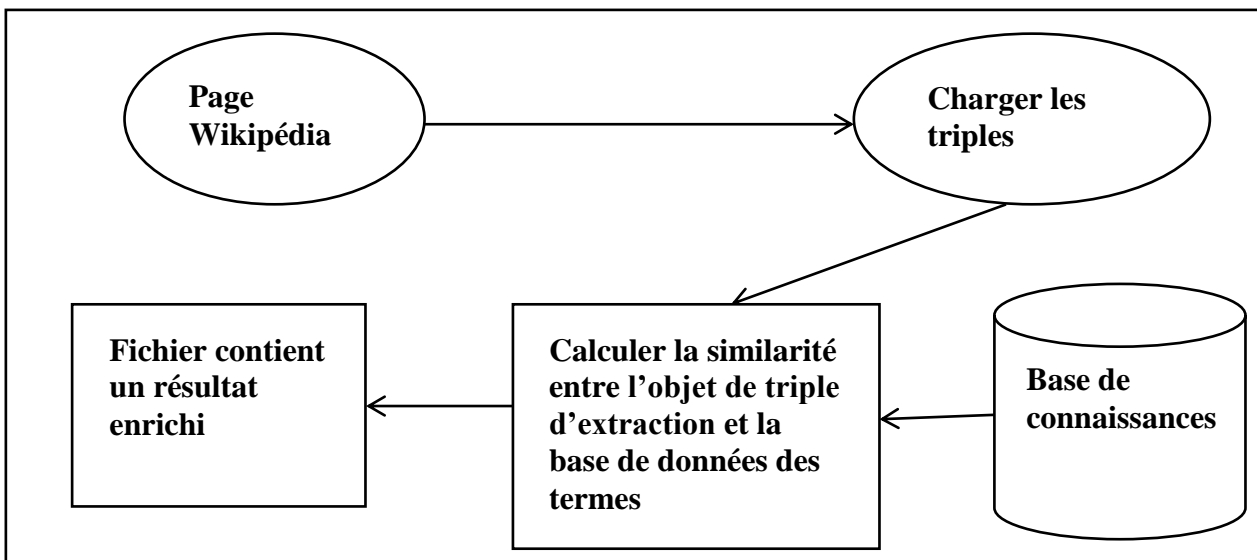


FIGURE 3.9 : ARCHITECTURE GENERALE DU SYSTEME.

Nous commençons notre système par un page Wikipédia, et puis charger les triples, ensuite calculer la similarité entre l'objet de triple d'extraction et la base de connaissances qui contient toutes les textes.

4. Le prétraitement de résultat de l'extraction framework de DBpedia

4.1 Le prétraitement

Depuis le domaine d'approximations pour les graphes RDF, les tenants de ces derniers provoquent des difficultés pendant le processus d'approximation. La plupart des données RDF sont créées de façon automatique par des machines donc ils peuvent contenir des informations non compatibles avec le langage naturel. Le prétraitement de ces données est la solution pour cette situation afin d'améliorer la qualité des réponses approximatives. L'approche supporte le prétraitement automatique et manuel. Par exemple si une requête contient le mot « computer science », mais le graphe contient « www.univ-annaba/computer_science » donc le prétraitement va traiter cette dernière et essaye de reprocher son contenu vers le langage naturel et ceci est un processus très important. Le résultat de prétraitement sur un graphe RDF est un graphe simple avec des nœuds et arcs lisible. Dans cette approche, ce graphe est appelé un graphe RDF normalisé donc le terme normalisation est utilisé dans le reste de thèse.

4.1.1 Normalisation des triples RDF

Les données en RDF sont composées d'une collection de triples. Il faut normaliser ces derniers afin de faire une normalisation globale de graphe RDF. Un triple RDF est composé d'un sujet, prédicat et objet donc il faut traiter ces derniers pour une normalisation de ce triple. L'approche appliqué plusieurs techniques afin de faire une bonne normalisation, et qui consistent en la :

- Suppression des caractères additionnels : les composantes peuvent avoir des caractères comme : '/', '#', '*', '_', '-'. Ces derniers ne sont pas utilisés dans les requêtes donc il faut les supprimer pour faciliter la recherche et améliore le processus d'approximation. Par exemple, le terme « has_title » contient le caractère « _ » et par l'application de ce

processus on obtient « has title » qui est une phrase plus significative que le terme original.

- Décomposition des mots : les données peuvent contenir des mots qui sont composés de plusieurs sous mots. Dans la requête originale ces termes ne sont pas supportés donc il faut traiter et appliquer une décomposition de ces termes pour découvrir des sous mots plus lisibles. Par exemple, le mot « ComputerScience » est composé de deux sous mots « Computer » et « Science » qui sont plus lisibles que le mot original. Donc ce processus est très important afin d'améliorer et simplifier le calcul des valeurs des similarités entre les mots.
- Résoudre les abréviations : on obtient un graphe RDF avec un contenu plus significatif si on a une technique qui a la capacité de résoudre le problème des abréviations. Avec cette application, chaque abréviation sera remplacée par son mot original. Par exemple, le terme « SW » sera remplacé par « Semantic Web » qui est lisible que le premier. Ce processus nécessite l'accès à des ressources additionnelles comme les ontologies et les dictionnaires. On peut résoudre ce problème manuellement ou automatiquement par des programmes.
- Supprimer les nœuds blancs : le graphe RDF cible peut contenir des nœuds blancs qui peuvent être supprimés pour diminuer la taille de ces données et faciliter la recherche. Ces nœuds ne sont pas ciblés par les requêtes.

L'objectif principal de normalisation est de produire un graphe RDF normalisé qui contient des mots qui sont plus proches de langage naturel. Ceci va améliorer de façon importante l'approximation et la qualité des réponses approximatives pour une bonne satisfaction. Des ressources additionnelles comme WordNet, YAGO [Suchanek et al., 2008], Wikipedia et DBpedia peuvent être utilisées pendant la normalisation. Il faut respecter le domaine associé avec les données RDF cible pendant l'application des changements au niveau de contenu de ces graphes. Le prétraitement est vu comme une approximation appliquée aux données RDF.

On présente ici un exemple d'un triple RDF t_1 qui est normalisé vers un autre triple RDF $norm_t_1$ par l'application de processus de prétraitement.

t_1 : $norm_t_1$:
 $s = \text{www.univ} - \text{annaba.org}$ → $norm_s = \text{Annaba university}$

$p = \text{www.univ - annaba/resource/HasDomain} \rightarrow \text{norm}_p = \text{Has Domain}$ $o = \text{www.univ - annaba/Computer_Science} \rightarrow \text{norm}_o = \text{Computer Science}$

5. Une nouvelle mesure de similarité linguistique

Dans cette section, nous présentons une mesure de similarité linguistique basée sur les sens des concepts [Djeddaï et al., 2013]. L'ontologie WordNet est utilisée comme ressources lexicales pour l'extraction des sens des concepts. Ensuite, la mesure est étendue afin de calculer la similarité entre les concepts qui contiennent plusieurs termes.

5.1 Présentation de WordNet

WordNet [Fellbaum, 1998] est une ressource lexicale pour la langue anglaise. Il regroupe des termes (nom, verbes, adjectifs, etc.) dans des ensembles des synonymes nommés *synsets*. Un *synset* regroupe les termes qui représentent un concept donné. Les *synsets* sont reliées avec des relations sémantiques c.-à-d., relations de généralisation/spécialisation et relations de composé/composante. D'après la structure et le contenant de WordNet, ce dernier peut être utilisé comme une ressource qui aide à calculer la valeur de similarité linguistique entre les concepts. Pour un concept c , la fonction $\text{syn}(c)$ retourne tous les *synsets* associés avec le concept c .

5.2 Une mesure de similarité basée sur les sens des concepts

Dans cette section, nous présentons une mesure de similarité qui est basée sur les sens des concepts extraits depuis WordNet. La motivation principale est que les mesures basées sur la syntaxe ne sont capables de mesurer la similarité entre deux concepts donnés si ces derniers se diffèrent au niveau des caractères. Par exemple, les concepts « car » et « automobile » sont différents, mais ils ont une similarité au niveau des sens puisque ces deux concepts sont des machines utilisés pour le transport. L'utilisation d'une ressource additionnelle comme WordNet peut améliorer le calcul de similarité.

L'objectif principal de cette mesure est le calcul de valeur de similarité entre les étiquettes des nœuds et entre les étiquettes des arcs. Ceci se fait pendant les processus d'approximation où ces valeurs sont utilisées comme des facteurs pour mesurer le degré d'approximation dans les

chemins approximatifs et alors les réponses approximatives. Une présentation de cette métrique est donnée dans ce qui suit.

Soit $S_{com} = Syn(c1) \cap Syn(c2)$ l'ensemble des sens communs entre les deux concepts $c1$ et $c2$. Soit $min(|Syn(c1)|, |Syn(c2)|)$ la cardinalité minimum entre les deux ensembles $Syn(c1)$ et $Syn(c2)$. la mesure de similarité est trouvée depuis l'analyse de la mesure suivante [Fellah et al., 2008] :

$$sim1(c1, c2) = \frac{\lambda(S_{com})}{min(|Syn(c1)|, |Syn(c2)|)} \quad (3.1)$$

Cette métrique est basée sur les sens communs entre les concepts $c1$ et $c2$. Elle retourne 1.0 si $c1$ (ou $c2$) est un synonyme de $c2$ (ou $c1$), mais si l'ensemble des sens de $c1$ (ou $c2$) est dans l'ensemble des sens de $c2$ (ou $c1$) alors elle donne aussi 1.0. Par exemple, le concept "machine" a 8 sens et le concept "motorcar" a 1 sens qui est dans les sens de « machine ». Par l'utilisation de cette métrique, on trouve que : $sim1(machine, motorcar) = \frac{1}{min(8,1)} = \frac{1}{1} = 1$.

Donc « machine » est le synonyme de « motorcar » ce qui n'est pas correct puisque le premier concept est la généralisation de dernier concept. Le problème c'est que cette mesure est basée sur les sens communs donc vous proposez une mesure qui se base sur les sens qui ne sont pas communs. Soit $S_{dif}(c1, c2)$ l'ensemble des sens qui ne sont pas dans l'intersection de $Syn(c1)$ et $Syn(c2)$, donc :

$$S_{dif}(c1, c2) = union - intersection \quad (3.2)$$

$$S_{dif}(c1, c2) = |(Syn(c1) \cup Syn(c2))| - |(Syn(c1) \cap Syn(c2))| \quad (3.3)$$

Donc la mesure proposée est la suivante (où $U = Syn(c1) \cap Syn(c2)$ est l'ensemble d'union):

$$sim2(c1, c2) = 1 - \frac{|sense\ non\ communs|}{|union|} = 1 - \frac{|union| - |intersection|}{|union|} \quad (3.4)$$

$$Sim2(c1, c2) = 1 - \frac{|S_{dif}|}{|union|} = 1 - \frac{|S_{dif}|}{|(Syn(c1) \cap Syn(c2))|} \quad (3.5)$$

$$Sim2(c1, c2) = 1 - \frac{|U| - |S_{com}|}{|U|} \quad (3.6)$$

Si $Syn(c1) = Syn(c2)$ c.-à-d., pas des sens non communs puisque l'un est le synonyme de l'autre), alors $sim2(c1, c2) = 1 - 0 = 1$, $sim2(machine, motorcar) = 1 - \frac{7}{8} = 1 - 0.87 = 0.13$ (7 sens communs). Les limites suivantes montrent que les valeurs trouvées par la mesure proposée sont toujours dans [0,1].

Si le nombre de sens communs est augmenté vers le maximum alors la valeur de similarité s'étant vers 1. Ceci est indiqué dans la limite suivante :

$$\limite_{|S_{com}| \rightarrow |U|} Sim2(c1, c2) = 1 - \frac{|U| - |U|}{|U|} = 1 - 0 = 1 \quad (3.7)$$

Si le nombre de sens communs est diminué alors la valeur de similarité s'étant vers 0. Ceci est indiqué vers la limite suivante :

$$\limite_{|S_{com}| \rightarrow 0} Sim2(c1, c2) = 1 - \frac{|U| - 0}{|U|} = 1 - 1 = 0 \quad (3.8)$$

Les limites précédentes prouvent que la mesure proposée prend ses valeurs dans l'intervalle [0,1]. Pour profiter des avantages des deux mesures, nous avons combiné les deux dans une seule mesure de similarité comme suivant :

$$sim_senses(c1, c2) = \omega1 * sim1(c1, c2) + \omega2 * sim2(c1, c2) \quad (3.9)$$

Les poids $\omega1$ et $\omega2$ sont utilisés pour augmenter ou diminuer l'importance des valeurs des similarités entre les concepts $c1$ et $c2$. Ces poids sont ajustés selon les préférences des utilisateurs donc ils peuvent préférer la mesure basée sur les sens communs ou celle qui est basée sur les sens non communs. Les valeurs de $\omega1$ et $\omega2$ sont initialisées par défaut à 0.5.

5.3 Étendre la mesure de similarité

La mesure de similarité Sim_senses est utilisée dans le cas si les concepts comportent un seul terme comme « help » et « aid », mais elle n'est pas utile si les concepts sont composés de plusieurs termes comme « code of student ». L'objective dans cette section est de trouver une autre mesure qui est construite à partir de sim_senses .

Soit N_1 et N_2 les ensembles des termes respectivement dans les concepts $c1$ et $c2$. La mesure sim_senses est capable de trouver une valeur de similarité entre chaque pair des termes $(t1, t2)$ où $t1 \in N1$ et $t2 \in N2$. Ces valeurs sont utilisées pour calculer la similarité finale entre $c1$ et $c2$. Les termes ont des positions différentes dans les noms des concepts donc il faut respecter la différence entre ces positions pendant le calcul. La question suivante est posée: *comment représenter le respect de cette position dans la formule de calcul ?* La réponse à cette question est la pondération c.-à-d. des poids qui représentent ces différences. Ces poids sont calculés en utilisant la distance entre les positions des termes comme suit : soit i et j les

positions qui sont associés respectivement avec les termes t_1 et t_2 où $t_1 \in N_1$ et $t_2 \in N_2$. Le poids suivant est défini :

$$\omega(t_1, t_2) = 1 - \frac{\text{distance}(t_1, t_2)}{\text{max distance}} \quad (3.10)$$

$$\omega(t_1, t_2) = 1 - \left(\frac{\text{max}(i, j) - \text{min}(i, j)}{\text{max}(|N_1|, |N_2|) - 1} \right) \quad (3.11)$$

La distance est la différence entre les positions de t_1 et t_2 . Max distance est la distance maximale qui peut être trouvée entre deux termes dans N_1 et N_2 . Ceci est important pour que le poids soit toujours dans l'intervalle $[0,1]$ et donc la valeur de similarité toujours appartenant à l'intervalle $[0,1]$.

Dans ce qui suit nous présentons des limites qui prouvent que les valeurs trouver par $\omega(t_1, t_2)$ sont toujours dans l'intervalle $[0,1]$.

Si la valeur de distance est diminuée alors le poids calculé se dirige vers 1. Ceci est indiqué par la limite suivante :

$$\limite_{\text{distance}(t_1, t_2) \rightarrow 0} \omega(t_1, t_2) = 1 - 0 = 1 \quad (3.12)$$

Si la distance est augmentée au maximum c.-à-d., max distance alors la valeur de poids va diriger vers 0. Ceci est montré par la limite suivante :

$$\limite_{\text{distance}(t_1, t_2) \rightarrow \text{max distance}} \omega(t_1, t_2) = 1 - 1 = 0 \quad (3.13)$$

Les deux limites précédentes prouvent que la valeur de poids est toujours dans l'intervalle $[0,1]$. Ceci est relié avec la distance calculée qui a une influence importante sur le poids trouvé. Après trouver la réponse pour la question posée, la similarité entre un pair des termes est calculée comme suit :

$$\text{sim}(t_1, t_2) = \text{sim_senses}(t_1, t_2) * \omega(t_1, t_2) \quad (3.14)$$

Donc la valeur de similarité entre t_1 et t_2 est pondérée par le poids associé avec les deux termes. Ceci est pour respecter la différence entre les positions des termes.

Par exemple soit deux concepts $c_1 = \text{"NameOfBest"}$ avec $N_1 = \{\text{Name, Of, Best}\}$ et $c_2 = \text{"BestOfTheName"}$ avec $N_2 = \{\text{Best, Of, The, Name}\}$. L'objective est de calculer la similarité entre "Best" dans N_1 et "Best" dans N_2 donc trouver $\text{sim}(\text{"Best"}, \text{"Best"})$. D'abord, on calcule le poids associé avec ces deux termes comme suit :

$$\omega("Best", "Best") = 1 - \frac{distance("Best", "Best")}{max\ distance} = 1 - \left(\frac{max(3,1) - min(3,1)}{max(|N1|, |N2|) - 1} \right)$$

$$\omega("Best", "Best") = 1 - \left(\frac{3 - 1}{max(3,4) - 1} \right) = 1 - \frac{2}{3} = 0.33$$

Ensuite, on calcule la similarité entre ces deux termes en utilisant le poids trouvé comme suit :

$$sim("Best", "Best") = sim_senses("Best", "Best") * \omega("Best", "Best")$$

$$sim("Best", "Best") = 1 * 0.33 = 0.33$$

Dans le reste de la thèse, nous remplaçons *sim_senses* par la similarité finale suivante qui mesure le degré de similarité entre deux concepts *c1* et *c2* :

$$sim_senses(c1, c2) = \frac{\left(\sum_{t_i \in N_1} max_sim_token(t_i, N_2) + \sum_{t_i \in N_2} max_sim_token(t_i, N_1) \right)}{|N_1| + |N_2|} \quad (3.15)$$

La fonction *max_sim_token* (t_i, N_2) sélectionne la valeur de similarité maximale entre le terme $t_i \in N_1$ et tous les termes dans N_2 (la même chose pour *max_sim_token* (t_i, N_1)). Cette formule combine plusieurs avantages puisqu'elle se base sur les sens des termes, elle respecte les distances entre les termes et elle considère les valeurs maximales des similarités.

Nous présentons ici un exemple de calcul de similarité entre les deux concepts qui sont présentés dans l'exemple précédant. D'abord, il faut trouver les valeurs de *max_sim_token* pour les termes dans N_1 et N_2 :

Pour les termes dans N_1 , nous avons les valeurs suivantes :

- ❖ $max_sim_token(name, ti \in N2) = max(sim(name, best), sim(name, of), sim(name, the), sim(name, name)) = max(0,0,0,0) = 0$ ("name" et "name" ont une distance maximale donc $sim = 0$)
- ❖ $max_sim_token(of, ti \in N2) = max(0,1,0,0) = 1$
- ❖ $max_sim_token(best, ti \in N2) = max(0.33,0,0,0) = 0,33.$

Pour les termes dans N_2 , nous avons les valeurs suivantes :

- ❖ $max_sim_token(best, tj \in N1) = max(sim(best, name), sim(best, of), sim(best, best)) = max(0,0,0.33) = 0.33$
- ❖ $max_sim_token(of, tj \in N1) = max(0,1,0) = 1$

- ❖ $max_sim_token(the, tj \in N1) = max(0,0,0) = 0$
- ❖ $max_sim_token(name, tj \in N1) = max(0,0,0) = 0$

La similarité finale entre les concepts $c1 = "NameOfBest"$ et $c2 = "BestOfTheName"$ est calculé comme suit:

$$Simfinal("NameOfBest", "BestOfTheName") = \frac{(0+1+0.33)+(0.33+1+0+0)}{3+4} = \frac{2.66}{7} = 0.38$$

6. Conclusion du chapitre

Dans ce chapitre, on a présenté notre approche proposée pour l'enrichissement des résultats d'extraction de Dbpedia. On a montré une brève description sur l'extracteur de Dbpedia. Un prétraitement et une nouvelle mesure de similarité basé sens sont illustrés avec des exemples.

Chapitre 4

Implémentation et Evaluation

1. Introduction

Dans ce chapitre nous allons présenter le travail d'implémentation qu'on a fait. On a aussi présenté les évaluations de notre implémentation de mesure de similarité proposée et ainsi que notre enrichissement en utilisant Java Eclipse et plusieurs Java API telle que Jena.

2. Implémentation

L'implémentation consiste à traduire le résultat obtenu lors de l'étape de conception en un programme ou logiciel informatique exécuté sur machine en utilisant les outils de programmation adaptés au problème à traiter.

3. Langage de développement

3.1 Le langage JAVA



Java est un langage de programmation orienté objet développé par Sun Microsystems (aujourd'hui racheté par Oracle)

Le choix du langage JAVA pour le développement de ce projet a été motivé par les points suivants :

- **Portabilité** : En effet un programme développé en java peut s'exécuter sur n'importe quelle machine c'est-à-dire sur celle-ci dispose d'une machine virtuelle java (**JVM** en anglais).
- **Approche objet** : Elle est tout à fait adaptée à la manipulation de plusieurs types de documents, ou structures de données en tant qu'objet (ou même plusieurs versions du format d'un document peuvent être considérées comme des objets).
- **Fiabilité** : Java est très récent (1995), ce qui lui permet d'éviter les écueils de ses concurrents et les nombreux bugs souvent présents dans les anciens langages.
- **Manipulation** : Java permet de manipuler les ontologies OWL à l'aide de l'API Jena.

En effet :

- ✓ Il dispose d'une bibliothèque de classes très riches.
- ✓ Il est robuste: il permet une bonne gestion de mémoire (pas d'accès direct à la mémoire) et des exceptions.

On peut faire de nombreuses sortes de programmes avec Java:

- ✓ Des applications, sous forme de fenêtre ou de console ;
- ✓ Des applets, qui sont des programmes Java incorporés à des pages web ;
- ✓ Des applications pour appareils mobiles, avec J2ME ;
- ✓ et bien d'autres ! J2EE, JMF, J3D pour la 3D.(*memoire.pdf*, s. d.)

Pour notre projet, on a utilisé Java 8.

4. L'outil de développement

4.1 Eclipse

Eclipse est un EDI¹⁶, ou en anglais un IDE¹⁷. Développé par IBM à partir de ses ancêtres Visual Age et Visual Age For Java, il a depuis été rendu open-source et son évolution est maintenant gérée par la Fondation. Sa conception est complètement modulaire : basée sur des notions telles que "micro noyau OSGi¹⁸" (depuis la version 3) ou encore les plug-ins, ce qui fait d'Eclipse une boîte à outils facilement modifiable et extensible.

Eclipse est également devenu une plate-forme (Framework pour manipuler modèle et méta modèle), servant de socle à d'autres applications, que celles-ci soient destinées au développement logiciel ou à tout autre domaine. Cette plate-forme est nommée Eclipse RCP¹⁹.

Eclipse a pour objectif de produire et de fournir des outils pour la réalisation de logiciels, il été conçu uniquement pour produire des environnements de développement, cependant les utilisateurs et les programmeurs se sont rapidement mis à réutiliser ses briques logicielles pour des applications clientes.

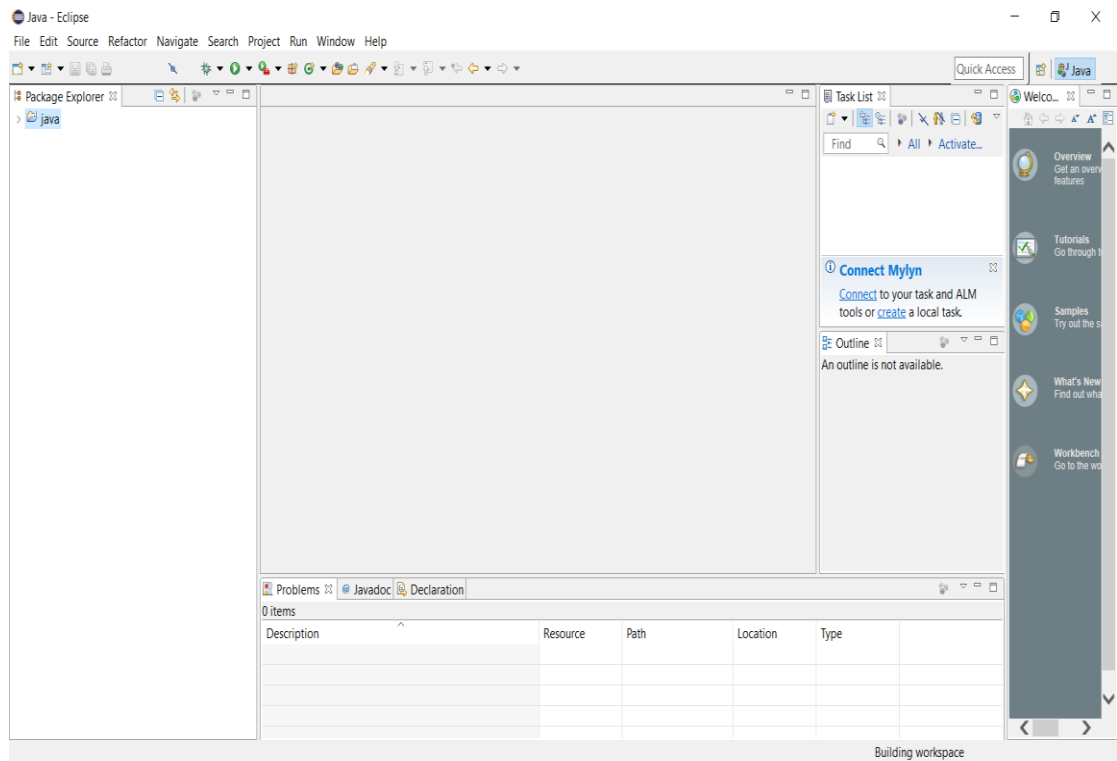


FIGURE 4.10 : FENETRE PRINCIPALE D'ECLIPSE 3.3 .

4.2 JENA

Jena est disponible à l'adresse : <http://jena.sourceforge.net/> .

Jena est un ensemble d'outils (une API) open source développé par *HP Labs semantic Web programme* ,permettant de lire et de manipuler des ontologies décrites en RDFS ou en OWL et d'y appliquer certains mécanismes d'inférences.(*Memoire Online - Conception d'une ontologie pour une plate forme d'enseignement à distance - Saloua & Amina Chettibi & Rouibah, s. d.*)

5. les étapes d'exportation d'un page Wikipédia vers fichier XML:

- 1) Entrez leurs titres dans la boîte de texte, à raison d'un titre par ligne.
- 2) Sélectionnez si vous désirez la version actuelle avec toutes les anciennes versions, avec les lignes de l'historique de la page ou simplement la page actuelle avec des informations sur la dernière modification.
- 3) Dans ce dernier cas vous pouvez aussi utiliser un lien, tel que Spécial: Exporter/Wikipédia: Accueil principal pour la page Wikipédia: Accueil principal.

6. L'organigramme

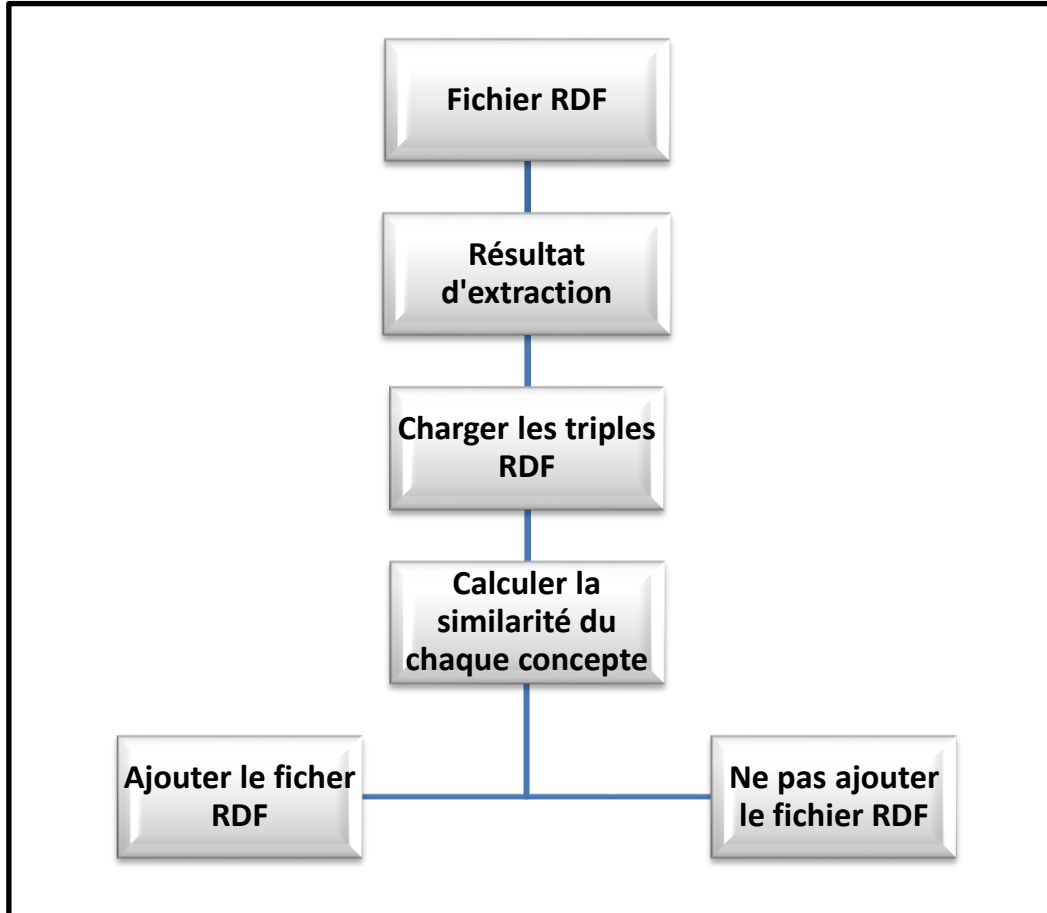


FIGURE 4.11 : L'ORGANIGRAMME DE NOTRE SYSTEME.

Jena charge le fichier RDF(les triples), ensuite charge ligne par ligne, chaque triple contient (sujet, prédicat, objet), nous concentrons sur l'objet. Nous comparons entre le texte et la base de connaissances, si S'ils sont similaires, nous ajoutons le fichier RDF dans lequel laisser Sujet et le Prédicat comme ils sont et changent l'Objet, s'ils ne sont pas similaires ne pas ajouter le fichier RDF.

7. L'évaluation

7.1 L'évaluation de la mesure de similarité proposée

Le tableau 4.1 présente des valeurs des similarités qui sont calculées entre des paires de concepts. Ces valeurs sont trouvées en utilisant *sim1*, *sim2* et *sim_senses* qui sont basées sur les sens des concepts. La mesure de *Levenshtein* est aussi utilisée dont des valeurs de similarité non significatives sont trouvées. Par exemple, la mesure *Levenshtein* indique

Implémentation et Evaluation

que « help » et « aide » sont non similaires avec une valeur de 0.86. Ceci n'est pas correct puisque ces deux concepts ont des sens communs. Depuis ce tableau, des valeurs significatives sont calculées avec les mesures basées sur les sens depuis WordNet. Comme « help » et « aide » où la valeur trouvée est 0.47.

<i>Concept1</i>	<i>Concept2</i>	<i>sim1</i>	<i>sim2</i>	<i>sim_senses</i>	<i>Levenshtein</i>
<i>Car</i>	<i>Machine</i>	0.25	0.13	0.14	0.14
<i>Car</i>	<i>Automobile</i>	0.20	0.08	0.33	0.0
<i>Teacher</i>	<i>Instructor</i>	0.50	0.17	0.75	0.30
<i>Begin</i>	<i>Start</i>	1.0	0.50	0.32	0.00
<i>Test</i>	<i>Trial</i>	0.45	0.18	0.30	0.20
<i>Category</i>	<i>Family</i>	0.43	0.18	0.30	0.25
<i>Help</i>	<i>Aid</i>	0.50	0.11	0.47	0.0
<i>Location</i>	<i>Placement</i>	0.67	0.28	0.25	0.22
<i>End</i>	<i>Terminate</i>	0.33	0.16	0.45	0.22
<i>dress_clothes</i>	<i>clean_clothes</i>	0.75	0.16	0.50	0.60
<i>Publish</i>	<i>Write</i>	1	1	0.11	0.40
<i>edition_home</i>	<i>publisher_house</i>	0.33	0.08	0.03	0.35
<i>Web</i>	<i>Network</i>	0.07	0.03	0.12	0.14

Tableau 4.1 : Des valeurs de similarité calculées avec *sim1*, *sim2*, *sim_senses* et *Levenshtein*

7.2 L'évaluation d'enrichissement

Implémentation et Evaluation

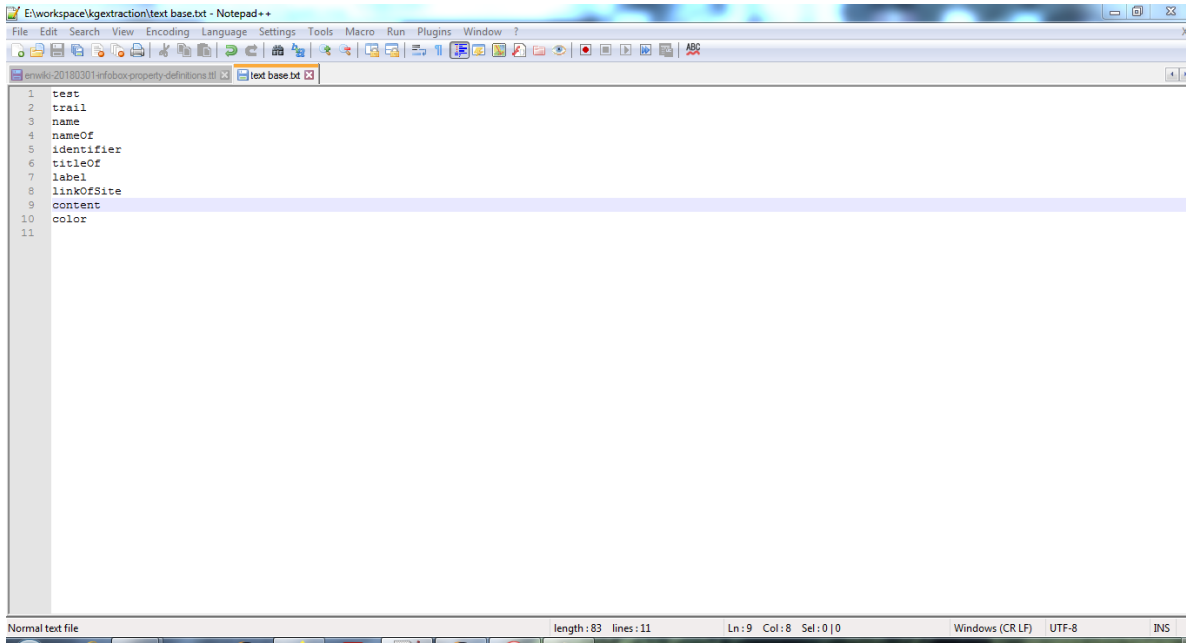
```
1 # started 2020-08-17T21:49:49Z
2 <http://dbpedia.org/property/date> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .
3 <http://dbpedia.org/property/date> <http://www.w3.org/2000/01/rdf-schema#label> "date"@en .
4 <http://dbpedia.org/property/itemcount> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .
5 <http://dbpedia.org/property/itemcount> <http://www.w3.org/2000/01/rdf-schema#label> "item count"@en .
6 <http://dbpedia.org/property/title> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .
7 <http://dbpedia.org/property/title> <http://www.w3.org/2000/01/rdf-schema#label> "title"@en .
8 <http://dbpedia.org/property/listclass> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .
9 <http://dbpedia.org/property/listclass> <http://www.w3.org/2000/01/rdf-schema#label> "listclass"@en .
10 <http://dbpedia.org/property/group> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .
11 <http://dbpedia.org/property/group> <http://www.w3.org/2000/01/rdf-schema#label> "group"@en .
12 <http://dbpedia.org/property/list> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .
13 <http://dbpedia.org/property/list> <http://www.w3.org/2000/01/rdf-schema#label> "list"@en .
14 <http://dbpedia.org/property/sisterportal> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .
15 <http://dbpedia.org/property/sisterportal> <http://www.w3.org/2000/01/rdf-schema#label> "sisterportal"@en .
16 <http://dbpedia.org/property/sisterprojectcolor> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .
17 <http://dbpedia.org/property/sisterprojectcolor> <http://www.w3.org/2000/01/rdf-schema#label> "sisterprojectcolor"@en .
18 <http://dbpedia.org/property/sisterproject> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .
19 <http://dbpedia.org/property/sisterproject> <http://www.w3.org/2000/01/rdf-schema#label> "sisterproject"@en .
20 <http://dbpedia.org/property/contributors> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .
21 <http://dbpedia.org/property/contributors> <http://www.w3.org/2000/01/rdf-schema#label> "contributors"@en .
22 <http://dbpedia.org/property/maincontent> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .
23 <http://dbpedia.org/property/maincontent> <http://www.w3.org/2000/01/rdf-schema#label> "maincontent"@en .
24 <http://dbpedia.org/property/property> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .
25 <http://dbpedia.org/property/property> <http://www.w3.org/2000/01/rdf-schema#label> "property"@en .
26 <http://dbpedia.org/property/sitelink> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .
27 <http://dbpedia.org/property/sitelink> <http://www.w3.org/2000/01/rdf-schema#label> "sitelink"@en .
28 <http://dbpedia.org/property/323> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .
29 <http://dbpedia.org/property/323> <http://www.w3.org/2000/01/rdf-schema#label> "323"@en .
30 <http://dbpedia.org/property/wikiproject> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .
31 <http://dbpedia.org/property/wikiproject> <http://www.w3.org/2000/01/rdf-schema#label> "WikiProject"@en .
32 <http://dbpedia.org/property/textcolor> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .
33 <http://dbpedia.org/property/textcolor> <http://www.w3.org/2000/01/rdf-schema#label> "textcolor"@en .
34 <http://dbpedia.org/property/description> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .
35 <http://dbpedia.org/property/description> <http://www.w3.org/2000/01/rdf-schema#label> "description"@en .
36 <http://dbpedia.org/property/hiddenbutton> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .
37 <http://dbpedia.org/property/hiddenbutton> <http://www.w3.org/2000/01/rdf-schema#label> "hiddenbutton"@en .
38 <http://dbpedia.org/property/name> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> <http://www.w3.org/1999/02/22-rdf-syntax-ns#Property> .
```

FIGURE 4.12 : EXEMPLE DE RESULTAT D'EXTRACTION DE L'EXTRACTEUR DBPEDIA.

```
5
6
7 public class SubWord
8 {
9     public static ArrayList ar1;
10    public static ArrayList ar2;
11    public SubWord()
12    {
13    }
14    public static String adapturi(String word){
15        String[]w=word.split("");
16        String[]l=w[w.length-1].split(".html");
17        return l[l.length-1];
18    }
19 }
20 public static String adaptproc(String word){
21     String[]w=word.split("");
22     String[]l=w[w.length-1].split(".html");
23     if (w.length>1)return w[w.length-2]+"_"+l[l.length-1];
24     else return l[l.length-1];
25 }
26 public static String adaptpre(String word){
27     String[]w=word.split("");
28     String[]l=w[w.length-1].split(".html");
29     return adapts(l[l.length-1]);
30 }
31 }
```

FIGURE 4.13 : LA CLASSE JAVA QUI FAIT LE PRETRAITEMENT.

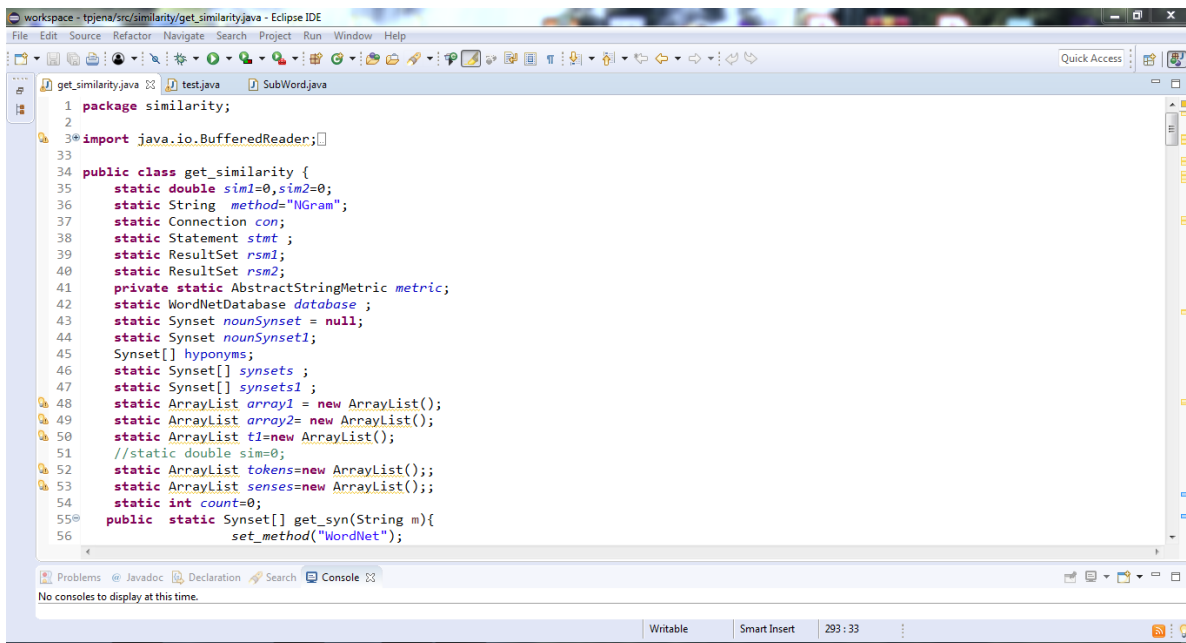
Implémentation et Evaluation



```
1 test
2 trail
3 name
4 nameOf
5 identifier
6 titleOf
7 label
8 linkOfSite
9 content
10 color
11
```

FIGURE 4.14 : LA BASE DE TEXTES.

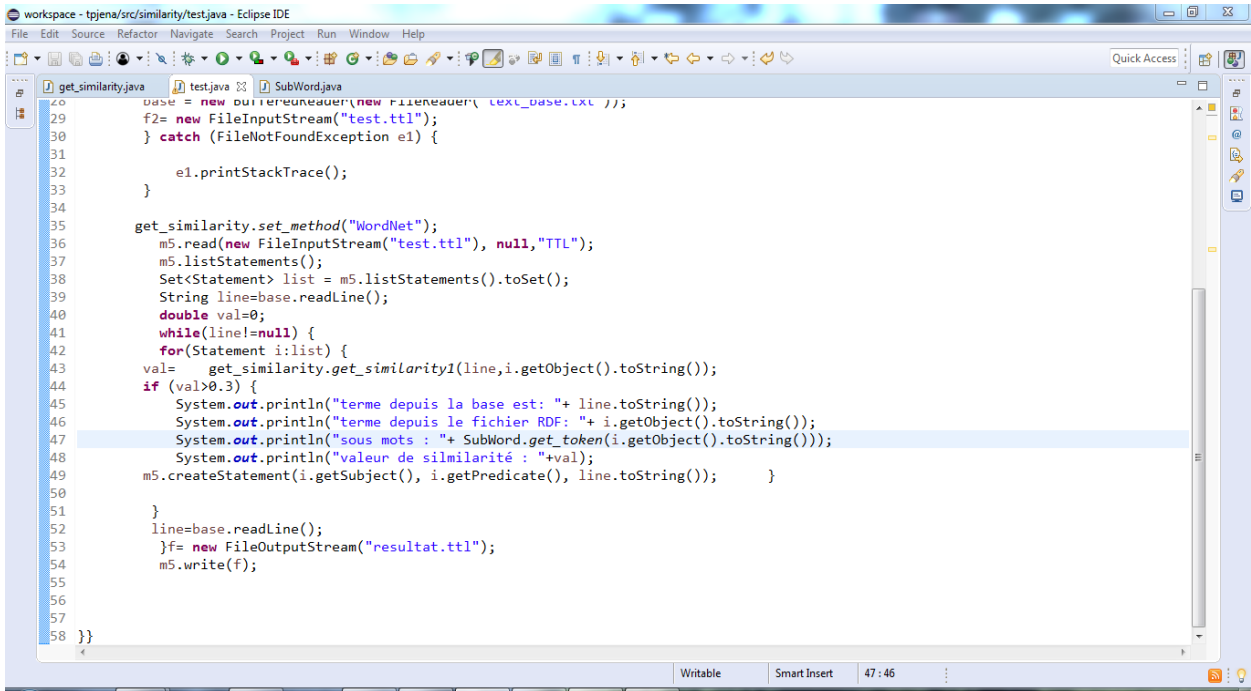
Ce fichier est utilisé pour calculer la similarité entre les littérales dans le résultat et les termes dans ce fichier.



```
1 package similarity;
2
3 import java.io.BufferedReader;
4
5 public class get_similarity {
6     static double sim1=0,sim2=0;
7     static String method="NGram";
8     static Connection con;
9     static Statement stmt ;
10    static ResultSet rsm1;
11    static ResultSet rsm2;
12    private static AbstractStringMetric metric;
13    static WordNetDatabase database ;
14    static Synset nounSynset = null;
15    static Synset nounSynset1;
16    Synset[] hyponyms;
17    static Synset[] synsets ;
18    static Synset[] synsets1 ;
19    static ArrayList array1 = new ArrayList();
20    static ArrayList array2= new ArrayList();
21    static ArrayList t1=new ArrayList();
22    //static double sim=0;
23    static ArrayList tokens=new ArrayList();;
24    static ArrayList senses=new ArrayList();;
25    static int count=0;
26    public static Synset[] get_syn(String m){
27        set_method("WordNet");
28    }
29 }
```

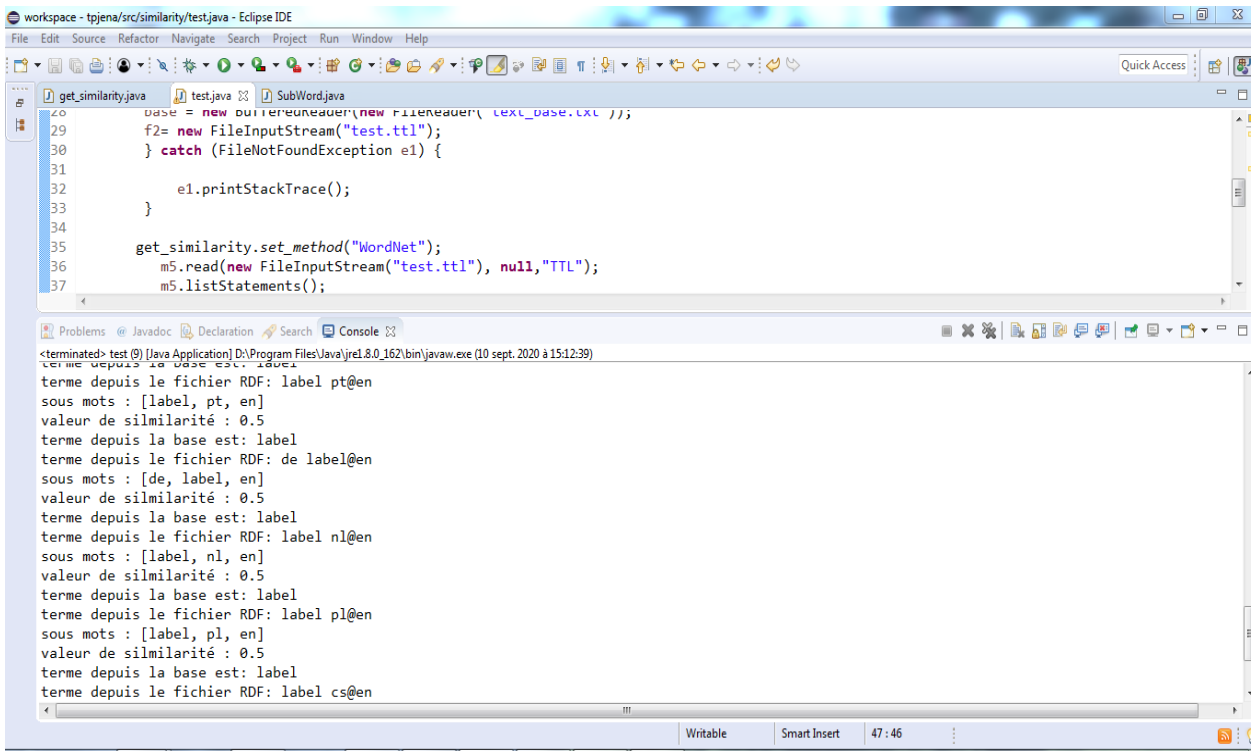
FIGURE 4.15 : LA CLASSE JAVA POUR LA MESURE DE SIMILARITE.

Implémentation et Evaluation



```
workspace - tpjena/src/similarity/test.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
get_similarity.java test.java SubWord.java
20 base = new BufferedReader(new FileReader("text_base.txt"));
29 f2= new FileInputStream("test.ttl");
30 } catch (FileNotFoundException e1) {
31
32     e1.printStackTrace();
33 }
34
35 get_similarity.set_method("WordNet");
36 m5.read(new FileInputStream("test.ttl"), null,"TTL");
37 m5.listStatements();
38 Set<Statement> list = m5.listStatements().toSet();
39 String line=base.readLine();
40 double val=0;
41 while(line!=null) {
42     for(Statement i:list) {
43         val= get_similarity.get_similarityI(line,i.getObject().toString());
44         if (val>0.3) {
45             System.out.println("terme depuis la base est: "+ line.toString());
46             System.out.println("terme depuis le fichier RDF: "+ i.getObject().toString());
47             System.out.println("sous mots : "+ SubWord.get_token(i.getObject().toString()));
48             System.out.println("valeur de silmilarité : "+val);
49             m5.createStatement(i.getSubject(), i.getPredicate(), line.toString());
50         }
51     }
52     line=base.readLine();
53 }f= new FileOutputStream("resultat.ttl");
54 m5.write(f);
55
56
57
58 }}
```

FIGURE 4.16 : LA CLASSE JAVA PRINCIPALE D'ENRICHISSEMENT

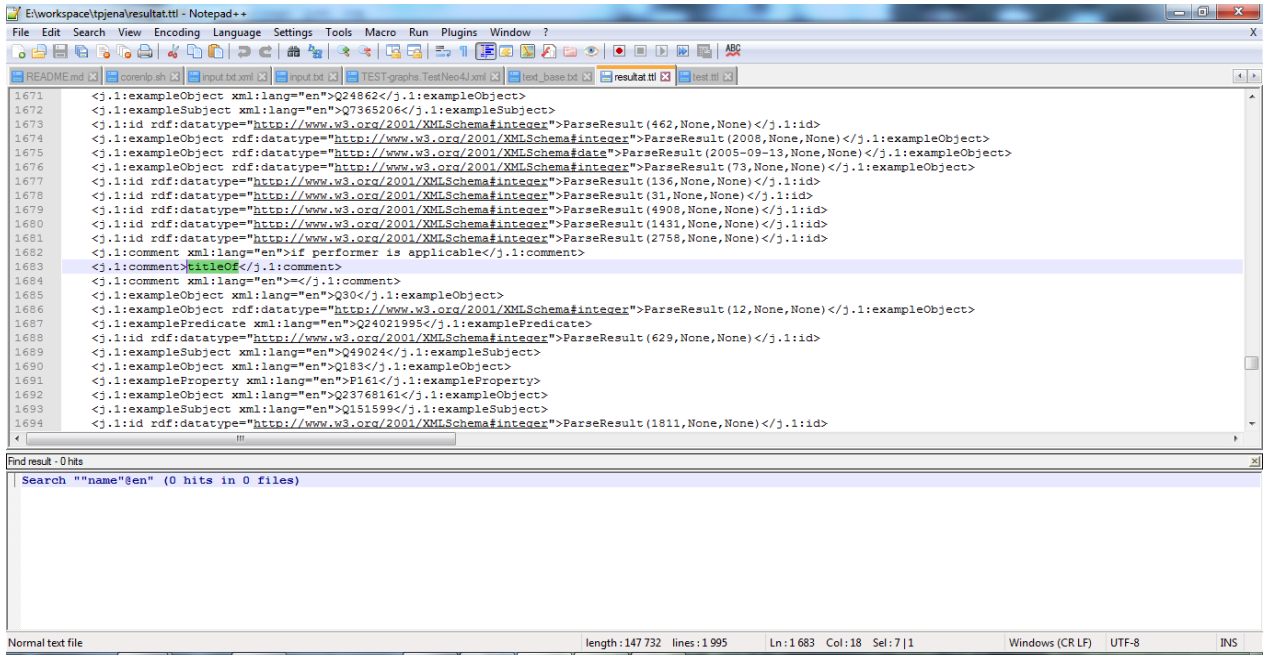


```
workspace - tpjena/src/similarity/test.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
get_similarity.java test.java SubWord.java
20 base = new BufferedReader(new FileReader("text_base.txt"));
29 f2= new FileInputStream("test.ttl");
30 } catch (FileNotFoundException e1) {
31
32     e1.printStackTrace();
33 }
34
35 get_similarity.set_method("WordNet");
36 m5.read(new FileInputStream("test.ttl"), null,"TTL");
37 m5.listStatements();

Problems Javadoc Declaration Search Console
<terminated> test (9) [Java Application] D:\Program Files\Java\jre1.8.0_162\bin\javaw.exe (10 sept. 2020 à 15:12:39)
terme depuis la base est: label
terme depuis le fichier RDF: label pt@en
sous mots : [label, pt, en]
valeur de silmilarité : 0.5
terme depuis la base est: label
terme depuis le fichier RDF: de label@en
sous mots : [de, label, en]
valeur de silmilarité : 0.5
terme depuis la base est: label
terme depuis le fichier RDF: label nl@en
sous mots : [label, nl, en]
valeur de silmilarité : 0.5
terme depuis la base est: label
terme depuis le fichier RDF: label pl@en
sous mots : [label, pl, en]
valeur de silmilarité : 0.5
terme depuis la base est: label
terme depuis le fichier RDF: label cs@en
```

FIGURE 4.17 : L'EXECUTION DE L'ENRICHISSEMENT

Implémentation et Evaluation



```
1671 <j.1:exampleObject xml:lang="en">Q24862</j.1:exampleObject>
1672 <j.1:exampleSubject xml:lang="en">Q7365206</j.1:exampleSubject>
1673 <j.1:id rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">ParseResult (462, None, None)</j.1:id>
1674 <j.1:exampleObject rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">ParseResult (2008, None, None)</j.1:exampleObject>
1675 <j.1:exampleObject rdf:datatype="http://www.w3.org/2001/XMLSchema#date">ParseResult (2005-09-13, None, None)</j.1:exampleObject>
1676 <j.1:exampleObject rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">ParseResult (73, None, None)</j.1:exampleObject>
1677 <j.1:id rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">ParseResult (136, None, None)</j.1:id>
1678 <j.1:id rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">ParseResult (31, None, None)</j.1:id>
1679 <j.1:id rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">ParseResult (4908, None, None)</j.1:id>
1680 <j.1:id rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">ParseResult (1431, None, None)</j.1:id>
1681 <j.1:id rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">ParseResult (2758, None, None)</j.1:id>
1682 <j.1:comment xml:lang="en">if performer is applicable</j.1:comment>
1683 <j.1:comment>titleOf</j.1:comment>
1684 <j.1:comment xml:lang="en"></j.1:comment>
1685 <j.1:exampleObject xml:lang="en">Q30</j.1:exampleObject>
1686 <j.1:exampleObject rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">ParseResult (12, None, None)</j.1:exampleObject>
1687 <j.1:examplePredicate xml:lang="en">Q24021995</j.1:examplePredicate>
1688 <j.1:id rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">ParseResult (629, None, None)</j.1:id>
1689 <j.1:exampleSubject xml:lang="en">Q49024</j.1:exampleSubject>
1690 <j.1:exampleObject xml:lang="en">Q183</j.1:exampleObject>
1691 <j.1:exampleProperty xml:lang="en">P161</j.1:exampleProperty>
1692 <j.1:exampleObject xml:lang="en">Q23768161</j.1:exampleObject>
1693 <j.1:exampleSubject xml:lang="en">Q151599</j.1:exampleSubject>
1694 <j.1:id rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">ParseResult (1811, None, None)</j.1:id>
```

Find result - 0 hits
Search ""name"@en" (0 hits in 0 files)

Normal text file length:147732 lines:1995 Ln:1.683 Col:18 Sel:7|1 Windows (CR LF) UTF-8 INS

FIGURE 4.18 : LE FICHER RDF QUI COMPORTE LE RESULTAT DE L'ENRICHISSEMENT

8. Conclusion du chapitre

Dans ce chapitre, nous avons présenté l'implémentation de notre approche proposée. On a montré aussi l'évaluation de cette implémentation. On a trouvé que la mesure proposée est achevée de trouver des termes similaires avec les autres termes dans les résultats de l'entasseur de Dbpedia.

Conclusion Générale

Conclusion générale

Nous avons présenté dans ce travail une approche pour l'enrichissement des fichiers RDF extraits depuis Wikipédia en utilisant une nouvelle mesure de similarité. On a utilisé aussi un processus de prétraitement pour simplifier les termes dans les triples afin de faciliter le calcul de similarité.

L'objectif principal de notre travail est de simplifier le processus de recherche des connaissances dans les fichiers RDF par ajouter des termes similaires qui peuvent être utilisés dans les requêtes Sparql.

Travaux futures :

Dans une future direction, on va essayer d'intégrer le calcul de similarité dans l'extracteur de Dbpedia pour gagner de temps et améliorer le processus d'extraction.

Références bibliographiques

[Aboulhamid & Rousseau, 2018] Aboulhamid, E. M., & Rousseau, F. (2018). *System Level Design with .Net Technology*. CRC Press.

[Augenstein et al., 2012] Augenstein, I., Padó, S., & Rudolph, S. (2012). LODifier : Generating Linked Data from Unstructured Text. In E. Simperl, P. Cimiano, A. Polleres, O. Corcho, & V. Presutti (Éds.), *The Semantic Web : Research and Applications* (Vol. 7295, p. 210-224). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-30284-8_21

[Exner & Nugues, 2012] Exner, P., & Nugues, P. (2012). *Entity Extraction : From Unstructured Text to DBpedia RDF Triples*. 906.

Insight into Semantic Web and why is it important today. (s. d.). Consulté 5 juin 2020, à l'adresse <https://medium.com/@yassine.hammar1/inight-into-semantic-web-and-why-its-the-next-technological-revolution-24521cec4459>

[Kejriwal, 2019] Kejriwal, M. (2019). *Domain-Specific Knowledge Graph Construction*. Springer.

[Kertkeidkachorn & Ichise, 2017] Kertkeidkachorn, N., & Ichise, R. (2017). T2KG : An End-to-End System for Creating Knowledge Graph from Unstructured Text. *AAAI Workshops*.

Memoire Online—Conception d'une ontologie pour une plate forme d'enseignement à distance—Saloua & Amina Chettibi & Rouibah. (s. d.). Consulté 1 septembre 2020, à l'adresse https://www.memoireonline.com/05/08/1145/m_conception-ontologie-plate-forme-enseignement-a-distance11.html

Memoire.pdf. (s. d.). Consulté 10 septembre 2020, à l'adresse <http://dspace.univ-guelma.dz:8080/xmlui/bitstream/handle/123456789/1475/memoire.pdf?sequence=1&isAllowed=y>

Références bibliographiques

[Fillmore, 1976] Fillmore, C.J. (1976), FRAME SEMANTICS AND THE NATURE OF LANGUAGE*. *Annals of the New York Academy of Sciences*, 280: 20-32. doi:[10.1111/j.1749-6632.1976.tb25467.x](https://doi.org/10.1111/j.1749-6632.1976.tb25467.x)

[Morsey et al., 2012a] Morsey, M., Lehmann, J., Auer, S., Stadler, C., & Hellmann, S. (2012a). DBpedia and the live extraction of structured data from Wikipedia. *Program*, 46(2), 157-181. <https://doi.org/10.1108/00330331211221828>

[Morsey et al., 2012b] Morsey, M., Lehmann, J., Auer, S., Stadler, C., & Hellmann, S. (2012b). DBpedia and the live extraction of structured data from Wikipedia. *Program: electronic library and information systems*, 46, 157 - 181. <https://doi.org/10.1108/00330331211221828>

[Rizzo, 2012] Rizzo, G. (2012). *Knowledge extraction from unstructured data and classification through distributed ontologies*. <https://doi.org/10.6092/polito/porto/2497074>

[Gruber, 1993] Gruber, Thomas R. (1993). A translation approach to portable ontology specifications (PDF). *Knowledge Acquisition* 5 (2), pp.199–220.

[Fellah et al., 2008] Fellah, A., Malki, M., & Zahaf, A. (2008). Alignement des ontologies : utilisation de WordNet et une nouvelle mesure structurelle [ontologies Alignment : using WordNet and a new structural measure]. *Conférence en Recherche d'Information et Applications* (pp. 401-408). Trégastel, France, Université de Renne.

[Djeddai et al., 2013] Djeddai, A., Seridi, H., & Khadir, T. (2013). A Query Approximating Approach over RDF Graphs. *International Journal of Information Technology and Web Engineering*, 8(4), 65-87, October-December 2013, DOI : 10.4018/ijitwe.2013100105, IGI global Publisher, USA.

[Suchanek et al., 2008] Suchanek, F. M., Kasneci, G., & Weikum, G. (2008). YAGO : A Large Ontology from Wikipedia and WordNet. *Journal of Web Semantic* 6(3) : 203-217, doi :10.1016/j.websem.2008.06.001.

Références bibliographiques

[Fellbaum, 1998] Fellbaum, C. (1998). *WordNet, an electronic lexical database*. Cambridge, Massachusetts : the MIT Press.