UNIVERSITÉ SAAD DAHLEB DE BLIDA1

Faculté des sciences

Département d'informatique

MEMOIRE DE MASTER

En Informatique

Option : Ingénierie Des Logiciels

THÈME :

Catégorisation sémantique des données liées.

Réalisé par

ZEFFOUNI Khaled Cherif

TAKARLI Mohamed Nadhir

Encadré par

Dr. M. FAREH

Dr. I. RIALI

Présidente Dr. N.LAHIANI

Examinateur Dr. M.HAMMOUDA

Juin 2023

Remerciements

Nous remercions ALLAH de nous avoir donné la santé et le courage afin de pouvoir réussir ce travail. Ce travail est l'aboutissement d'un long cheminement au cours duquel nous avons bénéficié d'encadrement, des encouragements et du soutien de plusieurs personnes, à qui nous tenons à dire profondément et sincèrement merci.

En premier lieu, nous adressons nos remerciements, les plus vifs, notre profonde gratitude et notre respects à notre promotrice Mme. FAREH d'avoir accepté de nous encadrer, pour les conseils et orientations tant précieux qu'elle nous a prodigué durant ce travail. Nous la remercie aussi vivement pour la démarche fructueuse qu'elle a adoptée pour nous introduire dans ce fabuleux domaine

Nous remercions aussi notre Co-promoteur M. RIALI Ishak qui nous a guidé dans notre travail et nous a aidé à trouver des solutions pour avancer. Nous présentons tous nos respects et nos remerciements aux membres du jury qui ont accepté d'évaluer ce travail.

Nos remerciements les plus sincères à toutes les personnes qui auront contribué de près ou de loin à l'élaboration de ce mémoire ainsi qu'à la réussite de cette formidable année universitaire

Résumé

Le web contient un grand nombre de données liées. La catégorisation automatique de ces données fournit un avantage à tous types de personnes ou logiciels cherchant à les exploiter.

La plupart de ces données sont difficilement accessible avec une probabilité très faible d'obtenir celles pertinentes. Le défi est de les structurer ce qui permettra d'extraire les connaissances pertinentes et pouvoir ainsi préparer des raisonnements et exploiter ces données.

Nous avons développé un système de catégorisation sémantique des données liées dans le but de contribuer à résoudre ce problème. Après avoir extrait les connaissances à partir de ces données, notre système calcule les distances sémantiques entre les différentes ressources. Ces distances sont ensuite classées en trois catégories, la distance terminologique, structurelle et extensionnelle. En utilisant ces distances sémantiques, nous réduisons la dimensionnalité des ressources, puis nous appliquons un algorithme de catégorisation automatique à base de densité pour obtenir des clusters de ressources ainsi que des points de bruit. Pour démontrer l'efficacité de notre système, nous avons réalisé une évaluation de notre système.

Mots clés :

RDF (Ressource Description Framework), Web de données, Données liées, Catégorisation, Mesure de similarité, Sémantique.

Abstract

The web contains a vast amount of linked data. Automatic categorisation of this data provides an advantage to all types of people or software seeking to exploit it.

Most of this data is difficult to access, with a very low probability of obtaining the relevant information. The challenge is to structure it in such a way as to extract the relevant knowledge and thus be able to prepare reasoning and exploit the data.

We have developed a semantic categorisation system for linked data in order to help solve this problem. Once the knowledge has been extracted from the data, our system calculates the semantic distances between the different resources. These distances are then classified into three categories : terminological, structural and extensional. Using these semantic distances, we reduce the dimensionality of the resources and then apply an automatic density-based categorisation algorithm to obtain resource clusters and noise points. To demonstrate the effectiveness of our system, we carried out an evaluation of our system.

Key words :

RDF (Ressource Description Framework), Data web, Linked data, Categorisation, Similarity measurement, Semantic.

ملخص

يحتوي الويب على كمية كبيرة من البيانات المرتبطة. يوفر التصنيف التلقائي لهذه البيانات ميزة لجميع أنواع الأشخاص أو البرامج التي تسعى لاستغلالها.

يصعب الوصول إلى معظم هذه البيانات مع احتمال منخفض جدًا للحصول على البيانات ذات الصلة. يكمن التحدي في هيكلتها مما يجعل من الممكن استخراج المعرفة ذات الصلة وبالتالي تكون قادرة على إعداد المنطق والاستدلالات.

لقد قمنا بتطوير نظام تصنيف دلالي للبيانات المرتبطة من أجل المساعدة في حل هذه المشكلة. بعد استخراج المعرفة من هذه البيانات ، يحسب نظامنا المسافات الدلالية بين الموارد المختلفة. ثم يتم تصنيف هذه المسافات إلى ثلاث فئات ، المسافة الاصطلاحية والهيكلية والامتدادية. باستخدام هذه المسافات الدلالية ، نقوم بتقليل أبعاد الموارد، ثم نطبق خوارزمية تصنيف تلقائية قائمة على الكثافة للحصول على مجموعات الموارد وكذلك نقاط الضوضاء. لإثبات فعالية نظامنا ، قمنا بتقييم نظامنا.

الكلمات الدالة:

إطار وصف الموارد ، شبكة البيانات ، البيانات المرتبطة ، التصنيف ، قياس التشابه ، الدلالات.

Table des matières

Table des figures

Liste des tableaux

INTRODUCTION GÉNÉRALE

Contexte

Au niveau du web, un nouvel essor s'est récemment développé autour du web sémantique et des données ouvertes et liées. Le web sémantique tend à promouvoir l'utilisation de formats de données qui facilitent le partage, la réutilisation, le traitement par des machines et qui permettent de produire de nouvelles connaissances grâce au raisonnement. Les données liées sont une méthode de publication des données qui favorisent le traitement automatisé et l'établissement de relations vers d'autres sources de données.

Le web de données permet de publier des données structurées et non structurées sur le web, non pas sous la forme de silos de données isolés les uns des autres, mais en les reliant pour constituer un réseau d'informations global. Les données liées visent à partager et à interconnecter des données structurées sur le web selon les principes des données liées, sous forme d'une représentation lisible par la machine. L'intérêt de construire un jeu de données liées, c'est lorsqu'elles entretiennent des liens avec d'autres données, ce qui permet d'étoffer les descriptions.

Problématique

L'émergence de nombreuses sources de données interconnectées, physiquement distribuées et gérées de manière autonome s'élève à la croissance rapide du cloud Linked Open Data, qui offre des opportunités, pour la modélisation et l'exploitation de ces données.

Les informations issues du web de données sont de grande taille, une énorme quantité d'informations est disponible sous différentes formes, structurée et non structurée. Malgré les efforts de la communauté du web sémantique pour fournir des techniques permettant de relier des entités entre des sources de connaissances, il est nécessaire de combler le fossé entre les données structurées du web sémantique et les contenus du web.

Malheureusement la plupart des informations sont difficilement accessible avec une probabilité très faible d'obtenir celles pertinentes. Il devient alors primordial de mettre les techniques qui permettent de décrire, de structurer, et de rechercher de telles informations, afin d'augmenter

le niveau de compréhension des humains sur les ressources disponibles sur le Web, mais aussi interprétables par des machines. Le problème relève du fait que le système et les utilisateurs doivent partager le même sens des données.

Structurer cette quantité de données est nécessaire pour pouvoir extraire les connaissances pertinentes, ce qui permettra de mieux connaître ces données pour pouvoir préparer des raisonnements et des inférences. La catégorisation des données liées représente une tâche importante pour organiser et structurer les données liées, offrant ainsi un outil automatique capturant à la fois les données et la sémantique. La difficulté est comment prendre en considération les critères liés au contexte des données liées dans le processus de catégorisation sémantique.

Objectif

L'objectif de ce projet consiste à concevoir et réaliser un système de catégorisation de données liées dans le contexte du web sémantique. En effet, la catégorisation des données liées est importante pour la bonne exploitation des données. Notre objectif est d'extraire les données pertinentes à partir du fichier RDF, de quantifier la différence entre les données et de les catégoriser.

Organisation du mémoire

Afin d'atteindre le but de notre travail, l'organisation de notre mémoire sera comme suit :

— **Chapitre 1 : Catégorisation des données du web sémantique**
Dans ce chapitre, nous avons abordé une étude sur le web de données, son historique et les différents types de données. Nous avons parlé des données liées, en présentant son architecture et les principes de données liées. Nous avons présenté aussi la catégorisation avec quelques techniques, les mesures de similarité et les travaux existants avec leur analyse.

— **Chapitre 2 : Conception du système**
Dans ce chapitre, nous présentons notre solution proposée pour la catégorisation, en détaillant les trois phases avec chaque étape, commençant par l'extraction des données jusqu'à l'affichage du résultat de la catégorisation.

— **Chapitre 3 : Implémentions et test du système**
Ce chapitre présente l'implémentation de la solution proposée. Nous présentons les différents outils utilisés, les différentes interfaces ainsi que des exemples d'exécution du système. Nous aborderons aussi l'évaluation du système en utilisant une métrique d'évaluation et la représentation graphique.

La conclusion de ce mémoire synthétise les principales étapes de construction de notre système, et dégage quelques perspectives de ce travail.

Chapitre 1

CATÉGORISATION DES DONNÉES DU WEB SÉMANTIQUE

1.1 Introduction

Le World Wide Web est la pierre angulaire de la vie moderne. Il est utilisé à tous les niveaux de la société, qu'il s'agisse de consulter les réseaux sociaux, de faire des achats, de travailler à distance ou de procéder à des transactions entre sociétés multinationales. Il influence tous les aspects de la vie sociale, culturelle et économique.

Le web a vu le jour sous la forme du web statique à la fin des années 1980 et s'est rapidement consolidé en tant que géant culturel et économique avec l'avènement du web interactif à la fin des années 2000. Actuellement, le WWW progresse vers sa prochaine itération grâce au web sémantique, qui est une collection de ressources web appelées données liées [22].

Le Web de données, aussi appelé le web sémantique ou web 3.0 représente la prochaine évolution majeure dans la connexion et la représentation de l'information. Il permet de relier des données d'une ressource à n'importe quelle autre ressource et d'être lisible par des ordinateurs afin qu'ils puissent effectuer automatiquement des tâches de plus en plus sophistiquées [22].

La catégorisation automatique des données, aussi appelée clustering, appartient au domaine de l'apprentissage automatique qui est une branche de l'intelligence artificielle. Elle consiste à catégoriser les éléments d'un dataset selon leurs niveaux de similarité. Les données présentant des caractéristiques similaires sont regroupées ensemble dans ce qu'on appelle des clusters.

Ce chapitre est organisé comme suit : nous commencerons par un aperçu du web et de son histoire, puis nous passerons au web sémantique, sa définition et ses composantes, notamment les ontologies. Nous aborderons également les données liées, avec des exemples de leur utilisation. Ensuite, nous présenterons la catégorisation des données, ainsi que quelques travaux sur la catégorisation des données liées. Enfin, nous terminerons par une analyse de l'état de l'art.

1.2 Évolution du web

Le World Wide Web communément défini comme la partie de l'internet qui est accessible par un navigateur est défini par la W3C (World Wide Web Consortium) comme un espace d'information dans laquel les ressources sont identifiées par des identifiants globaux appeler "Uniform resource identifier (URI)".

— **Le Web 1.0** était la première implémentation du web, qui a duré de 1989 à 2005. Il a été défini comme une toile de connexion d'informations. Selon Tim Berners-Lee le web était "read-only". Il permettait très peu d'interactivité où les utilisateurs pouvaient échanger entre eux (peer-to-peer) mais il était impossible d'interagir avec un site web [8].

— **Le Web 2.0** est la seconde génération du web, défini en 2004 par Dale Dougherty comme un web "read-write".

Tim O'Reilly définit le web 2.0 comme suit "Web 2.0 is the business revolution in the computer industry caused by the move to the internet as a platform, and an attempt to understand the rules for success on that new platform. Chief among those rules is this. Build applications that harness network effects to get better the more people use them." Le web 2.0 introduit le concept de plateforme ou les utilisateurs peuvent interagir et modifier les sites web [8].

— **Le Web 3.0** a été inventé par John mark en 2006. L'idée de base étant de définir les données de structure (métadonnées) et de les lier pour optimiser la recherche, l'automatisation, l'intégration, et la réutilisation à travers différentes utilisations.

Le Web 3.0 est aussi appelé web sémantique. Le web sémantique a été inventé par Tim Berners-Lee, inventeur du world wide web. Puis amélioré, étendu, et standardisé par la W3C, des langages, publications, et des outils ont déjà été développés [8].

Web 1.0	Web 2.0	Web 3.0
1996 - 2004	2004 - 2016	2016+
Le web hypertexte	Le web social	Le web sémantique
Tim Berners-Lee	Tim Berners-Lee, Dale Dougherty	Tim Berners-Lee
Lecture seule	Web de lecture et d'écriture	Web exécutable
Millions d'utilisateurs	Des milliards d'utilisateurs	Plus de Milliards d'utilisateurs
Unidirectionnel	Bi-directionnel	Environnement virtuel multi-utilisateurs

TABLE 1.1 – Comparison of Web 1.0, Web 2.0 and Web 3.0 [8].

1.3 Web sémantique

Aussi appelé web de données est un projet du W3C dirigé par Tim Berners Lee. C'est un réseau de données liées où chaque ressource est liée à travers le web aux autres ressources. Les metadata du web sémantique sont représentées par un graphe sous format RDF ou une ontologie. De plus, le format RDF est lisible par machine ce qui rend l'exploitation automatique de ces données plus simple et directe [4].

1.4 Une approche en couches du web sémantique

La construction d'un web sémantique se fait en plusieurs étapes, chaque étape construisant une couche au-dessus de la précédente. La construction d'une couche doit respecter deux principes de downward compatibility, ou un logiciel utilisant une couche doit pouvoir interpréter les informations provenant des couches inférieures. Et upward compatibility, ou un logiciel utilisant une couche doit avoir au moins une interprétation partielle des couches supérieures [8]. Cette architecture en couches est souvent représenté par un schéma illustré dans la figure 1.1.

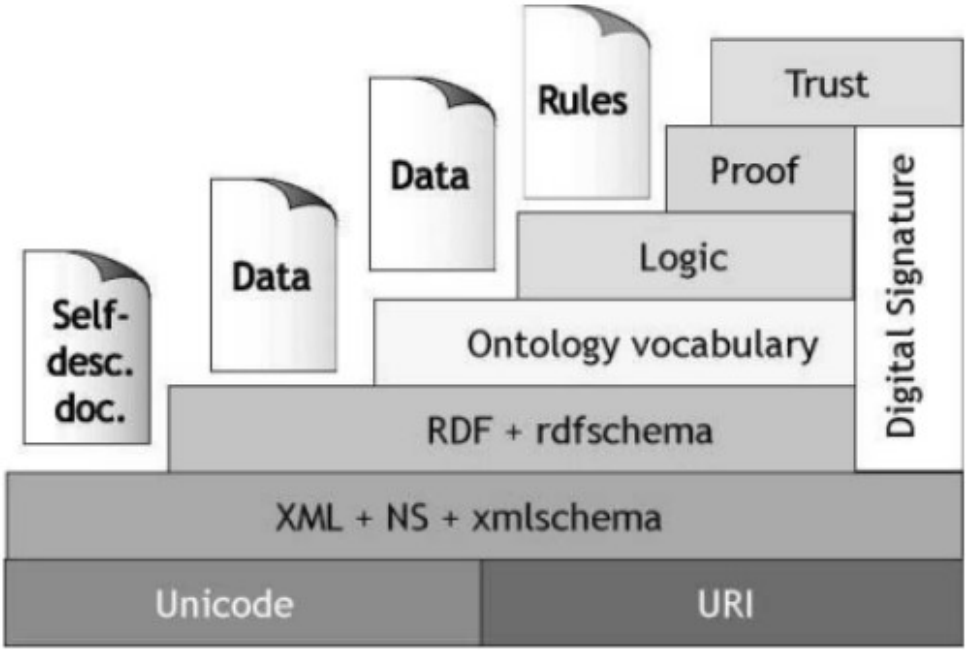


FIGURE 1.1 – Semantic Web Layered Architecture [8].

I. **Unicode et URI**

Unicode combine différents jeux de caractères tels que les jeux de caractères arabes, japonais, cyrilliques et latins et devient ainsi la norme de codage internationale de base pour les données du Web sémantique. Unicode attribue à chaque caractère son propre numéro, qui peut être interprété plus facilement et sans erreur par les ordinateurs. Cette affectation numérique rend Unicode indépendant du système, du programme et de la

langue. URI (Uniform Resource Identifier) est une chaîne de caractères qui identifie une ressource abstraite ou physique. Une telle source pourrait être des documents, des images, des fichiers téléchargeables ou similaires sur le Web ou des appareils physiques tels que des téléviseurs, des réfrigérateurs ou des radios. L'URI expose ces différentes sources et leur donne une adresse unique. L'URI utilise une variété de schémas de nommage et de méthodes d'accès différents tels que HTTP ou FTP. La sous-catégorie d'identificateurs de ressources uniformes (URI) qui est probablement la mieux connue des profanes est l'URL (Uniform Resource Locator), qui est utilisée pour adresser et identifier des sites web sur internet [1].

II. **XML**

Extensible Markup Language est un langage de balisage utilisé pour construire une structure arbitraire d'un document. En effet le vocabulaire et la combinaison des balises ne sont pas statiques, mais peuvent être définis selon l'application du XML [11].

Le XML a été dérivé du SGML (Standard Generalized Markup Language) un standard international utilisé pour la représentation de données indépendamment d'un système ou matériel qui est lisible par l'humain et la machine [1].

III. **RDF**

Resource Description Framework, est un standard du W3C qui est un modèle de données basic utilisé pour écrire des descriptions simples sur des objets web et un langage d'expression de graphes de données dirigé sous forme de triplets **sujet, prédicat, objet**, respectivement **ressource, propriété, valeur** (figure 1.2) [2].

— **Une ressource (sujet)** : est identifiable par une IRI (Internationalized Resource Identifier).

— **Une propriété (prédicat)** : c'est une relation entre un sujet et un objet.

— **Une Valeur (objet)** : peut être un sujet ou un littéral ou d'autre instance de triplets (Sujet, Prédicat, Objet).

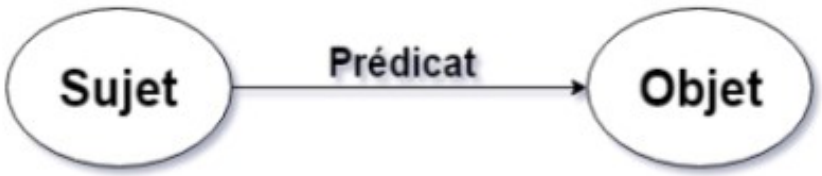


FIGURE 1.2 – Triplet sujet, prédicat, objet

1. https ://semantisches-web.net/technologien/unicode-und-uri/
2. https ://www.w3.org/TR/rdf12-concepts/

La figure 1.3 illustre un example de structure RDF.

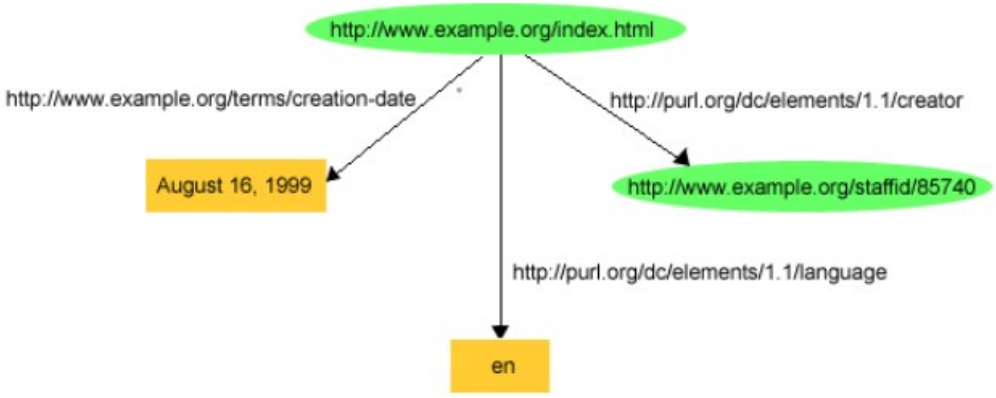


FIGURE 1.3 – Exemple d'une structure RDF [3]

Les déclarations RDF peuvent être incorporées dans des pages Web Hypertext Markup Language ou stockées dans des fichiers séparés et liées à des données dans le contenu Web. Lorsque RDF a été spécifié pour la première fois, les déclarations RDF étaient incorporées dans les documents XML liés au contenu Web. Bien que la syntaxe XML ait été la seule option de syntaxe spécifiée au départ, les normes d'encodage des déclarations RDF incluent désormais les trois syntaxes suivantes :

— Turtle : est la syntaxe de texte la plus populaire pour les déclarations RDF. Le W3C le décrit comme une "forme de texte compacte et naturelle" qui inclut des abréviations pour les modèles couramment utilisés.

— JSON-LD : utilise la syntaxe JSON pour les instructions RDF.

— N-Triples : est un sous-ensemble de la syntaxe Turtle, conçu pour être un format texte plus simple pour les instructions RDF afin d'améliorer la facilité d'utilisation par les humains qui écrivent des instructions. Ce format permet également aux programmes de créer et d'analyser plus facilement les déclarations RDF.

Un langage de requête RDF est utilisé pour accéder et gérer les informations stockées dans les graphes RDF. Les langages de requête RDF doivent être capables d'analyser les triplets RDF et d'interpréter et de produire des résultats liés au contenu des triplets, ainsi qu'aux relations entre les triplets [4].

IV. **RDFS**

Le RDF Schema est une extension sémantique du RDF. Il fournit des mécanismes pour décrire des groupes de ressources connexes et des relations entre ces ressources. Ces ressources sont utilisées pour déterminer des caractéristiques d'autres ressources. Le système

3. https ://www.w3.org/TR/rdf-primer/
4. https ://www.w3.org/TR/rdf12-concepts/

de classes et de propriétés du RDF Schéma est similaire au languages de programmation orienté objet, mais la différence réside dans les propriétés de ces instances. Le RDF Schéma décrit les propriétés en fonction des classes de ressources auxquelles elles s'appliquent. C'est le rôle du domaine (domain) et codomain (range). Le vocabulaire core défini par le RDF Schema est appelé RDFS.

Il est identifié par l'IRI suivant :

"http ://www.w3.org/2000/01/rdf-schema#"

Ce namespace définit plusieurs entités dont (rdfs :class), (rdfs :subclass), (rdfs :domain), et (rdfs :range). Ces entités sont utilisées pour modéliser des classes et des propriétés avec une restriction du domaine et de portée, cependant il ne permet pas d'exprimer l'exclusion et la négation , et il limite l'axiomatisation aux restrictions, pour cela il est considéré comme un langage ontologique simple [10]. La figure 1.4 illustre un exemple d'un schéma RDFS.

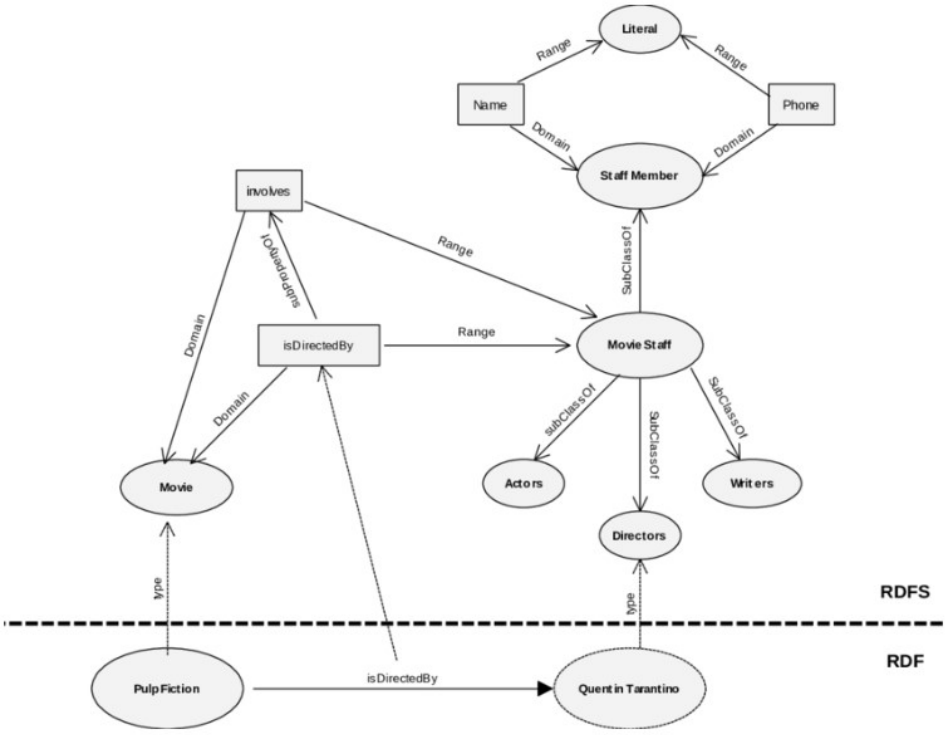


FIGURE 1.4 – Exemple d'un RDFS [10].

V. Ontologie

Ce terme est souvent utilisé pour définir une structure de connaissance sémantique partagée par des individus du même domaine. L'ontologie sémantique peut être une conceptualisation informelle construite à partir de concepts et leurs relations définies en langage naturel. Ou elle peut être construite de manière formelle en utilisant un langage logique. Dans le contexte du web une ontologie est une structure concrète, syntaxique qui modélise les liens sémantiques d'un domaine dans un langage lisible par machine [20].

a) **Les types d'ontologies**

— Les ontologies de haut niveau ou générique : elles décrivent des concepts très généraux comme l'espace, le temps, la matière, les objets, les événements, les actions, etc.

— Les ontologies de domaine : limitées à la représentation de concepts dans des domaines donnés (géographie, médecine, écologie, etc.) et qui spécialisent les concepts de l'ontologie de haut niveau.

— Les ontologies de tâches : ce type d'ontologies décrit le vocabulaire concernant une tâche générique (ex. : enseigner, diagnostiquer...), notamment en spécialisant les concepts d'une ontologie de haut niveau. Certains auteurs emploient le nom « ontologie du domaine de la tâche » pour faire référence à ce type d'ontologie.

— Les ontologies d'application : elles décrivent des concepts dépendant à la fois d'un domaine et d'une tâche particuliers dans ce domaine. Ces concepts correspondent souvent aux rôles joués par des entités.

— Les ontologies de représentation : elles conceptualisent les primitives des langages de représentation des ontologies des connaissances.

b) **Composants d'une ontologie**

— **Les concepts** : aussi appelés classes de l'ontologie, constituent les objets de base manipulés par les ontologies. Ils correspondent aux abstractions pertinentes du domaine du problème, retenues en fonction des objectifs que l'on se donne et de l'application envisagée pour l'ontologie, par exemple, la description d'un ensemble d'objets, d'une tâche, d'une fonction, d'une stratégie, d'un processus de raisonnement, etc [31].

Le concept peut être décomposé en trois éléments distincts (figure 1.5).

a. Le terme : Est un élément lexicale qui permet d'exprimer le concept en langue naturelle.

b. L'intention : contient la sémantique du concept, exprimée en termes de propriétés et attributs, contraintes.

c. L'extension : regroupe les objets manipulés à travers le concept [17].

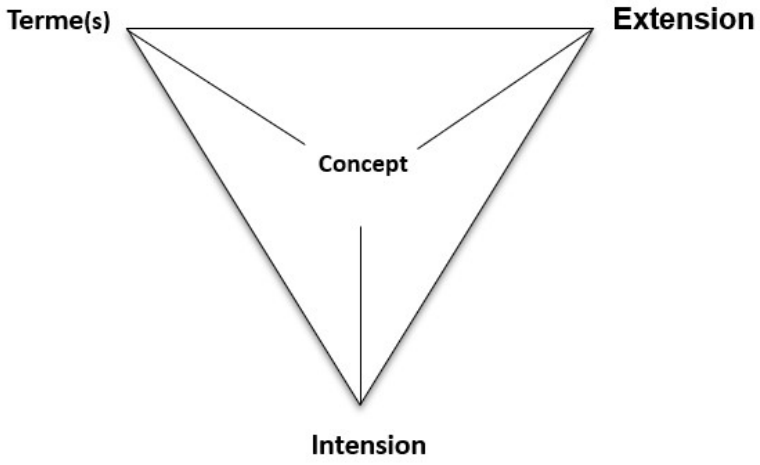


FIGURE 1.5 – Triangle sémantique

— **Les relations** : traduisent les interactions existant entre les concepts présents dans le domaine ciblé. Ces relations sont formellement définies comme tout sous ensemble d'un produit cartésien de n ensembles, c'est à dire R : C1 x C2x . . . x Cn
La relation de spécialisation (subsomption).
La relation de composition (méronymie).
La relation d'instanciation, etc.
Ces relations nous permettent de capturer, la structuration ainsi que l'interaction entre les concepts, ce qui permet de représenter une grande partie de la sémantique de l'ontologie [18].

— **Les axiomes** : permettent de modéliser des assertions toujours vraies, à propos des abstractions du domaine traduites par l'ontologie. Ils permettent de combiner des concepts, des relations et des fonctions pour définir des règles d'inférence et qui peuvent intervenir, par exemple, dans la déduction, la définition des concepts et des relations, ou alors pour restreindre les valeurs des propriétés ou les arguments d'une relation.

— **Les fonctions** : sont des cas particuliers de relations dans lesquelles le n-ième élément (extrant) de la relation est défini de manière unique à partir des n-1 éléments précédents (intrants).
Formellement,les fonctions sont définies ainsi : F : C1 x C2 . . . x Cn-1 $\rightarrow$Cn.

— **Les instances** : Ou individus constituent la définition extensionnelle de l'ontologie. Ils représentent des éléments singuliers véhiculant les connaissances à propos du domaine du problème.

VI. **OWL**

OWL (Web Ontology Language) est un langage de balisage sémantique utilisé pour la publication et le partage d'ontologies sur le World Wide Web. OWL est une extension du vocabulaire du RDF et est dérivée de DAML+OIL. OWL se décline en plusieurs variantes avec différents niveaux d'expressivité (OWL Lite, OWL Description Logic, OWL Full) [25].

— OWL Lite a été conçu pour une implémentation facile, fournissant une partie fonctionnelle de OWL aux utilisateurs.

— OWL DL a été conçu pour être compatible avec l'industrie de description logique et fournir un langage avec des propriétés de calcul pour les systèmes de raisonnement. OWL Full relâche quelques contraintes appliquées sur OWL DL pour offrire des caractéristiques qui peuvent être utiles pour une variété de base de données et représentation de connaissances, mais qui causent une incompatibilité avec les raisonneurs logiques.

— OWL Full et OWL DL supportent le même ensemble de structures de langage OWL. Leur différence dans les restrictions appliquées à l'utilisation de ses fonctionnalités et les fonctionnalités RDF.

— OWL Full permet de mélanger librement le OWL avec le RDF Schema et ne force pas une séparation stricte des classes,des propriétés,des individus et des valeurs. OWL DL met en place des restrictions sur l'incorporation du RDF et requiert une ségrégation des classes, des propriétés, des individus et des valeurs. OWL Lite est un sous-langage d'OWL DL qui ne supporte que partiellement la spécification OWL. OWL Lite se conforme aux mêmes restrictions sémantiques qu'OWL DL, ce qui vous permet d'avoir certaines caractéristiques désirables pour les moteurs de raisonnement. [3].

OWL est une extension du vocabulaire RDF. Ainsi, n'importe quel graphe RDF construit une ontologie OWL complète. De plus, le sens attribué à un graphe RDF par OWL inclut le sens attribué au graphe par RDF. Par conséquent, les ontologies OWL Full incluent le contenu RDF arbitraire, qui est traité de manière cohérente avec RDF. OWL associe un sens supplémentaire à certains triplets RDF.

Il existe de nombreux éditeurs d'ontologie en langage OWL, parmi eux

a) Protégé est l'un des éditeurs d'ontologies open source les plus connus, son architecture plug-in le rend polyvalent.

b) NeOn Toolkit est un éditeur d'ontologie open source, multi-plateforme, avec prise en charge de plug-in. Il est particulièrement adapté à des projets de grande envergure.

c) OWLGrEd est un éditeur d'ontologie visuelle propriétaire avec intégration cloud.

VII. **SPARQL**

SPARQL (SPARQL Protocol And RDF Query Language) est un langage de requête pour des structures RDF publié par le W3C. Il a été conçu en 2004 par le RDF Data Access Working Group, puis est devenu une recommandation du W3C en janvier 2008.

SPARQL est utilisé pour interagir avec des structures de données nativement sous format RDF ou vues comme RDF grâce à un middleware [28].

Exemple sur une requête simple :

PREFIX foaf : <http ://xmlns.com/foaf/0.1/>
SELECT ?name
WHERE
?person foaf :name ?name .

VIII. **Couche logique (logic)**

Est utilisée pour améliorer les langages de l'ontologie et pour permettre l'utilisation de connaissances spécifiques dans l'application [1].

IX. **Couche de preuve (proof)**

Implique le processus de déduction ainsi que la représentation de preuve dans les langages web et la validation de preuves [1].

X. **Couche de confiance (trust)**

Elle émerge grâce à la signature digitale et à d'autres types de connaissances basées sur des recommandations par des logiciels éprouvés ou des évaluations et certifications par des agences et des organismes de consommateurs. Étant au sommet de la pyramide, la confiance est un concept important de haut niveau : le web ne peut atteindre son plein potentiel que lorsque les utilisateurs ont confiance en ces opérations (sécurité) et en la qualité de ses informations [1].

1.5 Données liées

Les données liées consistent simplement à utiliser le web pour créer des liens entre des données provenant de différentes sources et qu'elles soient lisibles par machine.

Berners-Lee a mis en place un ensemble de règles visant à créer un espace web global qui relie toutes les données publiées. Utiliser des URI pour identifier toute chose sur le web. Utiliser des URI HTTP pour que les choses sur le web soit accessibles. Fournir avec les URI des informations exploitables, lisible par les humains et les machines, en utilisant des formats standards (RDF SPARQL). Inclure des liens vers d'autres URI pour permettre la découverte d'autres choses sur le web [5].

1.6 Les applications du web sémantique

Cette section présente un bref aperçu des applications actuelles du web sémantique.

1.6.1 Recherche dans le web sémantique

La recherche d'information est l'utilisation la plus fréquente du web et des moteurs de recherche très puissants ont été développés pour répondre à ce besoin. Cependant, chercher dans un web non indexé sémantiquement présente certains désavantages :

Comment un moteur de recherche peut-il mapper une requête sur des documents où des informations sont disponibles, mais ne récupère pas des informations intelligentes et significatives ?

Les résultats de la requête produits par les moteurs de recherche sont répartis sur différents documents qui peuvent être connectés par un lien hypertexte. Comment un moteur de recherche peut-il reconnaître efficacement des résultats aussi volumineux et désorganisés ?

Le web sémantique offre une solution aux deux problèmes en utilisant des annotations sémantiques et des modèles de requête basés sur des graphes [24].

1.6.2 E-commerce

Le commerce électronique doit permettre un échange plus fluide d'informations et de transactions entre tous les acteurs économiques, depuis l'offreur de produits ou services jusqu'aux clients finaux. On distingue usuellement deux scénarios : des offreurs aux clients (B2C – Business-to-Customer) et entre offreurs et grossistes (B2B – Business-toBusiness). Les applications du B2C permettent aux offreurs de produits et services de propager et présenter leurs offres, et aux clients, de trouver et de commander l'offre sélectionnée. En fournissant un accès unique à une large collection d'articles ou de services fréquemment mise à jour, une place de commerce électronique facilite la rencontre entre l'offre et la demande grâce à des outils de médiation commerciale. Les applications du B2B ont une plus longue histoire et utilisent les échanges informatisés via des structures de messages et de protocoles très codifiées, pré-établies et normalisées (EDI – Electronic Data Interchange ou Échange de Données Informatisés) récemment assouplies via des standards basées sur XML (eXtensible Markup Language) [23].

1.6.3 Applications médicales

La médecine est un des domaines d'applications privilégiés du Web sémantique comme elle l'a été, à une autre époque, des techniques de l'Intelligence artificielle, en particulier les systèmes experts. En effet c'est un domaine complexe où les informations à partager sont nombreuses et où il n'y a pas ou peu de solutions algorithmiques à ce partage comme à l'usage des connaissances, en particulier cliniques. Ainsi, un des principaux mécanismes du Web sémantique qui consiste à décrire les ressources via des annotations, revêt une importance capitale en bio-informatique, en particulier en ce qui concerne le partage des ressources génomiques.

Dans le contexte, plus ancien, de la recherche d'information, la médecine a une longue tradition de développement de thésaurus comme le MeSH (Medical Subject Heading) ou UMLS (Unified Medical Language System – http ://www.nlm.nih.gov/research/umls/umlsmain.html) et les utilise maintenant dans le cadre des mécanismes du Web sémantique. Enfin, et plus récemment, les services Web proposent des solutions à la problématique récurrente et non résolue de l'interopérabilité en médecine, en particulier dans le contexte des systèmes d'information hospitaliers (SIH) [23].

1.7 Catégorisation

La catégorisation consiste à découvrir les classes qui existent naturellement au sein d'un dataset basé sur des critères de similarité. Les méthodes de catégorisation on deux directives principales, les éléments d'une même classe doivent se ressembler le plus possible et les éléments de deux classes différentes doivent être aussi différents que possible.

1.7.1 La catégorisation hiérarchique

La catégorisation Hiérarchique est une méthode qui génère un arbre de clusters, aussi appelé dendrogramme, en utilisant des techniques de division et d'agglomération heuristiques. Les algorithmes qui utilisent la division sont appelés divisifs et ceux qui utilisent l'agglomération sont appelés agglomératifs [21].

Le calcul de la matrice de distance a chaque pas de cet algorithme le rend inadapté à la catégorisation de grand nombre de données. Le dendrogramme est découpé pour obtenir la catégorisation finale, le niveau du découpage détermine le nombre de clusters.

1.7.2 La catégorisation partitionnelle

Les algorithmes de catégorisation partitionnelle divisent les données en un nombre de clusters prédéfini. Grâce à leurs avantages, ils sont mieux adaptés pour la reconnaissance de motifs. Voici quelques exemples d'algorithmes partitionnels itératifs [7] :

1. K-means : est un algorithme de partitionnement itératif qui maximise la distance intra-cluster. Les points sont assignés aux clusters dont le centroïde est le plus proche. Les centroïdes sont des points virtuels représentant le centre du cluster, ils sont assignés aléatoirement pour la première itération.

2. Fuzzy C-means : est une version de K-means qui introduit la notion de flou, elle est mieux adaptée à une utilisation réelle car elle gère le chevauchement entre les clusters en utilisant des clusters flous.

3. K-harmonic means : est une méthode de catégorisation floue qui regroupe les données de telle manière que la somme des moyennes harmoniques des distances entre chaque entité et tous les centroïdes de cluster soit minimisée.
4. K-medoids : est une méthode de partitionnement qui répond au désavantage de k-means en utilisant des objets représentatifs au lieu de la moyenne de distance entre les éléments du cluster.

1.7.3 La catégorisation à base de densité

Ces algorithmes utilisent la densité des données dans une région. Si la densité dépasse un seuil, alors les données sont regroupées dans un cluster.

DBSCAN est un algorithme de catégorisation basé sur la densité. Selon DBSCAN, les zones à haute densité représentent des clusters et les zones à faible densité représentent les aberrations et le bruit. Il prend deux paramètres en entrée, le nombre de points minimum pour créer un cluster, appelé MinPts, et la distance maximum dans laquelle deux points sont considérés comme voisins, appelée epsilon. Ces deux paramètres sont utilisés pour créer des clusters à partir de zones à haute densité et pour ajouter un point à un cluster existant. Cet algorithme est particulièrement adapté pour traiter de grands ensembles de données, avec du bruit, et est capable d'identifier des clusters de tailles et de formes différentes [15].

1.8 Mesures de similarité

La mesure de similarité est un concept utilisé pour quantifier la similarité ou la proximité entre deux objets, éléments ou ensembles de données. Elle est largement utilisée dans divers domaines, tels que l'informatique, l'apprentissage automatique, la recherche d'information, la bioinformatique, etc.

La mesure de similarité vise à évaluer à quel point deux objets sont semblables ou comparables les uns aux autres. Elle repose généralement sur une métrique ou une fonction qui attribue une valeur numérique à cette similarité. Plus cette valeur est élevée, plus les objets sont considérés comme similaires, tandis qu'une valeur faible indique une faible similarité.

Il existe différentes approches pour mesurer la similarité, en fonction du domaine d'application et des types de données considérés. Certaines mesures de similarité couramment utilisées incluent [26] :

— **Similarité cosinus :** mesure la similarité entre deux vecteurs en utilisant le cosinus de l'angle entre eux.

— **Distance euclidienne :** mesure la distance entre deux points dans un espace euclidien.

— **Coefficient de Jaccard :** utilisé pour mesurer la similarité entre ensembles d'éléments.

Les mesures de similarité sont nécessaires pour la classification automatique basée sur l'apprentissage non supervisé, car ces algorithmes utilisent la similarité ou la distance entre les éléments pour décider dans quel cluster un élément est placé, voire le nombre de clusters. Dans notre cas, nous devons calculer la similarité sémantique pour pouvoir catégoriser des ressources RDF. Ils existent plusieurs types de similarité :

— **La similarité terminologique :** consiste à déterminer la similarité entre les termes de deux ressources. Elle est composée de deux parties, la similarité lexicale utilise des ressources externes pour déterminer la similarité des termes, tandis que la similarité syntaxique utilise le contenu des termes pour déterminer leur degré de similitude.

— **La similarité structurelle :** détermine le degré de similarité de deux ressources par rapport a leurs prédicats.

— **La similarité extensionelle :** mesure la similarité de deux ressources selon les objets de leur prédicat, aussi appelé extension.

1.9 Catégorisation dans le contexte des données liées

La catégorisation des données est un problème aussi ancien que la représentation des données elle-même, et la représentation RDF est identique à cet égard. En ce qui concerne la catégorisation de données RDF avec un apprentissage non supervisé, nous pouvons trouver les travaux suivants.

1.9.1 Travaux existants

1. **Christodoulou, K., Paton, N. W., and Fernandes, A. A. (2013, March). Structure inference for linked data sources using clustering :**

   Le but de ce travail est l'inférence de schémas à partir de dataset RDF. Pour parvenir au schéma, une catégorisation est nécessaire afin de regrouper les données en fonction d'une mesure structurelle.

   La distance entre les points de données est calculée dans une matrice de similarité utilisant la mesure de Jaccard. Cette distance est utilisée par un algorithme de catégorisation hiérarchique agglomératif, ainsi qu'une découpe en clusters en fonction du coefficient de silhouette de chaque cluster [9].

2. **Guo, Q., Ji, W., Zhong, S., and Zhou, E. (2013, March). The Analysis of the Ontology-based K-Means Clustering Algorithm :**

   Le travail de Guo et al. (2013) présente une analyse de l'algorithme de catégorisation K-Means basé sur l'ontologie. Le contexte de l'étude est l'analyse de grands ensembles de données complexes.

Les résultats montrent que l'algorithme K-Means basé sur l'ontologie améliore les performances de l'algorithme K-Means standard en termes de précision et de temps d'exécution. Il est également capable de traiter des ensembles de données plus complexes avec une meilleure précision [19].

3. **Qi, L., Lin, H. T., and Honavar, V. (2013, April). Clustering remote RDF data using SPARQL update queries :**

   Ce travail explore la problématique de la scalabilité dans le domaine du catégorisation des données liées. La taille des datasets de données liées a tendance à être très grande, c'est pourquoi ce travail s'intéresse à utiliser des requêtes SPARQL pour effectuer une catégorisation avec deux algorithmes différents : K-means et MROC. Les résultats montrent une amélioration du temps d'exécution avec K-means et une différence négligeable avec MROC [29].

4. **Du, Q., Dong, Z., Huang, C., and Ren, F. (2016). Density-based clustering with geographical background constraints using a semantic expression model :**

   Ce travail utilise une ontologie pour classe les biens immobiliers selon leur localisation géographique en prenant en compte les liens sémantiques extraits des cartes qui ont été utilisées comme datasets.

   La classification se fait avec une technique basée sur la densité. L'algorithme utilisé s'appelle C-DBSCANO, qui est une variante de DBSCAN avec des contraintes et une inclusion d'une ontologie. Ce travail est limité par une approche spécifique pour le domaine géographique et une dépendance ontologique, ce qui rendra difficile l'application de leur approche. En effet, créer une ontologie pour un domaine d'application afin de capturer la sémantique du domaine et les contraintes associées n'est pas une tâche facile. [12].

5. **Bouhamoum, R., Kellou-Menouer, K., Lopes, S., and Kedad, Z. (2018, April). Scaling up schema discovery for RDF datasets. :**

   Le but de ce travail est l'inférence de schéma à partir de données à grande échelle. La catégorisation des données étant une étape nécessaire de l'inférence de schéma, une technique de catégorisation à base de densité a été choisie. Une version adaptée à un grand nombre de données de l'algorithme DBSCAN avec une mesure de Jaccard a été utilisée [6].

6. **Eddamiri, S., and Benghabrit, A. (2019). An improved RDF data clustering algorithm :**

   Le travail intitulée "An Improved RDF Data Clustering Algorithm" étudie les données RDF qui sont utilisées pour représenter des informations sur le Web. L'objectif de ce travail est d'améliorer l'algorithme de catégorisation pour mieux organiser ces données.

Les auteurs ont proposé un algorithme de catégorisation basé sur la similarité entre les propriétés des entités RDF. Cet algorithme utilise différentes techniques dans le contexte d'une approche de catégorisation hiérarchique qui permet de regrouper les entités similaires dans des clusters. Ils ont également proposé une méthode de sélection automatique de la valeur de seuil de similarité pour déterminer la taille des clusters [13].

7. **Bamatraf, S. A., and BinThalab, R. A. (2019, December). Clustering RDF data using K-medoids :**

   Dans ce travail, les données RDF ont été prétraitées à l'aide de plusieurs algorithmes pour extraire une matrice de similarité basée sur des mesures syntaxiques et lexicales. L'algorithme K-médoïdes est ensuite utilisé pour classifier les données. Cette méthode offre de bons résultats avec un taux de précision au-dessus de 90 pour cent [2].

1.9.2 Critères de comparaison

Dans cette partie, nous examinerons différents travaux existants en les comparant selon plusieurs critères, et nous présenterons notre synthèse (table 1.9.2).

1. **Type de la mesure de similarité**

   La mesure de similarité est un critère permettant de faire la comparaison entre les travaux mentionnés en se basant sur les mesures terminologique (lexicale et syntaxique), structurelle, et extensionnelle.

2. **Type de catégorisation**

   Ce critère représente les types de catégorisation non supervisée, tels que la catégorisation hiérarchique, partitionnelle, basée sur la densité et basée sur la grille.

3. **Algorithme utilisé**

   Chaque catégorie de catégorisation non supervisée comprend plusieurs algorithmes comme k-means et k-medoids pour la catégorisation partitionnel, ou DBSCAN et SNN pour la catégorisation basée sur la densité.

4. **Formule de similarité**

   Il existe différentes métriques de similarité utilisées pour le calcul des distances tels que le coefficient de Jaccard, la similarité cosinus, la distance euclidienne, la corrélation de Pearson, la distance de Levenshtein et la distance de Jaro-Winkler.

5. **Dataset utilisé**

   Le choix du dataset utilisé pour tester le travail informe le domaine du travail ainsi que sa généricité.

Travaux	Type Similarité	Type Catégorisation	Algorithme	Formule	Dataset
Christodoulou et al. 2013	Structurelle	hierarchique	Aglomeratif	Jaccard	Jamendo and Magnatune
Guo et al. 2013	/	Partitionnel	K-means	Cosine	Movielens
Qi et al. 2013	/	Partitionnel	K-means	Euclidean, Cosine	Flickr(tags)
Du et al. 2016	Sémantique	Densité	DBSCAN	Basée sur une ontologie	Bing maps
Bouhamoum et al. 2018	/	Densité	DBSCAN	Jaccard	DBpedia, DBLP, Katrina and Charley
Eddamiri et al. 2019	/	Hiérarchique	Agglomeratif	Jaccard, Cosine, Sorensen	400 sets between groups of universities
Bamatraf et al. 2019	Terminologique	Partitionnel	K-medoids	/	SP2Bench, DrugBank

TABLE 1.2 – Comparaison des travaux existants pour la catégorisation dans le web sémantique.

1.9.3 Analyse

Il existe différents types de similarité dans le domaine des données liées. Plusieurs types ont été utilisés dans la littérature. La similarité structurelle compare les relations de chaque ressource. La similarité syntaxique et lexicale, le premier étant basé sur la chaîne de caractères qui compose le terme, le second étant une similarité basé sur une ressource externe comme WordNet. Une similarité sémantique est une similarité qui regroupe les similarités structurelles, terminologiques, et extensionnelle pour donner un taux de similarité des concepts qui comprend non seulement les valeurs mais aussi leur contexte.

Dans le cadre des formules de calcul des similarité, les mesures de Jaccard et de Sørensen sont utilisées pour comparer les relations des deux points à comparer et sont donc utilisées avec des données brutes. Les mesures cosinus et euclidiennes sont utilisées pour calculer la distance entre deux points qui ont été normalisés en vecteurs. Avec ces formules, les données doivent être converties en vecteurs préalablement, mais en contrepartie, elles peuvent être utilisées avec n'importe quel type de similarité, tant qu'elle est prise en compte par la conversion.

Nous remarquons que les principaux types de catégorisation non supervisée sont représentés dans la littérature. La catégorisation hiérarchique consiste à trouver les deux éléments les plus similaires ou les plus différents et à les regrouper ou les séparer jusqu'à obtenir une hiérarchie de tous les éléments du dataset. Ses avantages sont l'absence du nombre de clusters comme entrée et la bonne précision des clusters, mais comme les éléments sont testés un par un, le temps d'exécution devient un problème lors du traitement de datasets volumineux. Le résultat de l'algorithme n'est pas un groupe de clusters, mais une hiérarchie des éléments et doit être coupé par un autre algorithme, ce qui ajoute de la complexité et un temps d'exécution supplémentaire. Cette technique est représentée par l'algorithme agglomératif qui regroupe les deux éléments ou groupes les plus similaires et les place dans un même groupe.

La catégorisation partitionnelle est le regroupement des éléments par rapport à un centre qui est décidé selon l'algorithme utilisé. Les algorithmes partitionnels retournent les clusters en résultat, mais en contrepartie nécessitent le nombre de clusters en entrée et souffrent de problèmes de temps d'exécution avec les datasets volumineux. K-Means et K-Medoids sont les deux algorithmes partitionnels présentés dans les travaux mentionnés. L'algorithme K-means, dans l'exploration de données, commence par un premier groupe de centroïdes sélectionnés au hasard, qui sont utilisés comme point de départ pour chaque cluster, puis effectue des calculs itératifs pour optimiser les positions des centroïdes. Dans K-medoids, chaque cluster est représenté par l'un des points de données du cluster, ces points étant nommés médoides de cluster.

La catégorisation basée sur la densité utilise le nombre de voisins d'un élément, appelé densité. Les zones à haute densité sont considérées comme des clusters. Cette méthode a l'avantage de ne pas demander le nombre de clusters en entrée et elle est plus adaptée à des datasets volumineux. DBSCAN est l'algorithme qui représente la catégorisation basée sur la densité dans les travaux

existants. Cet algorithme utilise le nombre de voisins à proximité pour déterminer la densité. Les points avec peu ou pas de voisins sont considérés comme des données aberrantes.

Nous observons une diversité dans les datasets utilisés dans la littérature. Deux catégories émergent : les datasets génériques et les datasets spécialisés. Certains travaux utilisent des datasets spécialisés pour atteindre de meilleures performances dans un domaine donné. D'autres utilisent des datasets génériques qui servent à tester des travaux de catégorisation générale. On remarque également des travaux qui utilisent des datasets générés à partir de données non RDF.

1.10 Conclusion

Dans ce chapitre, nous avons exploré le domaine du web sémantique et des données liées, en commençant par leurs origines sur le web, puis en étudiant leurs caractéristiques et leurs langages de représentation. Nous avons également abordé des notions d'ontologie et présenté différentes techniques de classification non supervisée. Nous nous sommes intéressés à l'approche hiérarchique, à l'approche partitionnelle et à l'approche basée sur la densité. De plus, nous avons présenté quelques travaux qui mettent en œuvre ces techniques dans le cadre de la catégorisation des données liées.

Dans le chapitre suivant, nous allons présenter la conception de notre système.

Chapitre 2

CONCEPTION DU SYSTÈME

2.1 Introduction

La conception est une étape cruciale pour la réalisation d'un système. Dans ce chapitre, nous présenterons les points importants de notre système. Nous commencerons par les critères du système, dont le choix a été informé par l'analyse des limites des systèmes existants présentés lors du chapitre précédent. Puis nous présentons toutes les étapes d'exécution de notre système, en commençant par la phase de pré-catégorisation où nous avons chargé et préparé notre dataset. Ensuite, nous passons à la phase de catégorisation où nous effectuons le calcul de la distance et la catégorisation proprement dite. Enfin, nous effectuons une transformation des données pour visualiser le résultat final de notre système.

2.2 Les caractéristiques de notre système

La distance sémantique est une mesure qui combine les trois éléments qui composent le triangle sémantique, à savoir la distance terminologique, structurelle et extensionnelle. Les travaux mentionnés dans l'état de l'art se concentrent sur un ou deux types de similarité, mais notre système utilise la mesure sémantique pour une catégorisation sémantique.

Une formule de similarité est une façon de mesurer le taux de similarité entre deux éléments du dataset. Nous utilisons une similarité sémantique pour notre système en combinant trois mesures, terminologique, structurelle et extensionnelle. La similarité terminologique sera calculée en utilisant l'outil "WordNet" pour déterminer la similarité lexicale, ainsi que l'indice de Jaccard qui détermine la similarité syntaxique des termes des données. La similarité structurelle est calculée en comparant les labels des prédicats appartenant aux éléments. La similarité extensionnelle compare les objets reliés par des prédicats équivalents appartenant aux éléments.

Le but de notre système étant la catégorisation, le choix de la technique et de l'algorithme revêt une importance critique. En observant les travaux existants, nous remarquons que la catégorisation à base de densité est la plus adaptée à notre utilisation en raison de sa capacité à

gérer les datasets volumineux, et l'absence du nombre de clusters en entré. Le DBSCAN est un algorithme de catégorisation basé sur la densité, populaire en raison de sa capacité à traiter les données aberrantes.

2.3 Architecture globale de notre système

Notre système consiste à effectuer une catégorisation automatique des ressources d'un dataset de données liées. Le schéma global proposé est illustré dans la figure 1.1. Ensuite, nous détaillerons les différentes étapes et algorithmes utilisés.

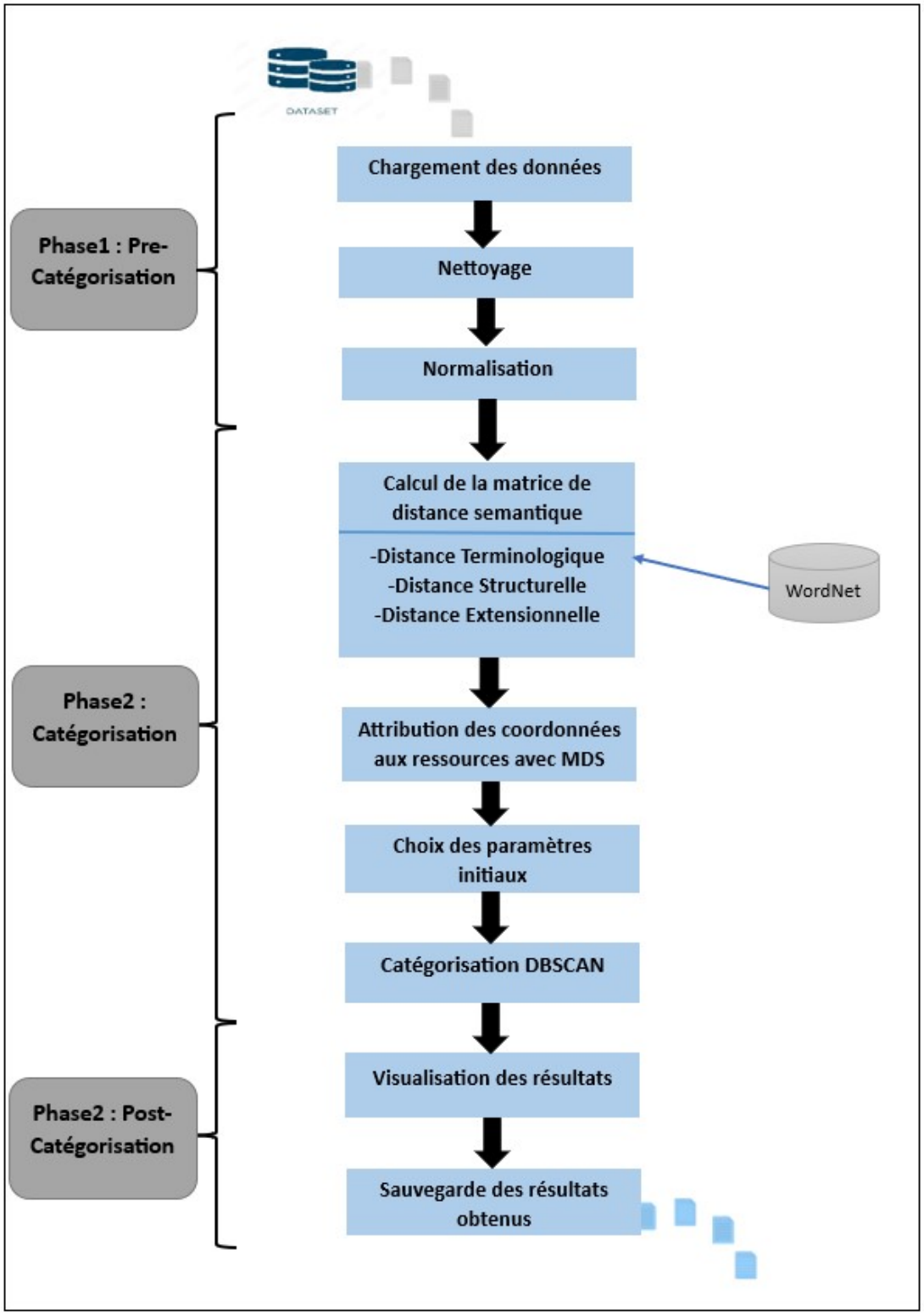


FIGURE 2.1 – Schéma global

2.4 Description de la solution proposée

Notre travail est composé de trois parties principales. Chaque partie est constituée de plusieurs étapes complexes qui transforment les données jusqu'à l'obtention du résultat voulu.

Phase 1 : Pré-catégorisation

Cette phase consiste à préparer et transformer les données pour qu'elles soient facilement utilisables par le reste du système. Nous commençons par télécharger le dataset. Ensuite, vient l'étape de nettoyage des données qui enlève les triplets incomplets et les données inutiles. Finalement, l'étape de normalisation transforme la structure des données, donc on aura trois étapes dans la phase de pré-catégorisation :

— Chargement des données

— Nettoyage

— Normalisation

Phase 2 : Catégorisation

Cette phase porte sur les traitements liés à la catégorisation ressources. Elle commence par trois étapes qui calculent une matrice de distance pour chaque type de similarité du triangle sémantique qui sont ensuite combinées dans une matrice de distance sémantique. Ensuite viens l'étape de catégorisation , ou les données sont catégorisées par rapport à leur distance sémantique. Cette phase est composé de sept étapes.

— Construction de la matrice de distance terminologique.

— Construction de la matrice de distance structurelle.

— Construction de la matrice de distance extensionelle.

— Construction de la matrice de distance sémantique.

— Transformation des données.

— Le choix des paramètres initiaux de DBSCAN.

— Catégorisation par DBSCAN.

Phase 3 : Post-catégorisation

La phase finale du système qui sert à montrer le résultat de la catégorisation de manière lisible et compréhensible. Le résultat de la catégorisation sera représenté sous forme de graphe de points où chaque couleur représente un cluster, donc elle comporte deux étapes :

— Visualisation des résultats.

— Sauvegarde des résultats obtenus.

2.4.1 Phase 1 : Pré-catégorisation

La première phase du système. Elle sert à préparer les données avant le catégorisation.

Étape 1 : Chargement des données

Dans cette étape, nous avons extrait les triplets du dataset, qui seront stockés dans la RAM pour être rapidement accessibles par le reste du système.

Étape 2 : Nettoyage

Cette étape consiste à éliminer tous les triplets inutilisables pour la catégorisation, tels que les triplets incomplets et les triplets dont la ressource ou l'objet est un fichier. Le but de cette étape est de réduire le niveau de bruit dans le dataset.

Étape 3 : Normalisation

Dans cette étape nous avons effectué les opérations suivantes pour rendre les données plus consistantes et exploitables.

1. Le remplacement de l'URI par son label (figure 2.2).

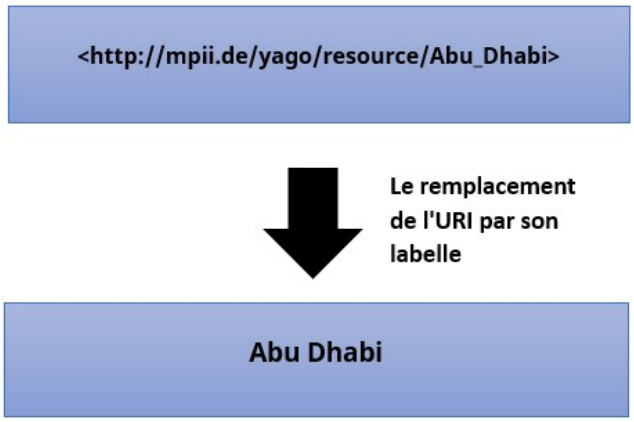


FIGURE 2.2 – Le remplacement de l'URI par son label

2. La normalisation de la case qui consiste à mettre tous les mots en minuscule.
3. La normalisation des liaisons qui consiste à remplacer les caractères de liaison par un espace.
   Exemple : "American-Samoa" devient "American Samoa"
4. La suppression des caractères spéciaux.
   Exemple : "NOM ?" devient "NOM"
5. La normalisation de la structure, chaque élément du dataset est sous forme "Ressource, Predicat1, Objet1, Predicat2, Objet2 ..."

2.4.2 Phase 2 : Catégorisation

La matrice de distance sémantique est une matrice carrée et symétrique qui ressemble à une matrice de distance euclidienne, sauf qu'on utilise la distance sémantique basée sur la similarité sémantique qui est la combinaison des trois type de similarité sémantique : la similarité terminologique, structurelle et extensionelle.

Dans cette phase nous présentons les méthodes utilisées pour calculer chaque type de distance sémantique.

2.4.2.1 Construction de la matrice de distance terminologique

Pour construire une matrice de distance teminologique, il nous faut d'abord calculer la similarité teminologique entre deux termes, qui est ensuite soustraite de un pour obtenir une distance teminologique.

La similarité teminologique est la mesure la plus importante du système car elle est utilisée pour calculer les autres types. Elle est composée de deux parties :

1. **La mesure syntaxique** :
   Elle permet de comparer les lettres des termes sans comprendre leur sens. Pour cette tâche, nous avons choisi d'utiliser l'indice de Jaccard qui calcule le taux de similarité entre deux ensembles. Le résultat donné est compris entre 0 et 1, car il correspond à la division de la cardinalité de l'intersection par la cardinalité de l'union des deux ensembles.

   $$Jaccard(U, V) = \frac{|U \cap V|}{|U \cup V|}$$

   Dans notre cas, les ensembles sont les chaînes de caractères et les éléments des ensembles sont les lettres desdites chaînes de caractères.

   Exemple : la similarité syntaxique entre "anguilla" et "aruba".

   "anguilla" est composé de 8 caracteres.

   "aruba" est composé de 5 caracteres.

   La cardinalité de l'union est de 33 et la cardinalité de l'intersection est de 5. L'indice de Jaccard pour ces deux chaînes est de 0.15.

   $$Jaccard(\text{"}anguilla\text{"}, \text{"}aruba\text{"}) = \frac{|\text{"}anguilla\text{"} \cap \text{"}aruba\text{"}|}{|\text{"}anguilla\text{"} \cup \text{"}aruba\text{"}|} = \frac{|2|}{|13|} = 0.15$$

2. **La mesure lexicale** :
   Elle permet de comparer les termes en utilisant une ressource externe qui prend en compte la signification du terme. Nous avons utilisé WordNet.

   Pour commencer, nous avons importé les fonctions wordtokenize et postag, ainsi que la classe WordNet de la bibliothèque NLTK. La première étape consiste à tokenizer la chaîne de caractères en séparant les mots (figure 2.3).

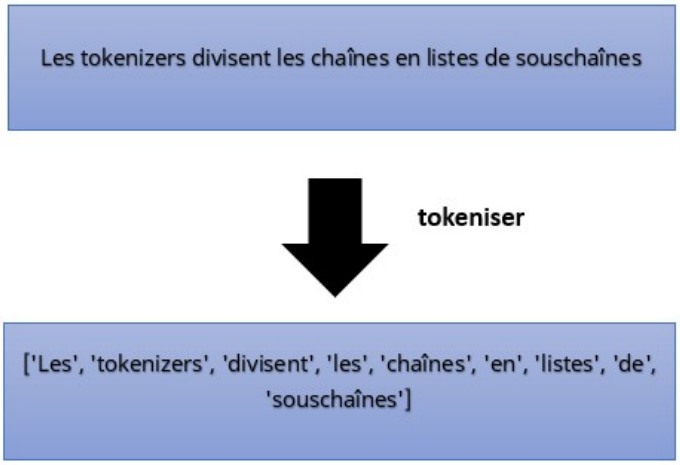


FIGURE 2.3 – Tokenization

Ensuite, les mots sont étiquetés selon leur nature. WordNet prend en compte quatre catégories de mots, les noms , les adjectifs, les verbes et les adverbes (figure 2.4).

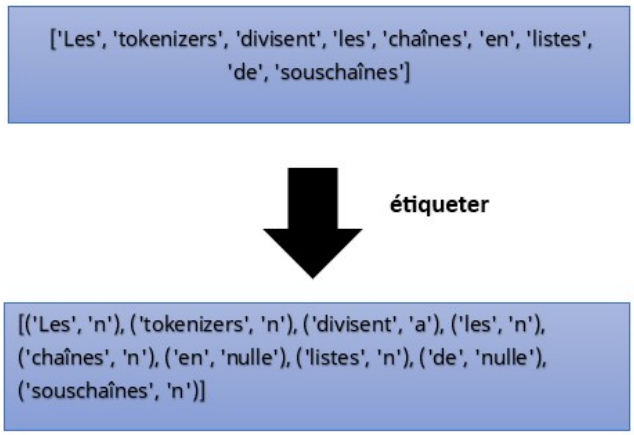


FIGURE 2.4 – Etiquetage

L'étape suivante consiste à extraire les synsets des mots étiquetés. Les synsets sont des objets de la classe WordNet qui comprennent la définition, les synonymes, le champ lexical ainsi que leur position dans différents arbres taxonomiques et sémantiques. Cette position nous permet de déterminer la similarité entre deux mots.

Nous avons opté pour une méthode appelée **"path similarity"** qui attribue un score entre 0 et 1 en utilisant la distance entre les deux mots dans l'arbre taxonomique. Cette distance est déterminée en utilisant le nombre d'arêtes qui les séparent à travers le plus court chemin dans l'arborescence de WordNet. Pour déterminer la similarité entre deux phrases, c'est-à-dire deux ensembles de mots étiquetés, nous avons utilisé l'algorithme suivant :

Algorithme 1 : Algorithme de calcul de la similarité lexicale

```
Input : synsets1, synsets2
Result : Similarité_lexicale
score=0;
compteur=0 ;
i=j=0;
n1 = longueur(synsets1);
n2 = longueur(synsets2);
for i in 0 to n1 do
    for j in 0 to n2 do
        best_score = max([path_similarity(synsets1[i],synsets2[j])])
    score =score+ best_score ;
    compteur=compteur+1;
Similarité_lexicale = Score/compteur;
```

Les deux résultats sont pondérés pour obtenir un score final entre 0 et 1 qui représente la similarité des deux termes donnés en entrée.

Cet algorithme est utilisé pour calculer la distance terminologique entre chaque élément qui sera chargé dans la matrice de distance terminologique.

2.4.2.2 Construction de la matrice de distance structurelle

Pour construire une matrice de distance structurelle, il nous faut d'abord calculer la similarité structurelle entre deux termes, qui est ensuite soustraite de un pour obtenir une distance structurelle.

La similarité structurelle compare les ressources par rapport à leurs prédicats. Elle commence par extraire les labels des prédicats des deux ressources. Une fois les labels séparés en deux groupes, leur similarité est calculée en utilisant la mesure terminologique. Dans notre fonction, nous comparons la similarité terminologique de chaque label du premier groupe avec les labels du deuxième. Cela nous donne un score temporaire qui sera divisé par le nombre d'opérations qui ont été effectuées pour nous donner un score final entre 0 et 1.

Algorithme 2 : Algorithme de calcul de la similarité structurelle

```
Input : listePrédicat1, listePrédicat2
Result : Similarité_structurelle
i=j=1;
score=0;
compteur=0;
n1 = longueur(listePrédicat1);
n2 = longueur(listePrédicat2);
while (i < n1) do
    while (j < n2) do
        score = score + similarité_terminologique(listePrédicat1[i], listePrédicat2[j]);
        compteur=compteur+1;
        j=j+1;
    i=i+1;
Similarité_structurelle = score / compteur;
```

Cet algorithme est utilisé pour calculer la distance structurelle entre chaque élément qui sera chargé dans la matrice de distance structurelle.

2.4.2.3 Construction de la matrice de distance extensionelle

Pour construire une matrice de distance extensionnelle, il nous faut d'abord calculer la similarité extensionnelle entre deux termes, qui est ensuite soustraite de un pour obtenir une distance extensionnelle.

La similarité extensionelle compare les ressources par rapport aux extensions. Elle compare les termes des extensions reliées par deux prédicats équivalents. Le degré de similarité des termes et l'équivalence des extensions se fait en utilisant la similarité terminologique. Comme dans la fonction précédente, on utilise un score temporaire et le nombre d'opérations pour calculer un score de similarité extensionnelle compris entre 0 et 1.

Algorithme 3 : Algorithme de calcul de la similarité extensionelle

```
Input : listeObjet1, listeObjet2, listePrédicat1, listePrédicat2
Result : Similarité_extensionelle
i=j=1;
score=0;
compteur=0;
n1 = longueur(listeObjet1);
n2 = longueur(listeObjet2);
while (i < n1-1) do
    while (j < n2-1) do
        if similarité_terminologique(listePrédicat1[i],listePrédicat2[j]) > 0.7 then
            score = score + similarité_terminologique(listeObjet1[i],listeObjet2[j]);
            compteur++;
        j=j+1;
    i=i+1;
if compteur == 0 then
    score = 0;
Similarité_extensionelle = score / compteur;
```

Cet algorithme est utilisé pour calculer la distance extensionelle entre chaque élément qui sera chargé dans la matrice de distance extensionelle.

2.4.2.4 Construction de la matrice de distance sémantique

Ayant trois matrices de distance représentant les trois types de similarité, nous construisons la matrice de distance sémantique en utilisant cette formule :

$$Msemantique[i,j] = \frac{Mterm[i,j] * p1 + Mstruc[i,j] * p2 + Mextent[i,j] * p3}{p1 + p2 + p3}$$

Où i et j représentent les indices des deux éléments à comparer, et p1, p2 et p3 sont les coefficients de chaque type de similarité.

2.4.2.5 Transformation des données

La nature non euclidienne de la distance sémantique cause un problème de dimensionnalité au niveau de la catégorisation, ou le nombre de dimension requise pour tracer les point sont trop élevé et cause des problèmes avec l'algorithme DBSCAN. C'est pour cela qu'on a crée un une approximation en deux dimensions de tout les éléments du dataset en utilisant la méthode MDS (Multi-Dimensional Scaling) qui utilise une matrice de distance, souvent euclidienne, pour réduire le nombre de dimensions des données. Nous avons donc utilisé notre matrice de distance sémantique pour générer des coordonnées pour chaque ressource.

2.4.2.6 Le choix des paramètres initiaux de DBSCAN

DBSCAN Sélectionne un point de données au hasard et recherche tous les points de données à une distance maximale fixée (epsilon) de ce point. Si le nombre de points trouvés est supérieur ou égal à un nombre minimal fixé (minPts), alors ce point de données est considéré comme un point de départ d'un cluster. Ensuite recherche tous les points de données accessibles à partir du point de départ en utilisant la même distance maximale et le même nombre minimal de points. Ces points sont ajoutés au cluster en cours de construction.
L'algorithme répète le processus pour tous les points de données qui n'ont pas encore été attribués à un cluster. Les points qui ne peuvent pas être assignés à un cluster sont considérés comme un point de bruit.
Avant de pouvoir exécuter l'algorithme, les valeurs d'epsilon et de minPts doivent être fixées. Il n'existe pas de méthode pour choisir la valeur de minPts optimale, mais certaines sources recommandent d'utiliser ln(n), où n est le nombre d'éléments (ou de ressources dans notre cas), lorsque la taille du dataset est suffisamment élevée. Contrairement à minPts, il existe une méthode pour déterminer epsilon, décrite dans [14], représentée dans l'algorithme suivant, qui consiste à dessiner un graphe représentant les distances entre chaque élément et son voisin le plus proche.
Le graphe a deux axes, le y représente la distance entre les éléments et le x représente les éléments triés dans un ordre ascendant. Nous avons pris un point au niveau du changement de courbure critique, la distance qui vous correspond est la valeur idéale pour epsilon par rapport à une valeur de minPts donnée. Contrairement à epsilon, la valeur optimale de minPts ne peut être déduite à partir des données de départ, mais elle peut être optimisée par rapport à une métrique d'évaluation après l'exécution de l'algorithme.

Algorithme 4 : Algorithme de sélection de la valeur optimale de epsilon

Input : liste_éléments, MinPts
Result : eps_values
for *i in n* **do**
  for *j in n* **do**
    $d(i,j) = distance\_semantique(x_i, x_j)$
    $graphe(i) = moyenne\_k\_plus\_proche(d(i,j), MinPts)$
$graphe = trie\_ascendant(graphe)$;

La valeur de epsilon correspond à un changement critique de la courbe

Un exemple d'exécution de cette méthode nous donne le graphe de la figure 2.5 :

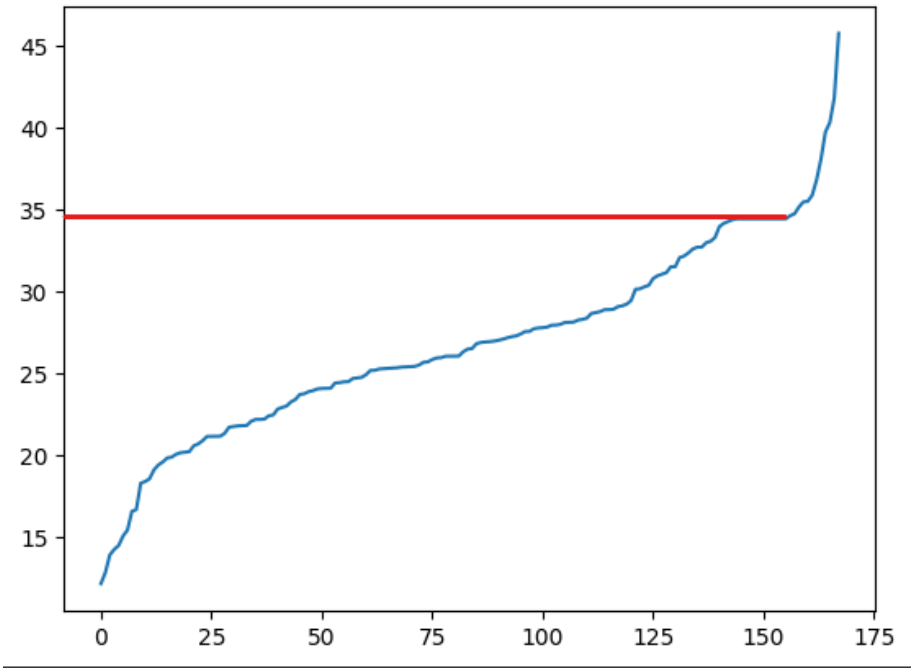


FIGURE 2.5 – Graphe de epsilon.

2.4.2.7 Catégorisation par DBSCAN

Ayant une matrice de distance, une valeur pour epsilon et minPts, nous pouvons utiliser l'algorithme de catégorisation DBSCAN. Le résultat se présentera sous forme de clusters et de points de bruit. Un exemple de catégorisation est illustré dans la figure 2.6.

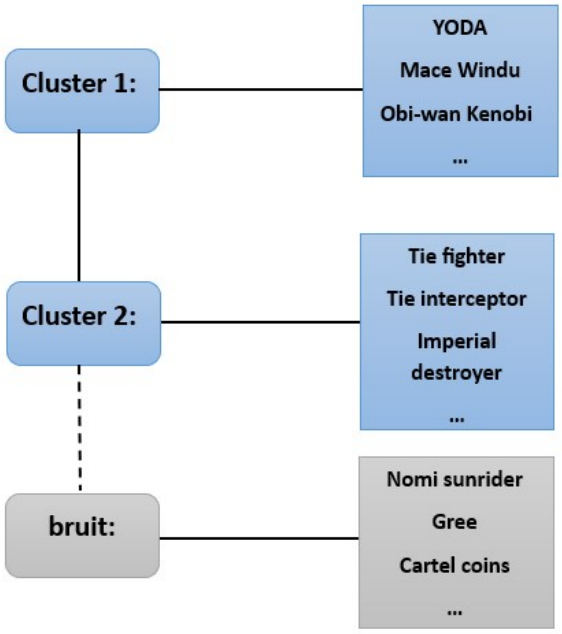


FIGURE 2.6 – Exemple de catégorisation par DBSCAN

2.4.3 Phase 3 : Post-catégorisation

Le but de la visualisation est de présenter les résultats sous une forme lisible et compréhensible. Nous avons utilisé les points représentatifs fournis par la méthode MDS pour créer un graphe de points où chaque point est coloré en fonction du cluster auquel il appartient.

Les points de bruit sont colorés en noir afin de les différencier du reste sans créer de distraction visuelle.

La figure 2.7 présente un exemple de cette méthode de visualisation des résultats.

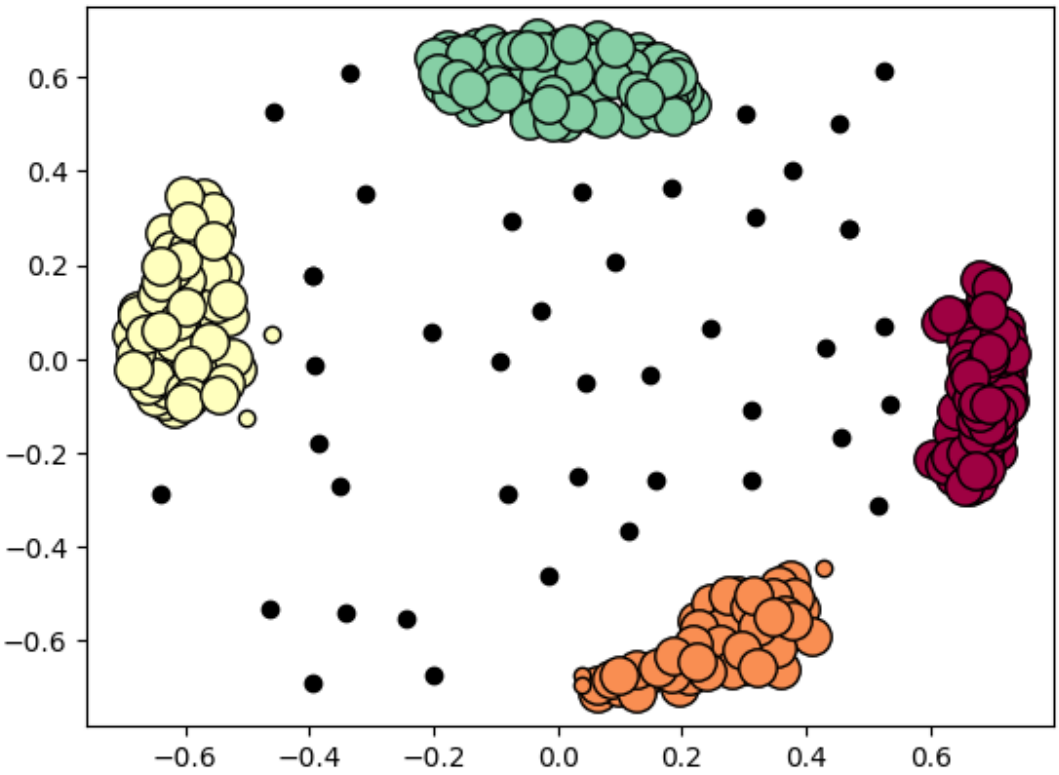


FIGURE 2.7 – Exemple de graphe de points colorés

2.5 Conclusion

Dans ce chapitre, nous avons présenté les étapes détaillées de notre solution pour la découverte des liens. En détaillant les trois phases : pré-traitement, calcul de la similarité sémantique, suivi de la catégorisation, pour finir avec la visualisation des résultats de la catégorisation.

Dans le chapitre suivant, nous allons implémenter et mettre en œuvre ce que nous avons déjà proposé dans ce chapitre, autrement dit l'implémentation et la validation des résultats de notre système.

Chapitre 3

L'IMPLÉMENTATIONS ET TEST DU SYSTÈME

3.1 Introduction

Après avoir réalisé la phase de conception de notre système, ce chapitre couvre tous les détails relatifs à l'aspect d'implémentation de notre approche, qui s'appuie sur la catégorisation des données présentées dans le chapitre précédent.

Pour ce faire, nous avons tout d'abord présenté les différents environnements de développement et les différents outils utilisés. Ensuite, nous décrivons de façon visuelle notre implémentation à travers des captures d'écran des différentes interfaces de notre système. Par la suite, nous procédons à la validation et aux tests de notre système. Enfin, nous affichons le résultat ainsi que sa représentation graphique.

3.2 Environnement de développement

Un bon environnement de développement est nécessaire au bon déroulement de l'implémentation de notre système. C'est pour cela que nous avons utilisé les outils suivants :

3.2.1 Matériel utilisé

Tout le travail et les tests présents dans ce mémoire ont été réalisés sur une machine avec les spécifications suivantes :

- **CPU** :AMD Ryzen 5 3600
  - 6 processeur physique
  - 12 processeur logique
  - 3.59 GHz vitesse de base
- **RAM** : 16 Go 3200hz DDR4

— **SYSTÈME D'EXPLOITATION** : Windows 10 Pro 64-bit

3.2.2 Langage utilisé

Nous avons utilisé le language de programation Python.

I. **Python**

Python est un langage de programmation générique interprété de haut niveau, inventé par Guido van Rossum à la fin des années quatre-vingts pour être un successeur du langage de programmation ABC. Il repose sur une philosophie de conception qui permet la lisibilité du code. Il fournit des constructions permettant une programmation claire. Python propose un système de typage dynamique et une gestion automatique de la mémoire. Il prend en charge plusieurs paradigmes de programmation, notamment orienté objet, impératif, fonctionnel et procédural. La phrase "batteries included" est souvent utilisée pour décrire Python grâce à sa bibliothèque standard complète et polyvalente, offrant beaucoup de fonctionnalités sans avoir à installer des bibliothèques externes.

L'interpréteur Python est facilement étendu avec de nouvelles fonctions et de nouveaux types de données implémentés en C ou C ++ (ou d'autres langages pouvant être appelés à partir de C). Il convient également comme langage d'extension pour les applications personnalisables [30] [1].

Le choix de ce langage présente les avantages suivants :

— Python est gratuit.

— Python offre une bibliothèque standard complète et expansive.

— Python est orienté objet, mais n'impose pas ce type de programmation.

— Python offre la possibilité d'utiliser les bibliothèques d'apprentissage les plus populaires et reconnues.

II. **Bibliothèques utilisées**

a) **Scikit-learn**

Scikit-learn est une bibliothèque d'apprentissage automatique pour le langage de programmation Python. Il comporte divers algorithmes de classification, de régression et de catégorisation, notamment des support-vector machines, random forests, MDS, k-means and DBSCAN, et est conçu pour interagir avec les bibliothèques numériques et scientifiques Python NumPy et SciPy [27].

1. https ://peps.python.org/pep-0206/

b) **NLTK**

Natural Language Toolkit est une bibliothèque logicielle en Python permettant un traitement automatique des langues, développée par Steven Bird et Edward Loper du département d'informatique de l'Université de Pennsylvanie. En plus de la bibliothèque, NLTK fournit des démonstrations graphiques, des tutoriels, ainsi que la documentation de l'interface de programmation (API) [2].

WordNet est une base de données lexicale pour la langue anglaise, développé en 1986 à l'Université de Princeton et faisant partie du corpus NLTK [16].

WordNet peut être utilisé avec le module NLTK pour trouver la signification des mots, des synonymes, des antonymes, etc.

c) **RDFLIB**

RDFLib est une bibliothèque Python pour travailler avec RDF, un langage simple mais puissant pour représenter la connaissance. Cette bibliothèque contient des parseurs/sérialiseurs pour presque toutes les sérialisations RDF connues, telles que RDF/XML, Turtle, N-Triples et JSON-LD [3].

d) **Pandas**

Pandas est une bibliothèque écrite pour le langage de programmation Python pour la manipulation et l'analyse de données. En particulier, il propose des structures de données et des opérations de manipulation de tableaux numériques et de séries chronologiques [4].

e) **Numpy**

NumPy est une bibliothèque pour langage de programmation Python, destinée à manipuler des matrices ou tableaux multidimensionnels ainsi que des fonctions mathématiques opérant sur ces tableaux [5].

f) **Matplotlib**

Matplotlib est une bibliothèque de visualisation de données en Python. Elle offre des fonctionnalités permettant de créer une grande variété de graphiques, de diagrammes et de visualisations interactives. Matplotlib est flexible et hautement personnalisable, offrant un contrôle précis sur chaque élément graphique [6].

III. **L'environnement d'exécution**

a) **Visual Studio Code**

Est un éditeur de code open-source créé par Microsoft pour les systèmes d'exploitation Windows et Linux. Il offre une grande flexibilité en termes de langages de

2. https ://www.nltk.org/
3. https ://rdflib.dev/
4. http ://pandas.pydata.org/pandas-docs/stable/getting_started/overview.html
5. https ://numpy.org/doc/stable/
6. https ://matplotlib.org/

programmation pris en charge, notamment Java, Javascript, Python et C++. Grâce à ses extensions, il permet de développer des applications Web.

b) **Google Colab**
Google Colab appelé aussi Colaboratory est un service gratuit offert par Google. Il repose sur l'environnement Jupyter Notebook et a été conçu pour faciliter la formation et la recherche en apprentissage automatique. Cette plateforme permet de réaliser l'entraînement de modèles de Machine Learning directement dans le cloud, sans nécessiter l'installation d'un logiciel sur votre ordinateur, à l'exception d'un navigateur.

3.3 Présentation de l'application

Le potentiel de notre système peut être entravé si l'utilisateur n'est pas familier avec le langage de programmation Python. C'est pour cela que nous avons réalisé une interface graphique pour simplifier l'utilisation de notre système.

La figure 3.1 représente la page d'accueil du système.

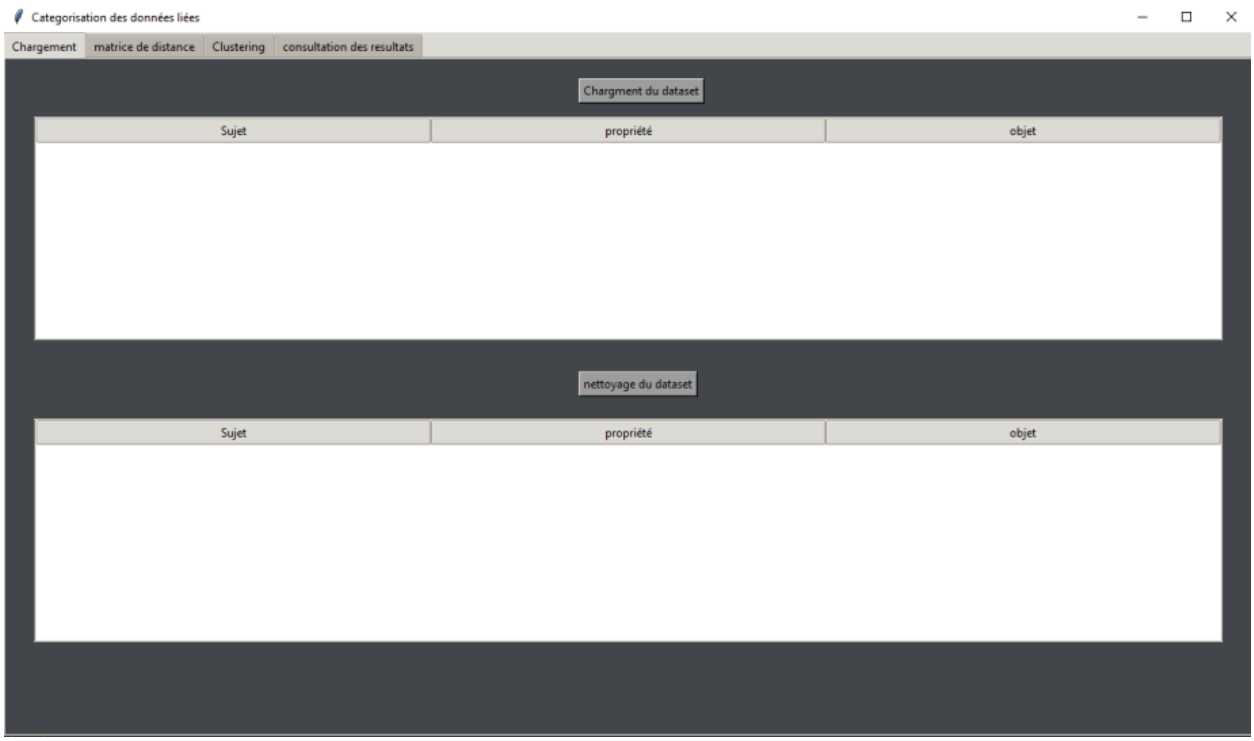


FIGURE 3.1 – L'interface d'accueil.

L'utilisateur peut charger un dataset sous n'importe quel format RDF. La figure 3.2 montre la fenêtre de chargement du dataset.

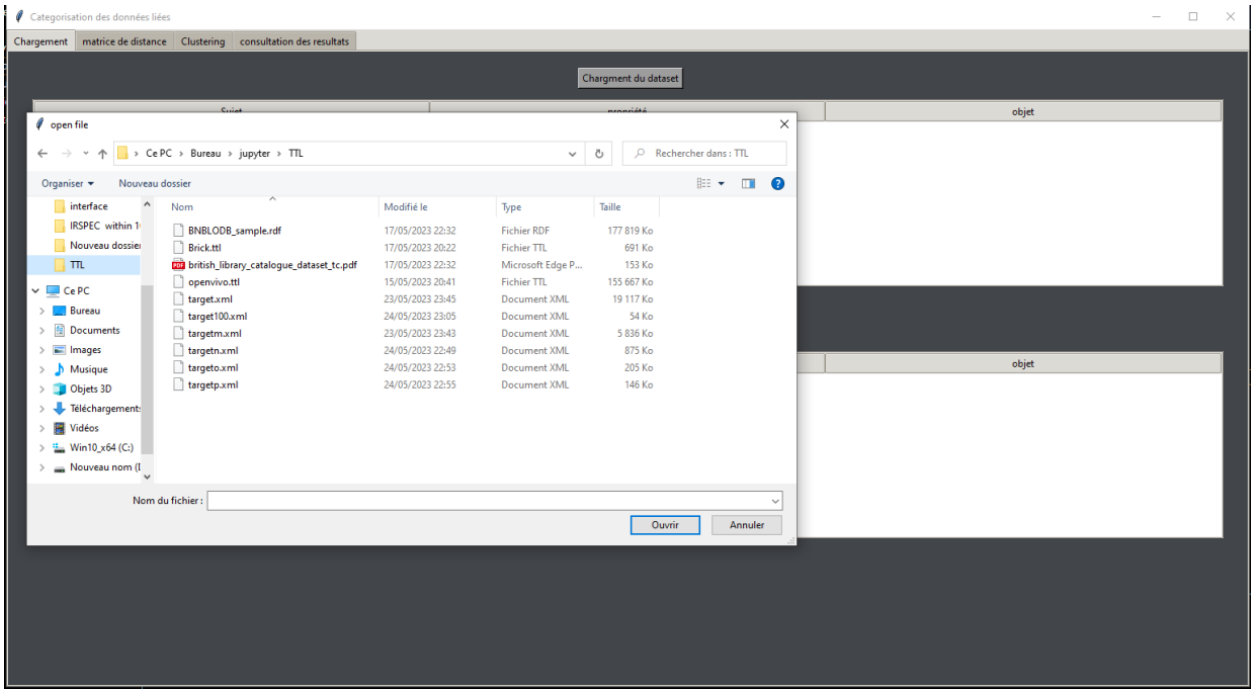


FIGURE 3.2 – Chargement de dataset.

Une fois le dataset chargé, l'interface affiche les URI des triplets pour confirmer visuellement le chargement du fichier (figure 3.3).

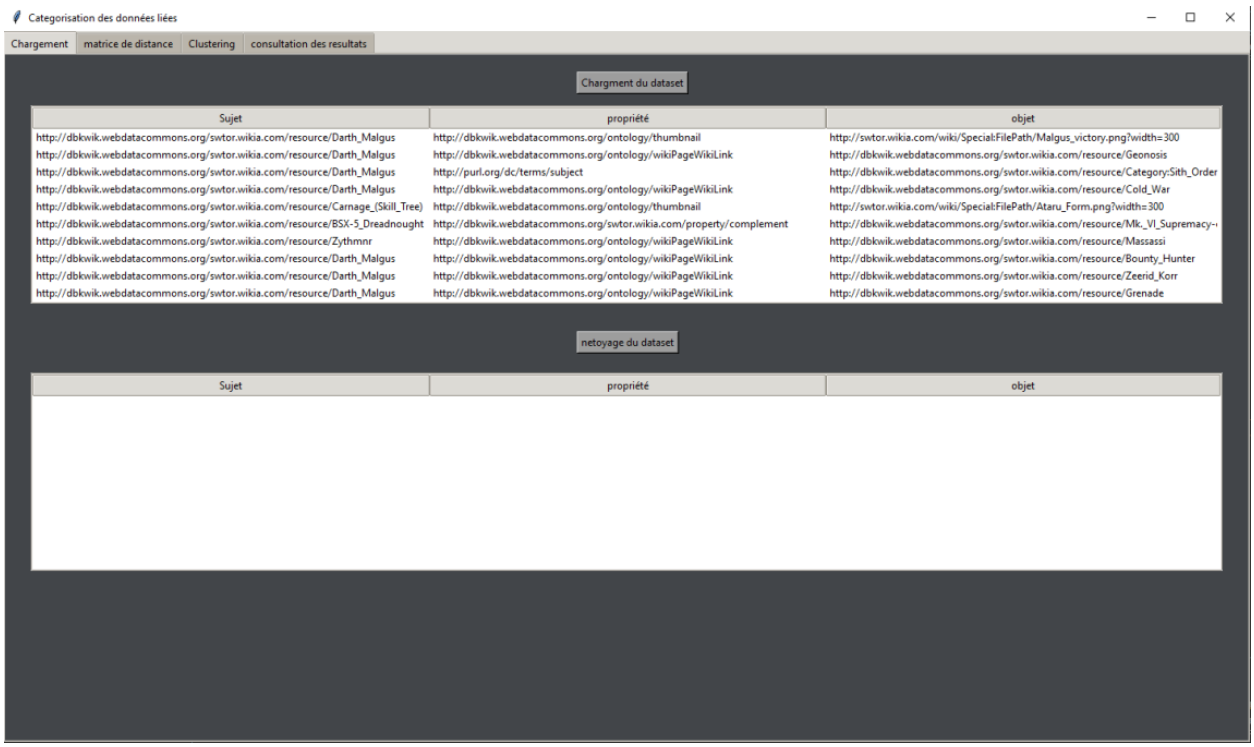


FIGURE 3.3 – Dataset chargé

L’utilisateur appuie sur le bouton "nettoyage de dataset"pour transformer la liste de triplets URI en une liste de triplets (figure 3.4).

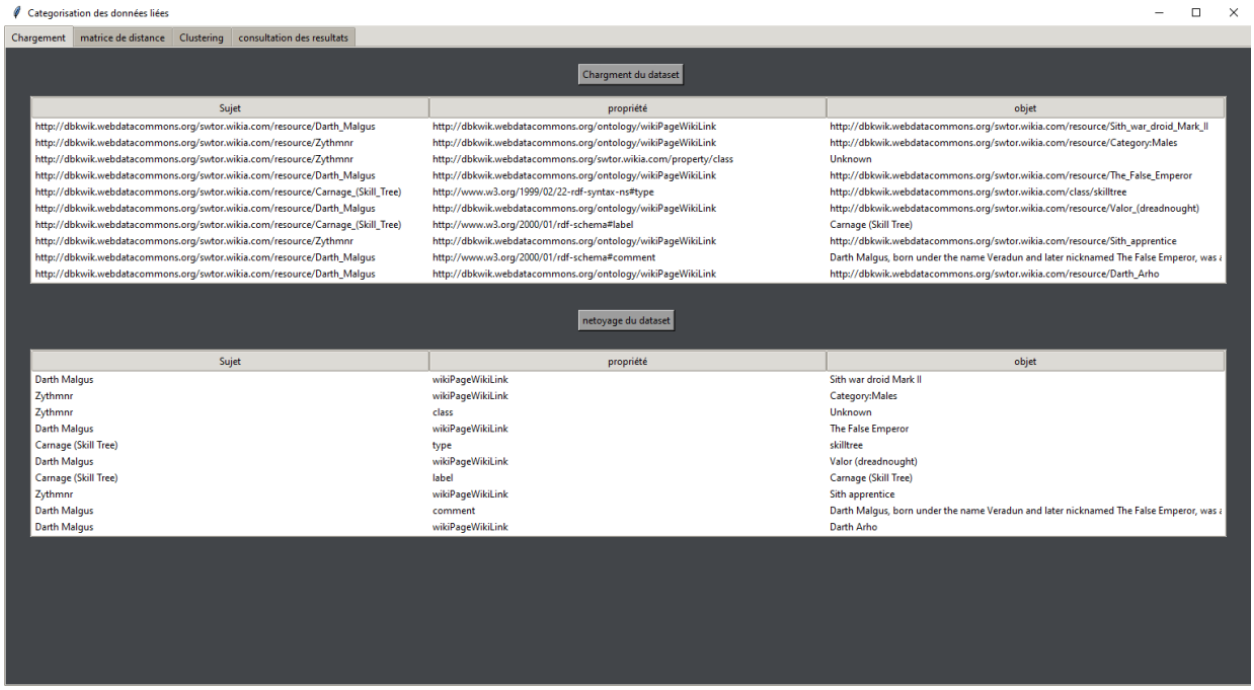


FIGURE 3.4 – Nettoyage de dataset.

La figure 3.5 montre la deuxième page de l’interface où les trois boutons à gauche permettent de lancer le calcul de chaque matrice de similarité. Le système enregistre les matrices dans le dossier racine pour une utilisation ultérieure.

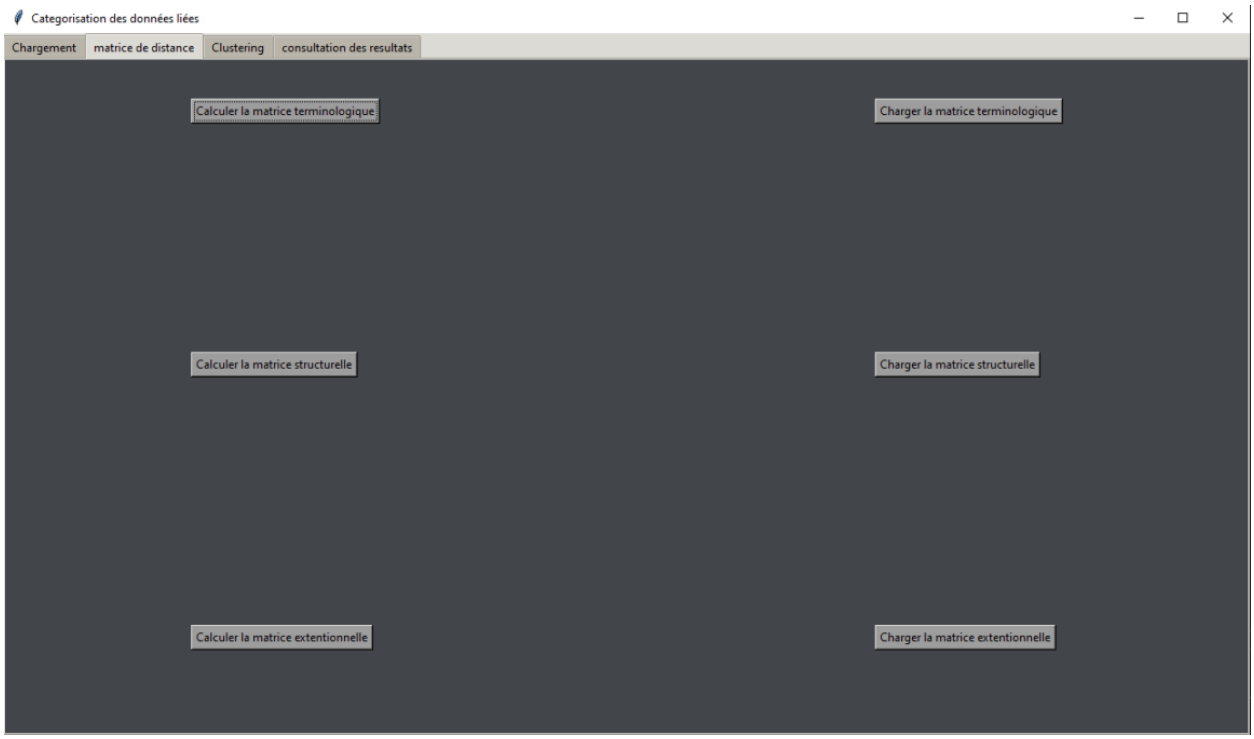


FIGURE 3.5 – Calculer les matrices de distance.

Les trois boutons à droite permettent à un utilisateur qui détient des matrices de similarité sémantique de les charger directement dans le système sans avoir à les calculer. La figure 3.6 montre la fenêtre de chargement de la matrice.

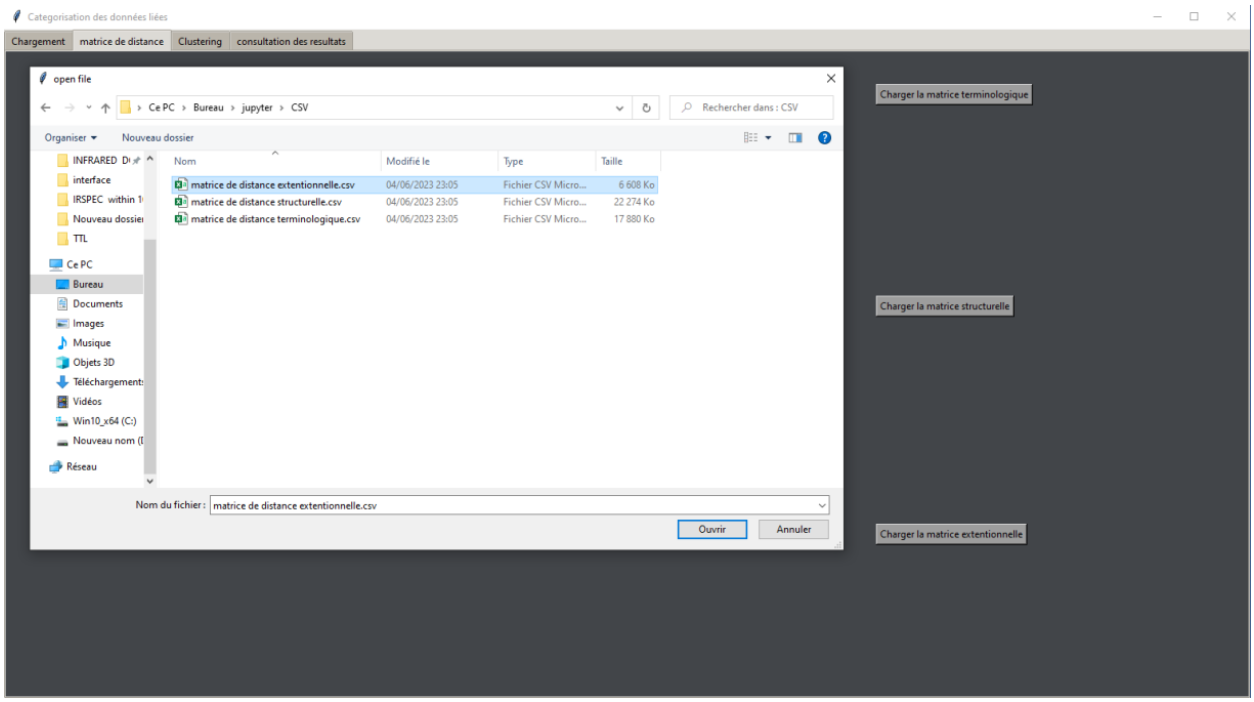


FIGURE 3.6 – Chargement des matrices de distance.

La figure 3.7 montre la troisième page de l'interface. L'utilisateur a accès à plusieurs champs où il peut entrer les valeurs des coefficients de pondération de chaque type de similarité ainsi que la valeur du paramètre point minimum de l'algorithme DBSCAN. L'utilisateur doit appuyer sur un bouton intitulé "Lancer la catégorisation" pour lancer l'algorithme.

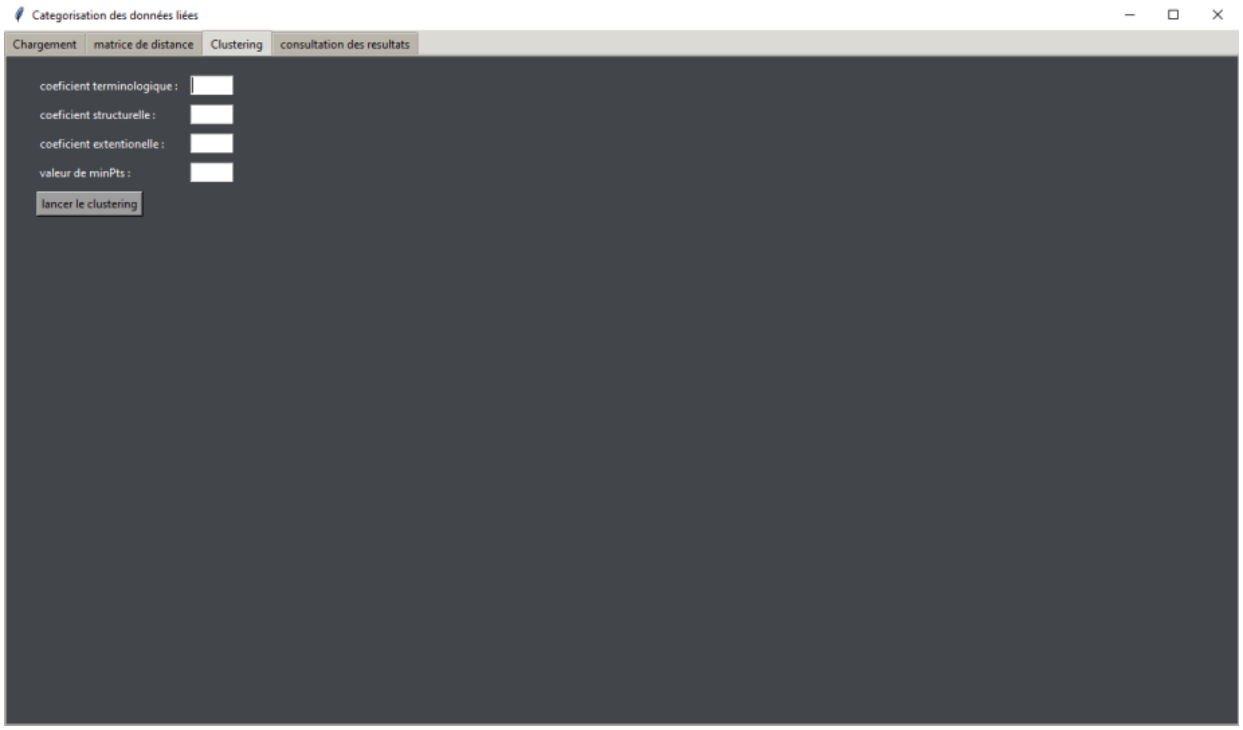


FIGURE 3.7 – Lancer la catégorisation.

Le résultat de la catégorisation s'affiche sous forme de graphe coloré. La valeur de la métrique d'évaluation sera également affichée dans l'interface (figure 3.8).

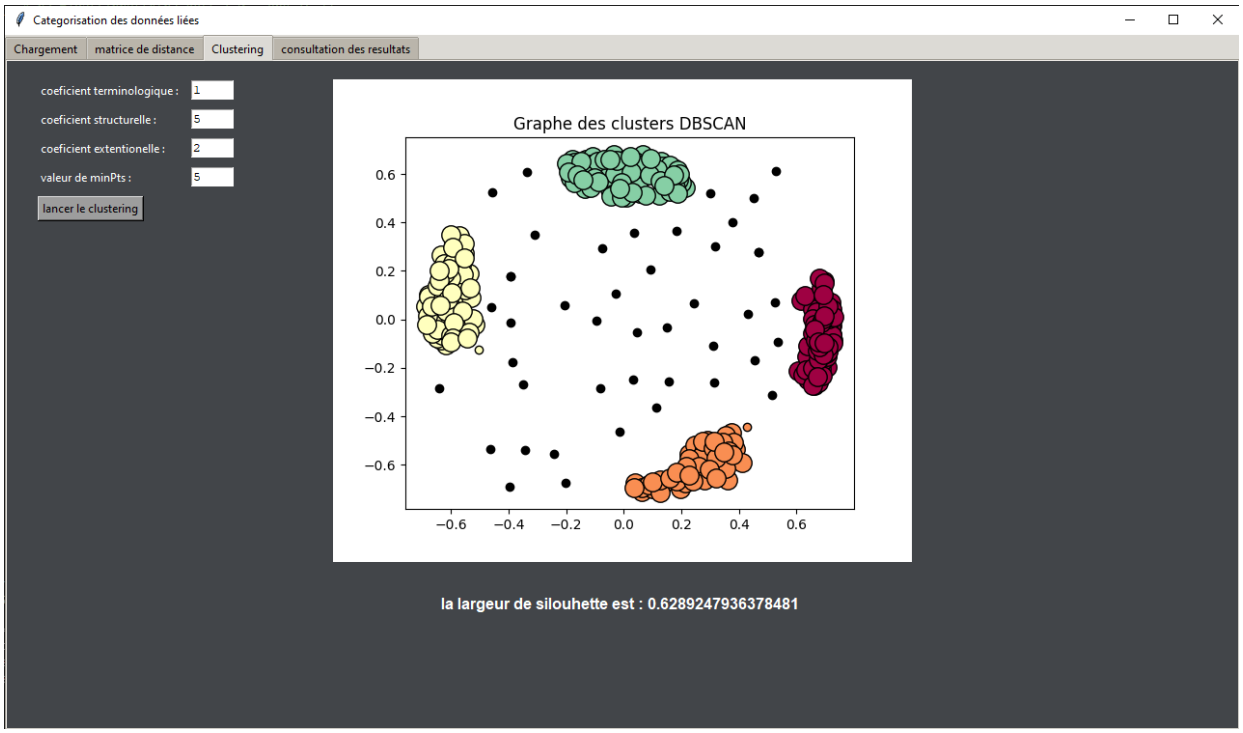


FIGURE 3.8 – Résultat la catégorisation.

La figure 3.9 montre la quatrième page de l'interface, qui affiche les ressources appartenant à un cluster sélectionné avec une liste déroulante placée au-dessus.

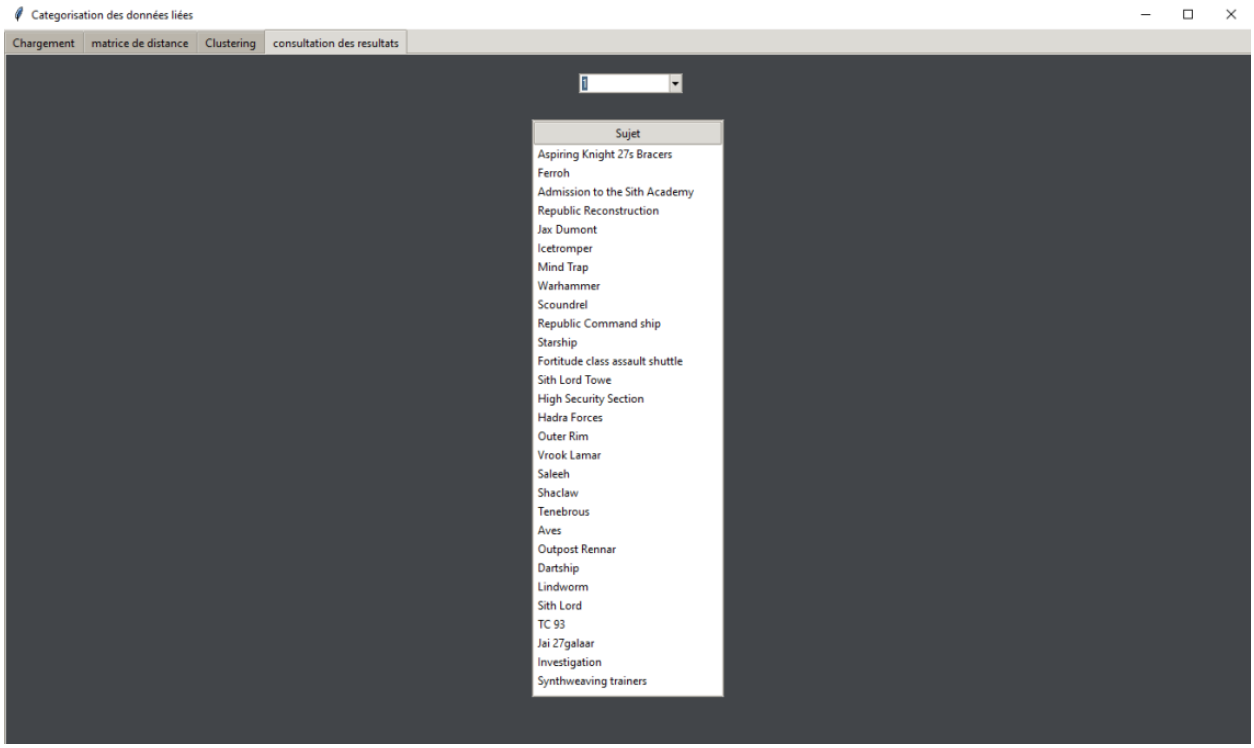
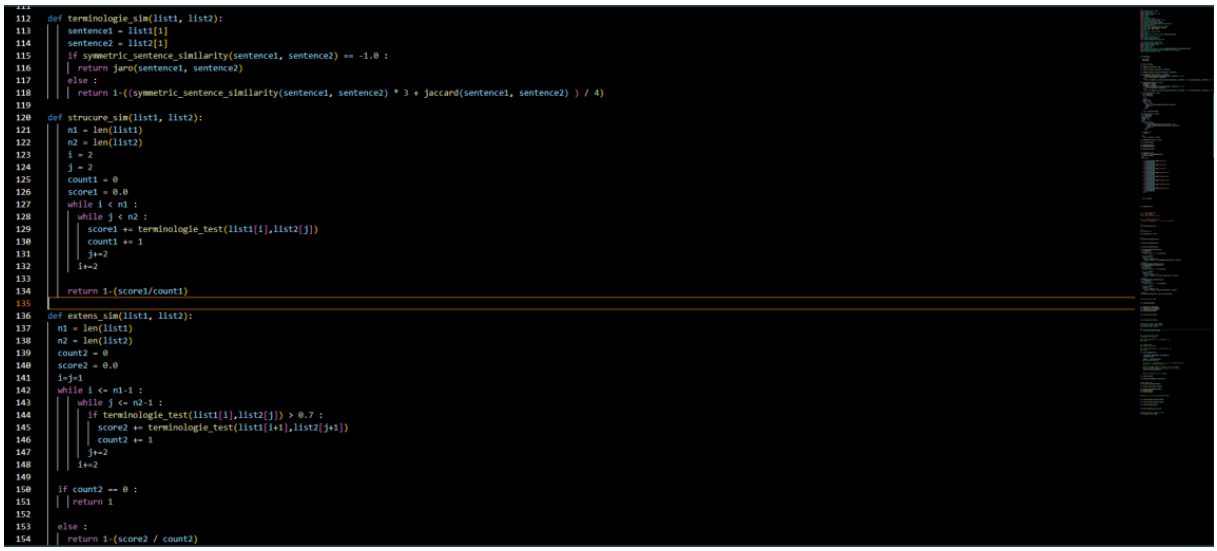


FIGURE 3.9 – Consultation des résultats.

Le code ci-dessous, représente comment nous avons calculé la similarité sémantique.



```
def terminologie_sim(list1, list2):
    sentence1 = list1[1]
    sentence2 = list2[1]
    if symmetric_sentence_similarity(sentence1, sentence2) == -1.0 :
        return jaro(sentence1, sentence2)
    else :
        return 1-((symmetric_sentence_similarity(sentence1, sentence2) * 3 + jaccard(sentence1, sentence2) ) / 4)

def strucure_sim(list1, list2):
    n1 = len(list1)
    n2 = len(list2)
    i = 2
    j = 2
    count1 = 0
    score1 = 0.0
    while i < n1 :
        while j < n2 :
            score1 += terminologie_test(list1[i],list2[j])
            count1 += 1
            j+=2
        i+=2

    return 1-(score1/count1)

def extens_sim(list1, list2):
    n1 = len(list1)
    n2 = len(list2)
    count2 = 0
    score2 = 0.0
    i=j=1
    while i <= n1-1 :
        while j <= n2-1 :
            if terminologie_test(list1[i],list2[j]) > 0.7 :
                score2 += terminologie_test(list1[i+1],list2[j+1])
                count2 += 1
            j+=2
        i+=2

    if count2 == 0 :
        return 1

    else :
        return 1-(score2 / count2)
```

FIGURE 3.10 – Code de calcul de la similarité.

3.4 Tests et évaluation

Afin de tester la fiabilité et la validité de notre système, nous avons choisi un graphe RDF d'un wiki portant sur les jeux vidéo dans l'univers Star Wars comme dataset. Nous avons effectué la catégorisation et évalué sa qualité.

3.4.1 Résultats expérimentaux et discussion

Dans cette section, nous présentons les résultats obtenus par les différentes mesures de distance sémantique.

Deux ressources avec leurs prédicats et objets ont été sélectionnées pour effectuer les tests. (table 3.1).

Ressource	prédicat	objet
Zythmnr	allegiance	Brotherhood of the Sith
	race	Massassi
	abstract	Zythmnr was a male Massassi Sith apprentice
Vector Hyllus	likes	Diplomacy helping people exploring alien cultures
	race	human

TABLE 3.1 – Exemple de deux ressources RDF

Les formules de distance sont utilisées pour calculer les distances terminologique, structurelle, extensionnelle et la synthèse de toutes les distances, la distance sémantique entre les deux ressources présentées dans la table. 3.2.

Les types de distances	Distance
Distance terminologique	0.85
Distance structurelle	0.86339
Distance extensionelle	1.00
Distance sémantique	**0.90446**

TABLE 3.2 – Formules de distance et leurs résultats

Interprétation des résultats :
Dans le tableau 3.2 nous remarquons les résultats des formules de distance par rapport aux deux ressources présentées dans le tableau 3.1.

— La distance terminologique compare les termes "Zythmnr" et "Vector Hyllus" qui sont des nom propre et ne sont pas comparés lexicalment. La comparaison syntaxique donne une distance élevée car les deux chaînes de caractères on peu en commun.

— La distance structurelle compare les prédicats des deux termes. Nos ressources ont un seul prédicat en commun résultant en une distance élevée.

— La distance extensionnelle nous donne la distance maximale possible. Cela est dû au manque d'objet similaire. Le seul prédicat équivalent entre les deux ressources a deux extensions très différentes.

3.4.2 Mesures d'évaluation utilisées

Les mesures d'évaluation sont des métriques utilisées pour quantifier la qualité de la catégorisation. Pour évaluer cette qualité nous avons calculé la largeur de silhouette, qui est la moyenne des coefficients de silhouette. Ces derniers sont calculés en utilisant :

— La distance moyenne intra-cluster a.

— La distance moyenne du cluster le plus proche b.

Selon la formule suivante :

$$\textbf{Coefficient de silhouette} = \frac{(b-a)}{max(a,b)}$$

Pour clarifier, b est la distance entre un élément et le cluster le plus proche dont l'élément ne fait pas partie. Notez que le coefficient de silhouette n'est défini que si le nombre de clusters plus le bruit est supérieur à un.

Epsilon	0.030	0.370	0.046	0.080	0.100	0.060
MinPts	4	20	6	5	7	9
Largeur de silhouette	**-0.068**	**0.170**	**0.559**	**0.629**	**0.640**	**0.530**

TABLE 3.3 – Valeur de la métrique d'évaluation par rapport a différents paramètres de DBSCAN

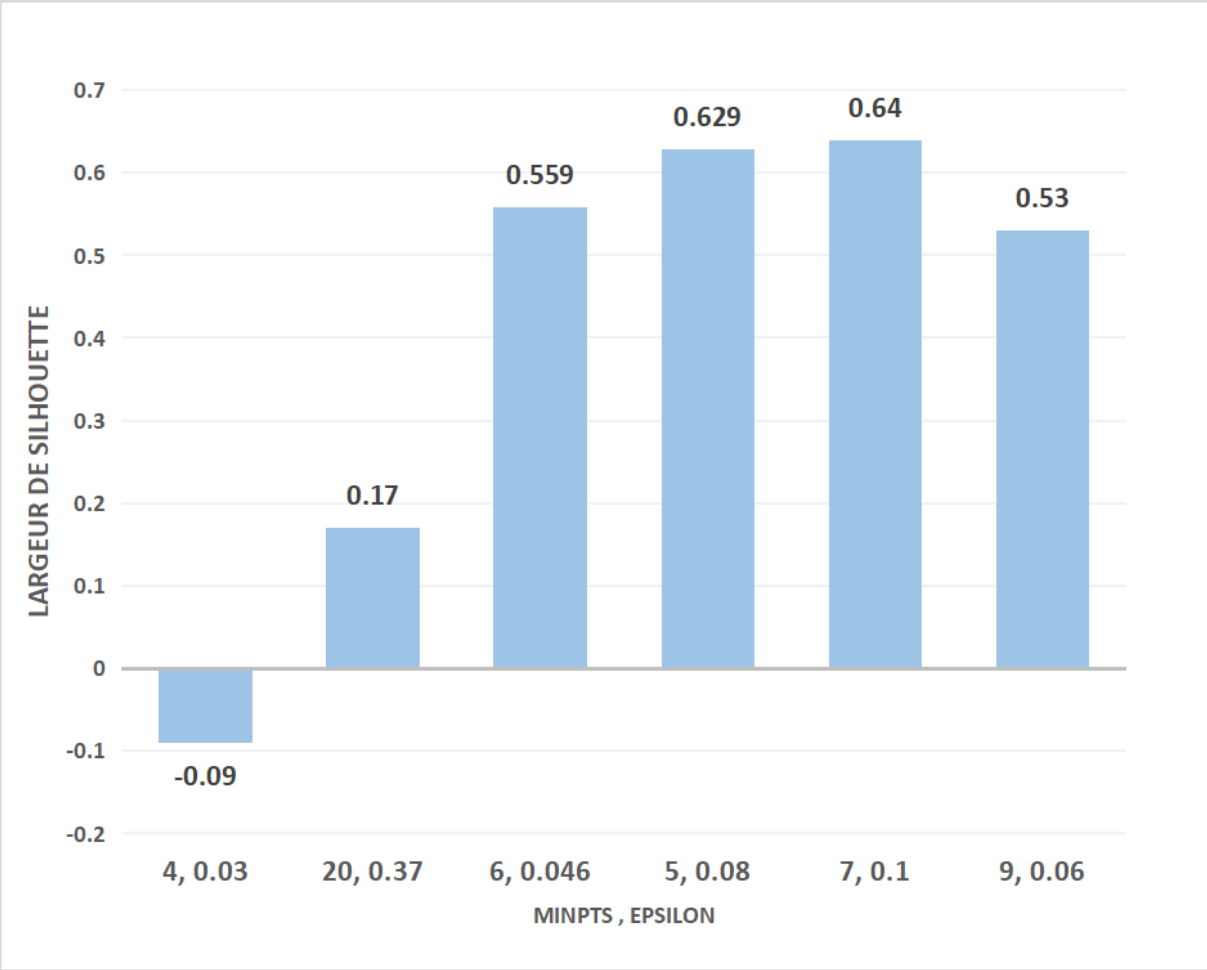


FIGURE 3.11 – Histograme illustrant l'évaluation selon différents paramètres de DBSCAN.

Interprétation des résultats :

Dans l'exemple illustré dans la table 3.3 et la figure 3.11, on remarque une variation de la métrique d'évaluation selon les paramètres de DBSCAN.

— Dans le cas où les paramètres sont trop petits, on obtient une mauvaise largeur de silhouette, car la catégorisation crée un grand nombre de clusters où les éléments ne sont pas placés dans leur cluster optimal.

— Une valeur des paramètres trop grande cause le même problème dans le sens inverse, où la catégorisation place la majorité des points dans le même cluster.

— Entre ces deux extrêmes, on obtient des valeurs de largeur de silhouette acceptables. Les valeurs initiales ont été choisies en utilisant les méthodes décrites dans l'étape de choix des paramètres initiaux du chapitre deux. Ce résultat est représenté par la troisième colonne du tableau et de la figure.

— En itérant sur les paramètres de l'algorithme DBSCAN, nous obtenons une multitude de résultats parmi lesquels nous avons choisi trois pour montrer les possibles variations de la métrique d'évaluation. On remarque un pic où le minpts est égal à douze et epsilon est égal à zéro virgule douze. On remarque également que la largeur de silhouette commence sa descente lorsque la valeur de minpts atteint trente.

3.5 Conclusion

Dans ce chapitre nous avons décrit les outils et l'environnement utilisés pour le développement de notre système. Le résultat final de notre travail a été présenté sous forme d'interface.

Nous avons aussi testé et validé nos résultats en utilisant une métrique d'évaluation (largeur de silhouette) et itéré la catégorisation en se basant sur le résultat de la métrique et la représentation graphique.

Chapitre 4

CONCLUSION ET PERSPECTIVE

Le web de données contient une grande masse de ressources web liées entre elles. Les données liées permettent de facilement partager la connaissance à travers le format RDF. La catégorisation des ressources RDF offre la possibilité de catégoriser les ressources web de manière sémantique contextuelle. L'objectif de ce projet était de proposer un système de catégorisation sémantique des données liées.

Dans le premier chapitre, nous avons défini le web et son évolution jusqu'à sa troisième version. Le web sémantique avec ses couches, à savoir l'Unicode et l'URI, l'XML, le RDF, le RDFS, les ontologies, l'OWL, le SPARQL, ainsi que les couches de logique, de preuve et de confiance. Les données liées avec des exemples de leur utilisation dans les domaines de la recherche sémantique, du e-commerce et des applications médicales. Nous avons ensuite abordé le domaine de la catégorisation, qui consiste à placer les données dans des groupes avec d'autres données similaires. Nous avons défini ses types, qui sont la catégorisation hiérarchique, partitionnelle, et celle basée sur la densité. Nous avons ensuite défini les mesures de similarité en général avant d'aborder les mesures de similarité pertinentes à notre travail, qui sont la similarité terminologique, structurelle et extensionnelle. Pour clore ce chapitre, nous avons présenté les travaux existants et analysé leurs caractéristiques et leurs limites.

Pour répondre à notre problématique, nous avons conceptualisé un système répondant aux besoins de catégoriser les données liées de manière sémantique dans le deuxième chapitre. Notre système comprend trois phases. La première, appelée phase de pré-catégorisation, contient les étapes de préparation et de normalisation des données. La deuxième, appelée phase de catégorisation est la phase la plus lourde du système, elle contient les étapes de calcul des matrices de similarité, où l'on calcule la matrice terminologique en utilisant WordNet avec l'indice de Jaccard, la matrice structurelle en comparant les libellés des prédicats, et la matrice extensionnelle en comparant les termes des objets de prédicats équivalents. nous employant aussi la réduction de dimensionnalité et la catégorisation elle-même avec l'algorithme DBSCAN. La dernière phase est la phase de visualisation et enregistrement des résultats.

Le dernier chapitre est consacré à l'implémentation et l'évaluation du système. Nous avons présenté les outils et environnement de développement utilisés pour réaliser le système. Ensuite nous avons présenté l'interface graphique créée pour rendre le système plus accessible.

Nous avons testé et évaluer notre système par rapport aux tests réalisés afin d'évaluer les différentes formules de distance proposés dans notre travail, avec des exemples. Nous avons évalué également la catégorisation avec l'utilisation de la largeur de silhouette, les résultats sont encourageants.

En ce qui concerne les perspectives de ce travail, nous souhaitons optimiser davantage le système pour réduire le temps de calcul et le rendre directement accessible sur le web, ainsi d'utiliser des modèles mathématiques pour combiner les différentes valeurs de similarité.

Bibliographie

[1] G. Antoniou and F. Van Harmelen. *A semantic web primer*. MIT press, 2004.

[2] S. A. Bamatraf and R. A. BinThalab. Clustering rdf data using k-medoids. In *2019 First International Conference of Intelligent Computing and Engineering (ICOICE)*, pages 1–8. IEEE, 2019.

[3] S. Bechhofer, F. Van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, L. A. Stein, et al. Owl web ontology language reference. *W3C recommendation*, 10(2) :1–53, 2004.

[4] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific american*, 284(5) :34–43, 2001.

[5] C. Bizer, T. Heath, and T. Berners-Lee. Linked data : The story so far. In *Semantic services, interoperability and web applications : emerging concepts*, pages 205–227. IGI global, 2011.

[6] R. Bouhamoum, K. Kellou-Menouer, S. Lopes, and Z. Kedad. Scaling up schema discovery for rdf datasets. In *2018 IEEE 34th International Conference on Data Engineering Workshops (ICDEW)*, pages 84–89. IEEE, 2018.

[7] M. E. Celebi. *Partitional clustering algorithms*. Springer, 2014.

[8] N. Choudhury. World wide web and its journey from web 1.0 to web 4.0. *International Journal of Computer Science and Information Technologies*, 5(6) :8096–8100, 2014.

[9] K. Christodoulou, N. W. Paton, and A. A. Fernandes. Structure inference for linked data sources using clustering. In *Proceedings of the Joint EDBT/ICDT 2013 Workshops*, pages 60–67, 2013.

[10] T. A. Costa and J. P. Leal. Publishing linked data with dapress. 2013.

[11] S. Decker, S. Melnik, F. Van Harmelen, D. Fensel, M. Klein, J. Broekstra, M. Erdmann, and I. Horrocks. The semantic web : The roles of xml and rdf. *IEEE Internet computing*, 4(5) :63–73, 2000.

[12] Q. Du, Z. Dong, C. Huang, and F. Ren. Density-based clustering with geographical background constraints using a semantic expression model. *ISPRS International Journal of Geo-Information*, 5(5) :72, 2016.

[13] S. Eddamiri, A. Benghabrit, et al. An improved rdf data clustering algorithm. *Procedia computer science*, 148 :208–217, 2019.

[14] M. N. Elbatta. An improvement for dbscan algorithm for best results in varied densities. In *The Islamic University-Gaza Palestine*, 2012.

[15] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996.

[16] C. Fellbaum. Wordnet. In *Theory and applications of ontology : computer applications*, pages 231–243. Springer, 2010.

[17] T. R. Gruber. A translation approach to portable ontology specifications. *Knowledge acquisition*, 5(2) :199–220, 1993.

[18] M. Gruninger and M. Fox. "the logic of enterprise modelling"in brown, j and o'sullivan, d, editors, reengineering the enterprise 83±98, 1995.

[19] Q. Guo, W. Ji, S. Zhong, and E. Zhou. The analysis of the ontology-based k-means clustering algorithm. In *Conference of the 2nd International Conference on Computer Science and Electronics Engineering (ICCSEE 2013)*, pages 734–737. Atlantis Press, 2013.

[20] E. K. Jacob. Ontologies and the semantic web. *Bulletin of the American Society for Information Science and Technology*, 29(4) :19–19, 2003.

[21] S. C. Johnson. Hierarchical clustering schemes. *Psychometrika*, 32(3) :241–254, 1967.

[22] C. A. Khanzode and R. D. Sarode. Evolution of the world wide web : from web 1.0 to 6.0. *International journal of Digital Library services*, 6(2) :1–11, 2016.

[23] A. Léger and J. Charlet. Applications du web sémantique. *France Telecom R&D, Mission de recherche STIM, AP-HP & INSERM ERM*, 202.

[24] G. Madhu, D. A. Govardhan, and D. T. Rajinikanth. Intelligent semantic web search engines : a brief survey. *arXiv preprint arXiv :1102.0831*, 2011.

[25] D. L. McGuinness. Owl web ontology language overview w3c recommendation 10 february 2004. *http ://www. w3. org/TR/2004/REC-owl-features-20040210/*, 2007.

[26] E. Negre. Comparaison de textes : quelques approches. 2013.

[27] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn : Machine learning in python. *the Journal of machine Learning research*, 12 :2825–2830, 2011.

[28] J. Pérez, M. Arenas, and C. Gutierrez. Semantics and complexity of sparql. *ACM Transactions on Database Systems (TODS)*, 34(3) :1–45, 2009.

[29] L. Qi, H. T. Lin, and V. Honavar. Clustering remote rdf data using sparql update queries. In *2013 IEEE 29th International Conference on Data Engineering Workshops (ICDEW)*, pages 236–242. IEEE, 2013.

[30] G. Van Rossum and F. L. Drake Jr. *Python tutorial*, volume 620. Centrum voor Wiskunde en Informatica Amsterdam, The Netherlands, 1995.

[31] D. Yassir-Montet. *Une Approche hybride de gestion des connaissances basée sur les ontologies : application aux incidents informatiques*. PhD thesis, Lyon, INSA, 2006.