MASTER EN SCIENCES INFORMATIQUES

Étude des principes d'interprétation sémantique tabulaire dans l'Open Data

Cabello, Anthony; Verhelle, Romain

Award date:
2021

Awarding institution:
Universite de Namur

Université de Namur
Faculté d'informatique
Année académique 2020–2021

Etude des principes d'interprétation Sémantique tabulaire dans l'Open Data

Anthony Cabello
Romain Verhelle

Promoteur : _____(Signature pour approbation du dépôt - REE art. 40)
Anthony Cleve

Co-promoteur : Rabeb Abida

Mémoire présenté en vue de l'obtention du grade de
Master en Sciences Informatiques.

Remerciements

D'abord, nous tenons à remercier notre promoteur, le Professeur Anthony Cleve et la doctorante Rabeb Abida qui nous ont conseillé et guidé dans nos recherches et notre partie pratique tout au long de la rédaction de ce mémoire sur un sujet passionnant ; ainsi que Joëlle Harnïe pour la relecture de notre travail ainsi que les personnes qui nous ont soutenus durant cette période compliquée.

Enfin, nous remercions la Faculté d'Informatique de l'Université de Namur, son doyen Wim Vanhoof, son vice-doyen Laurent Schumacher et l'ensemble du personnel pour nous avoir permis d'évoluer et de progresser davantage dans notre milieu personnel et professionnel.

Résumé

Ces dernières années, le monde de l'Open Data s'agrandit de jour en jour. Les bases de données ne cessent de croitre ainsi que l'intérêt pour l'annotation de tableaux sur le Web. Notre travail s'intéresse surtout aux Linked open data (LOD). C'est un système qui permet de relier différentes sources de données entre elles. Ces dernières sont libres d'accès et d'utilisation permettant à tout individu de pouvoir les utiliser. L'interprétation sémantique (STI) des tables est un processus qui exploite la sémantique des bases de connaissances (KB) afin de pouvoir annoter les colonnes, cellules et relations d'une table. Ce mémoire commence par synthétiser la définition, les méthodes et les meilleurs outils existants de cette interprétation. Il décrit ensuite le développement d'une solution qui permet d'annoter un ensemble de tableaux de manière efficiente et de les comparer entre eux grâce à DBpedia. Cette solution se base sur des outils existants. Elle comble certaines lacunes de ces derniers et amène une preuve de concept d'un procédé complet d'annotation et de comparaison de tableaux. Enfin nous discuterons des résultats, des limitations et des perspectives d'amélioration pour les travaux futurs.

Mot-clefs: Semantic Table Interpretation, STI tool, Dataset Matching, HTML Table, Open Data, Knowledge Graph, LOD.

Absract

In recent years, the world of Open Data is growing day after day. Databases is growing up and so does the interest in annotating tables on the Web. Our work focuses on Linked open data (LOD), a system that allows to link different sources of data between them. These data are free to access and use, allowing everyone to use it. Semantic Table Interpretation (STI) is a process that exploits the semantics of knowledge bases (KB) to annotate columns, cells and relations of a table. This paper begins by summarizing the definition, methods and best existing tools for this interpretation. It then describes the development of a solution that allows to annotate a set of tables in an efficient way and to compare them with DBpedia. This solution is based on existing tools. It fills in some of the gaps in these tools and provides a proof of concept for a complete table annotation and comparison process. Finally, we will discuss the results, limitations and perspectives for future work.

Key-words: Semantic Table Interpretation, STI tool, Dataset Matching, HTML Table, Open Data, Knowledge Graph, LOD.

TABLES DES MATIERES

Liste des abréviations, sigles et acronymes

Abréviation	**Signification**
CEA	*Cell Entity Annotation* Correspondance entre une cellule et une entité d'un KG.
CTA	*Cell Entity Annotation* Assignement d'un type d'un KG à une colonne.
CPA	*Column Predicate Annotation* Assignement d'une propriété d'un KG avec une relation entre deux colonnes.
GS	Gold standard
KB	Knowledge Base
KG	Knowledge Graph
KGE	Knowledge Graph Embedding
LOD	Linked open data
STI	Semantic Table Interpretation
Semantic Table Interpretation	Principe permettant de donner une annotation sémantique à un tableau.
URI	*Uniform Resource Identifier*
dbr	Dbpedia ressource
dbo	Dbpedia ontology
dbp	Dbpedia properties

1 Chapitre 1 : Introduction

Depuis plusieurs années, on observe une augmentation significative de fournisseurs en données ouvertes liées, ***Linked Open Data*** (LOD) en anglais. Ces derniers, en respectant les principes de ces données, ont réussi à créer espace de données appelé le **Web Sémantique** [1].

En 2020, un total de 25 millions de tables relationnelles et de 500 millions de pages Web ont été trouvées sur le Web [2]. Cet engouement créé va donc amener certains défis à relever pour faciliter son utilisation et sa découverte par le monde [3].

Ce travail va s'intéresser au challenge de la compréhension sémantique de jeux de données qu'on peut retrouver dans des tableaux CSV ou bien encore dans des tableaux HTML. Un principe de compréhension et d'interprétation est fortement utilisé dans ce cas-ci, celui de l'interprétation tabulaire sémantique, ***Semantic Table Interpretation*** en anglais (**STI).** L'approche STI permet, grâce à un ensemble de connaissances, de définir une table, son contexte, et, de pouvoir lier cette dernière à d'autres données se trouvant sur le Web [2].

Le Web grandissant de jour en jour, le principal problème majeur lié à cette interprétation est de pouvoir donner un sens sémantique à nos tableaux, et ce, de manière précise et performante.

Le concept de STI est au cœur de ce travail, il met en relation un ou plusieurs tableaux avec des bases de données. Grâce à ces bases de données, il est alors possible de donner un sens sémantique aux tableaux via des annotations et également d'analyser et de comparer les données du tableau pour éventuellement les intégrer.

Ce concept va donc permettre d'identifier les colonnes et les cellules des tableaux reçus via plusieurs grandes étapes, l'annotation de colonne (CTA), l'annotation de cellule (CEA) et l'annotation de propriété (CPA).

Ce travail sera divisé en différents points. Nous allons dans un premier temps établir une introduction et un contexte pour permettre une compréhension des éléments clés de l'open data pour permettre une compréhension à un lecteur moins aguerris [Chapitre 1].

Ensuite, nous présenterons l'état de l'art qui sera découpé en plusieurs chapitres. Le premier [Chapitre 2] explique notre méthode de recherche, d'analyse et de traitement des informations. Notre question de recherche principale a été divisées en plusieurs sous-questions dont les chapitres suivants présentent chacun une réponse à l'une d'elle. Nous présenterons donc la définition du STI [Chapitre 3], les méthodes d'interprétation existantes [Chapitre 4] et les meilleurs outils existants [Chapitre 4.13].

Afin de compléter la partie théorique et d'illustrer notre recherche par un cas pratique, nous allons présenter le développement d'un nouvel outil qui met en avant une interprétation sémantique fiable et la liaison de schémas entre des tableaux [Chapitre 5.10]. Cet outil, basé sur des méthodes et outils existants, nous allons démontrer comment analyser et traiter de manière efficace ces tableaux et ensuite être capable de les lier entre eux.

Finalement, nous terminerons avec ce qu'il reste à améliorer dans la méthode que nous avons développée [Chapitre 9] et une conclusion décrivant les points faibles et forts de notre recherche [Chapitre 10].

1.1 Contexte

1.1.1 Introduction

Les données ouvertes liées, *linked open data* en anglais (LOD), sont des données accessibles par tous via le Web. Ces dernières peuvent être publiées et réutilisées sans permission et sans devoir payer un tiers. Elles sont récoltées dans le Web des données pour être utilisées dans différentes applications Web. Le principe le plus utilisé pour le moment est de charger les données dans un dépôt local et ensuite de les utiliser dans des applications spécifiques [3].

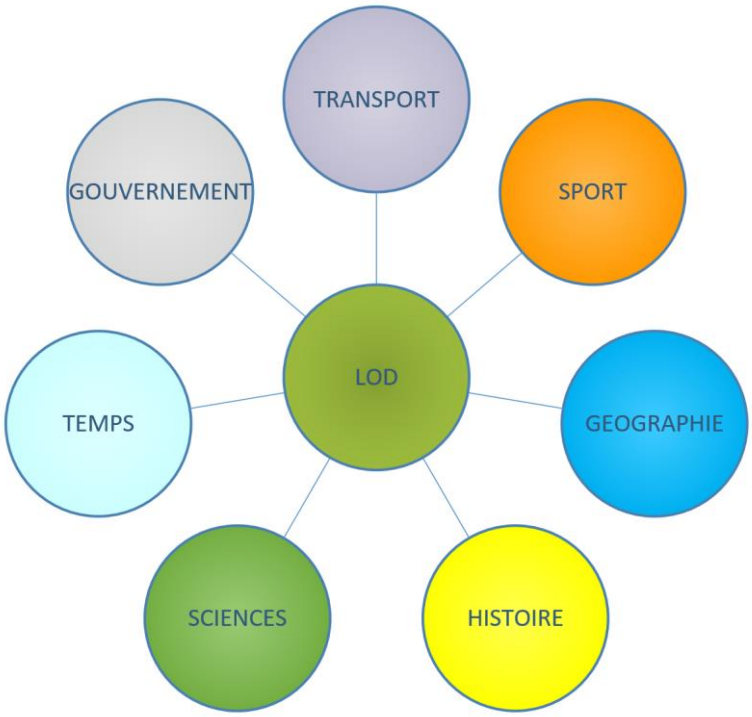


Figure 1 Exemples de thèmes open data

Les données ouvertes respectent plusieurs principes comme la **disponibilité**, la **réutilisation**, la **distribution** et la **participation universelle**.

- Disponibilité : Les données sont totalement accessibles et téléchargeables sur le Web.
- Réutilisation et distribution : Les données sont transmises dans un format qui peut être facilement réutilisable et facile à distribuer sur le Web.
- Participation universelle : Tout le monde doit être capable d'utiliser ces données sans aucune discrimination.

Ces concepts permettent de mettre en place l'interopérabilité qui est le moyen de lier plusieurs sources de données [4]. On peut voir l'espace LOD comme un espace de données unique, ou plus simplement en tant qu'unique base de données à l'échelle du Web disponible pour tout le monde et dont la force principale est leur possible utilisation directe par des applications tierces [3].

1.1.2 Le Web Sémantique

En 2001, Tim Berners-Lee a présenté le Web Sémantique, comme un *Web of data* qui contient des documents mais aussi des entités (des personnes, des organisations) et la représentation des relations qui existent entre celles-ci [3].

Les composants essentiels du Web Sémantique sont les bases de connaissances et les graphes de connaissances.

- Une base de connaissances, *knowledge base* en anglais (KB), stocke des informations factuelles sur des entités.
- Un graphe de connaissances, *knowledge graph* en anglais (KG), est un graphe contenant des entités représentées par des nœuds et des relations qui sont représentés par des arêtes. C'est une base de connaissances avec des relations.

Pour mettre en pratique le principe du Web Sémantique, Tim Berners-Lee a aussi mis en place quatre principes [3] qui permettent de définir les LOD. C'est un moyen efficace pour relier différentes sources entre elles. En faisant ça, il est possible d'enrichir considérablement les informations sur le Web. Par exemple, un utilisateur va avoir accès à des données qu'il n'avait pas soupçonnées lors de sa recherche initiale. Contrairement à la vision à part entière du Web Sémantique, les LOD concernent principalement la publication de données structurées respectant ces différents principes.

Les quatre principes sont les suivants :

a. Identification URI

Tous les éléments d'un jeu de données doivent être identifier par un URI, de l'anglais *Uniform Resource Identifier,* qui est une chaîne de caractères permettant d'identifier sans ambiguïté une ressource particulière [1].

b. URI déréférençable

Tous les URI doivent être déréférençables. Lorsqu'un client http peut rechercher un URI à l'aide du protocole HTTP et récupérer une description de la ressource, cela s'appelle un URI déréférençable. Les URI déréférençables s'appliquent aux URI utilisés pour identifier des documents HTML classiques [1].

c. RDF/SPARQL

Lors de la consultation d'un URI, des informations utiles sont fournies, en utilisant les normes (RDF, SPARQL). Un fichier RDF est une syntaxe qui a été définie par W3C pour définir un graphique RDF comme un document XML. Un RDF contient un triplet de valeur :

- Le sujet qui représente la ressource à analyser.
- Le prédicat qui représente une propriété de la ressource.
- L'objet qui représente une donnée.

Le sujet est lié à l'objet via un le prédicat comme le montre la Figure 2.

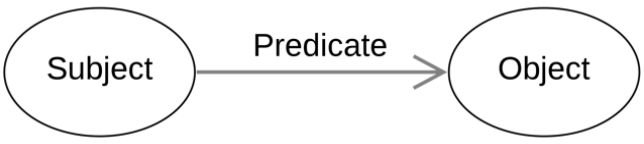


Figure 2 : Image représentant un triplet RDF

Un exemple concret de fichier RDF peut se faire via une phrase. Par exemple, « l'Atomium est situé à Bruxelles ». Le sujet correspond à « l'Atomium », le prédicat correspond à « est situé à » et l'objet est « Bruxelles ».

SPARQL est le langage utilisé pour parcourir les fichiers RDF. Les applications peuvent accéder aux LOD sur le Web en interrogeant un SPARQL ENDPOINT [1].

Le SPARQL Endpoint est un service Web qui respecte le protocole SPARQL, c'est une manière de transférer des requêtes SPARQL de clients à un service Web capable de l'exécuter et de renvoyer le jeu de données résultat [5].

d. Liens URI

Inclure des liens vers d'autres URI afin qu'ils puissent découvrir encore plus de donnée sur le Web et donc avoir une vue plus complète [1].

Ces quatre principes sont résumés dans la Figure 3 :

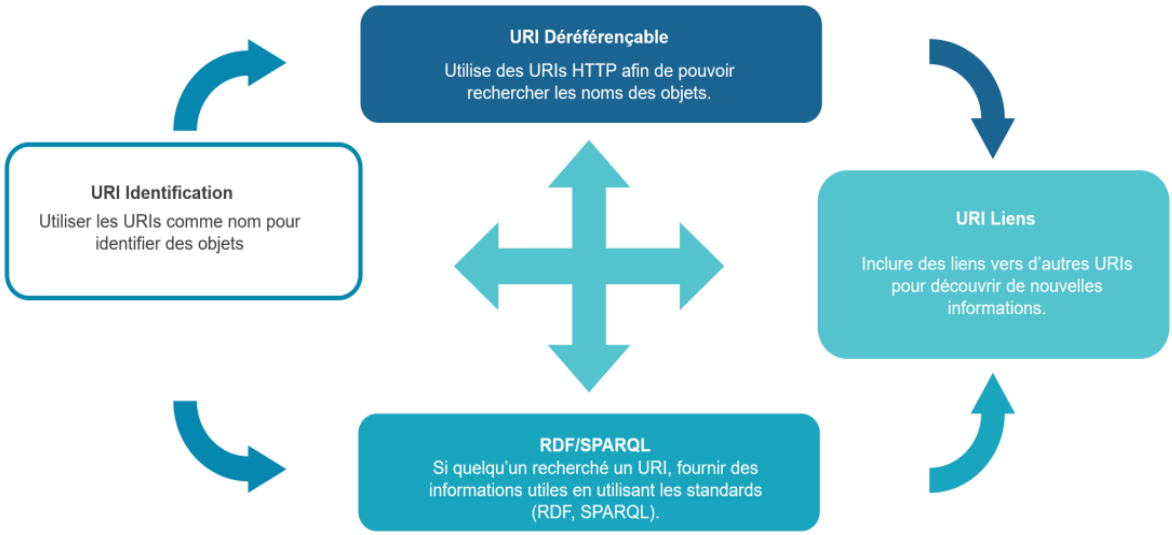


Figure 3 : Les quatre principes du Linked Open Data

En utilisant ces concepts, il est possible de relier des jeux de données entre eux et d'enrichir considérablement les données mises à disposition. Par exemple, plutôt que d'avoir accès uniquement aux données se trouvant dans un jeu de données comme DBpedia, il est possible d'utiliser des données d'autres ensembles dans une même recherche.

1.1.3 Jeux de données composants le LOD

La Figure 4 montre différents ensembles qui constituent le Web of data. Certains de ces ensembles sont utilisés régulièrement comme par exemple DBpedia, Wikidata, OpenCyc, etc.

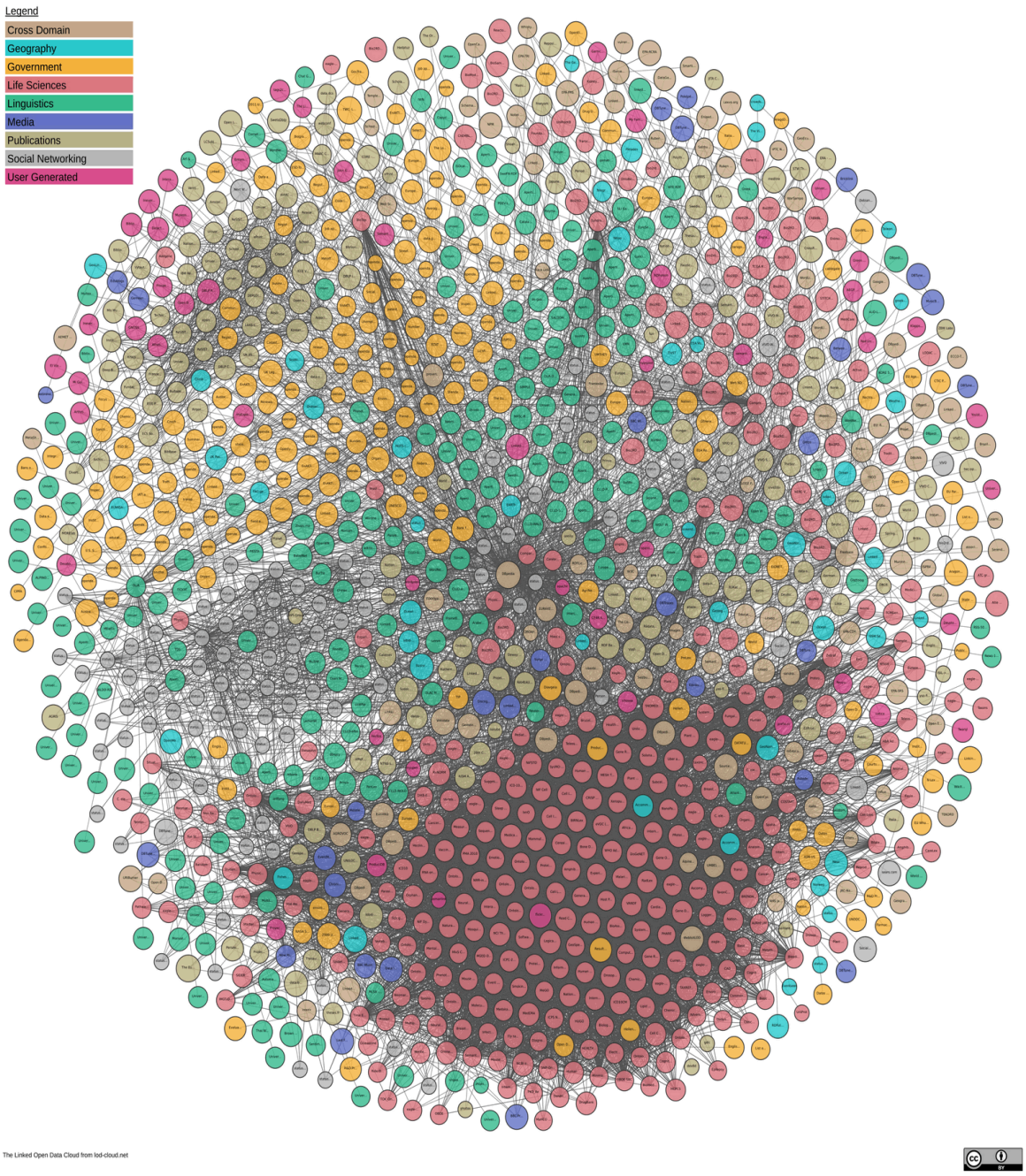


Figure 4 : Représentation de Knowledge Graphs interconnecté[1] pour l'année 2021.

[1] https://lod-cloud.net/

DBpedia

DBpedia[2] [6] est un catalogue qui est mis à jour une fois par an. Il est créé à partir d'informations structurées provenant de Wikipédia comme par exemple : des tables d'informations (infobox), des coordonnées géographiques, des liens externes, etc. La Figure 4 montre que DBpedia est le centre du système LOD et qu'il est lié à un grand nombre d'autres ensembles tel que Freebase, YAGO, OpenCyc, etc.

Son rôle central dans le système LOD fait qu'il est largement utilisé par la communauté scientifique pour la création d'applications, d'algorithmes ou bien d'outils [7]. Il est composé d'un ensemble de fichier RDF pointant vers des sources externes.

A partir de DBpedia, plusieurs sections peuvent être utilisées :

- Les ressources[3] (préfixe utilisé : dbr) permettent de représenter un article Wikipédia. Une ressource DBpedia et un article Wikipédia sont liés, par exemple, l'article Barack Obama a comme ressource Dbpedia dbr:Barack_Obama.
- Les propriétés[4] (préfixe utilisé : dbp) représentent les propriétés des infobox de Wikipédia. Les infobox se retrouvent sur la partie supérieur droite des pages Wikipédia.
  Par exemple :

Fonctions	
44e président des États-Unis	
20 janvier 2009 – 20 janvier 2017 (8 ans)	
Élection	4 novembre 2008
Réélection	6 novembre 2012
Vice-président	Joe Biden
Gouvernement	Administration Obama
Prédécesseur	George W. Bush
Successeur	Donald Trump

Figure 5 : InfoBox de Barack Obama sur le site Wikipédia

Cela représente une liste d'attributs qui résume l'article Wikipédia correspondant. Comme par exemple dbp:predecessor correspond au prédécesseur de Barack Obama, George W.Bush.

[2] https://www.dbpedia.org/
[3] http://dbpedia.org/resource/
[4] http://dbpedia.org/property/

- Les ontologies[5] (préfixe utilisé : dbo) qui sont décrites par des propriétés (dbp). Comme par exemple dbo : person, dbo : Politician… L'ensemble des ontologies de DBpedia peut être retrouvée ici[6].

Freebase

Freebase [8] permet aux utilisateurs finaux de fournir/modifier des données structurées dans des KG. Freebase intègre des données de Wikipédia, MusicBrainz et d'autres catalogues. Freebase a été intégré dans Wikidata [7], certaines méthodes utilisent ce KB, il est donc important de bien le comprendre et de l'expliquer.

Freebase était utilisé comme base pour le KG de Google [9]. Cette base de connaissance est créée à partir d'objets, de faits, de types et de propriétés. Chaque objet est identifié via un Machine ID (Mid), il peut avoir un ou plusieurs types et il peut utiliser les propriétés de ces types dans le but de fournir des faits [9].

En reprenant l'exemple de Barack Obama, l'objet Barack Obama possède le mid : /m/02mjmr et le type /government/us_president (il a bien sur d'autres types) ce qui permet de définir un fait avec la propriété suivante : government/us_president/presidency_number la valeur de cette propriété est le 44 puisque Barack Obama est le 44ème président des Etats-Unis.

Freebase utilisait des valeurs composées (CVT) lorsqu'il y a plus d'une relation avec un objet. Par exemple :

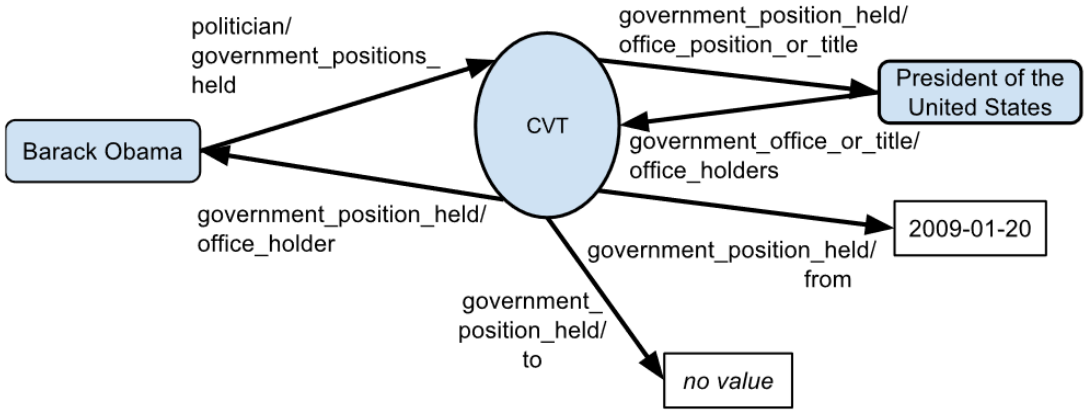


Figure 6 : Relation d'objets dans Freebase. Image reproduite de [9]

[5] http://dbpedia.org/ontology/

[6] http://mappings.dbpedia.org/server/ontology/classes/

Ici pour la position de président, Barack Obama est président des USA, il a commencé son mandat le 20 janvier 2009, sa date de fin n'est pas connue puisqu'au moment où l'article a été écrit il était encore président.

Wikidata

L'objectif du projet est de fournir des données qui peuvent être utilisées par tous les projets Wikipédia, y compris Wikipédia. Wikidata ne stocke pas seulement des faits, mais aussi les sources correspondantes, de sorte que la validité des faits peut être vérifiée. Les utilisateurs peuvent ajouter et modifier les informations. De plus, le schéma est maintenu et étendu sur la base d'accords communautaires [7].

Wikidata fonctionne par les deux notions d'*item* et de *statement* :

- Un *item* correspond à une entité peut posséder plusieurs labels, descriptions, des alias dans différentes langues, des liens vers d'autres entités et il possède un identifiant unique appelé « qid » [9].
- Un *statement* se compose d'une affirmation (claim) et de 0 ou plusieurs références à cette affirmation. L'affirmation est composée d'une paire de valeur. Par exemple, dans la figure 6, le taxon correspond à Pantera Leo et sa paire, qui est son qualificateur, possède comme valeur Carl Linnaeus qui n'est autre que l'auteur du taxon.

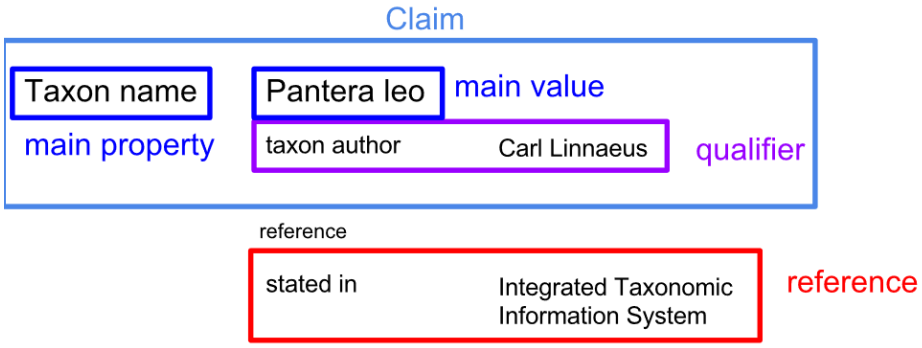


Figure 7 : Statement wikidata. Image reproduite de [9]

Les deux derniers KB n'ont été que peu utilisés dans ce travail donc nous nous contentons juste de les expliquer brièvement.

OpenCyc

OpenCyc stocke des millions d'informations concernant tout ce qu'il y a du sens. Par exemple, chaque arbre est une plante. Elle permet à des IA de raisonner d'une manière similaire à un être humain [7].

Yago

YAGO comprend des informations extraites de Wikipédia (par exemple, catégories, redirections, infobox), de WordNet [23] (par exemple, syntaxes, hyponymie), et de GeoNames. [7]

Le Web des données se compose d'éléments comme ceux-ci pour créer un environnement toujours plus complet et utile.

1.1.4 Interprétation Sémantique

Il est intéressant de pouvoir enrichir les bases de connaissances pour contribuer à leur utilisation par un large éventail d'applications. Afin de pouvoir intégrer de nouvelles données, il est nécessaire de pouvoir les interpréter.

La sémantique est une branche de la linguistique qui vise à étudier les sens du langage. Son analyse permet de traiter et d'évaluer la compréhension des phrases et des mots, elle est donc critique dans le traitement du langage naturel [10].

Le concept d'alignement d'une table sur une ontologie grâce à son interprétation sémantique est connue sous le nom d'interprétation sémantique tabulaire, *Semantic Table Interpretation* en anglais (STI). Cette activité permet la transformation de données textuelles en formats lisibles par machine pour permettre l'exécution de diverses tâches d'intelligence artificielle, par ex. recherche sémantique et extension de jeu de données [2]. Il s'agit d'un processus d'annotation des éléments du tableau.

1.1.5 Conclusion

Ce chapitre permet de comprendre ce qu'est l'open data, c'est-à-dire un domaine récent qui représente les données accessibles par tous via le Web et également le Web Sémantique comme un Web of data contenant des entités et la représentation des relations qui existent entre elles.

Le Web Sémantique est composé par des bases de connaissances (KB) ou des graphes de connaissances (KG) et il respecte quatre grands principes : identification URI, URI déréférencable, RDF Sparql et liens URI.

A l'heure actuelle, il existe de nombreuses bases de connaissances dont chacune d'entre elles peuvent être utilisées dans la découverte et la liaison de données existantes ou nouvelles. Par exemple, DBpedia, YAGO, Freebase, Wikidata ou encore OpenCyc.

Le concept permettant la compréhension d'une table en vue de son utilisation et son intégration dans une base de connaissance est appelé *Semantic Table Interpretatio*n (STI).

1.2 Motivation

Ce travail a pour but l'étude de l'interprétation sémantique tabulaire afin de permettre la comparaison entre schémas dans les données ouvertes et ainsi permettre leur intégration dans des bases de connaissances. En faisant cela, il est possible de faire correspondre deux ou plusieurs fichiers sans que ceux-ci aient été créés dans le but d'être liés. Par exemple, si une liaison est faite entre un fichier CSV contenant des noms de présidents et un fichier CSV contenant un ensemble de lieux, il va être possible de faire correspondre certains lieux à ces présidents (pays d'origine, pays qu'ils dirigent, ...).

Ce travail va donc faire le lien entre les principes du STI, les outils et les méthodes existantes avec un travail pratique qui permet l'interprétation et la comparaison de schémas entre tableaux de manière efficiente.

1.3 Questions de recherche

La question de recherche est la suivante :

« Comment automatiser efficacement l'interprétation sémantique tabulaire afin de pouvoir lier des jeux de données entre eux ? »

Afin de répondre au mieux à notre question de recherche, nous avons décidé de découper celle-ci en plusieurs sous questions de recherche.

La première sous question s'oriente vers le STI. Le but est de comprendre de quoi il s'agit, ainsi que son utilité dans ce travail.

La seconde sous-question étudie les différentes méthodes existantes afin d'en extraire les informations essentielles, les limites actuelles et les améliorations possibles.

La dernière sous question s'oriente vers les outils STI. Le but est de savoir s'il en existe et si oui les expliquer brièvement en ne reprenant que ceux qui peuvent potentiellement nous intéresser dans notre recherche.

QR1 : « *Comment fonctionne le principe de STI dans la comparaison de schémas open data ?* »

QR2 : « *Quelles méthodes d'interprétation sémantique des tables et d'inférence de schémas peuvent être appliquées pour permettre une intégration dans le Web des LOD ?* »

QR3 : « *Quels sont les outils existants respectant le principe STI pertinents dans le cadre de ce travail ?* »

2 Chapitre 2 : Etat de l'art

2.1 Introduction

Ce chapitre présente la méthode utilisée pour répondre aux questions de recherche qui ont été définies dans le chapitre d'introduction de ce mémoire [Chapitre 1 : Introduction]. Les réponses à ces questions sont présentées dans les chapitres suivants, qui vont du plus haut niveau au plus bas. C'est-à-dire que la première question va poser les bases et expliquer de manière assez générale ce qu'est le STI [Chapitre 3]. La deuxième question est déjà plus spécifique puisqu'elle se base sur l'étude de différentes méthodes qui permettent de répondre aux principes STI [Chapitre 4]. Et pour finir, la troisième question qui est la plus spécifique puisque celle-ci reprend l'étude des outils existants [Chapitre 4.13]. Au niveau de la deuxième et troisième question, des tableaux de comparaisons ont été créés pour qu'il soit possible d'analyser et différencier facilement les méthodes et les outils. Les chapitres suivants présenteront respectivement les résultats issus de cet état de l'art [Chapitre 7] ainsi que les limitations de celui-ci [Chapitre 8].

2.2 Méthodes

2.2.1 Introduction

Cette revue systématique se base sur les règles de la « *Systematic Review* » (SLR) [11].

Afin de pouvoir présenter ce mémoire nous avons suivi ce plan :

1. Motiver le besoin de réaliser un état de l'art.
2. Définir notre question de recherche. Nous avons choisi une découpe en plusieurs sous-questions afin de pouvoir y répondre au mieux.
3. Définir notre protocole à suivre, basé sur les règles de la SLR, il est décrit tout au long de ce chapitre.
4. Rechercher la littérature, grâce à un ensemble de mots-clés.
5. Sélectionner des articles pertinents.
6. Lire, sélectionner et extraire des informations importantes. Nous avons défini différents critères de sélection et ensuite avons procédé au résumé des articles pertinents.
7. Evaluer et valider ces informations vis-à-vis de nos questions de recherche.
8. Synthétiser les informations.
9. Rédaction de l'état de l'art.
10. Rédaction d'une méthode permettant de répondre à la question de recherche.

2.2.2 Recherche

Nous avons effectué nos recherches sur les moteurs de recherche suivants :

- Google Scholar
- Researchgate.net
- Semantic Scholar

Ces moteurs de recherche nous ont permis de récolter l'ensemble de nos articles. Pour sélectionner ces articles nous avons lu leur titre puis leur abstract pour savoir si ceux-ci pouvaient nous intéresser. Les articles qui ont été cités plusieurs fois dans les articles que nous avons pu lire sont aussi repris dans notre sélection.

Les articles que nous avons le plus utilisés sont ceux que nous avons trouvé sur Google Scholar. Concernant Google Scholar, c'est un moteur de recherche gratuit et facile d'utilisation. Son principe est le même que le moteur de recherche de Google. Ensuite, une recherche plus approfondie a été faite via Researchgate.net, ce dernier est aussi un outil gratuit et facile d'utilisation. Comme dernier outil, nous avons utilisé Semantic Scholar qui est un moteur de recherche universitaire recommandé par l'UCL. Nous avons également récupéré des articles qui se trouvent dans les bibliographies des différents articles que nous avons utilisés. Nous avons également enrichi notre ensemble d'articles avec ceux que nous avons reçu de deux chercheurs en sciences informatiques au sein de l'Université de Namur ayant travaillé sur l'Open Data, Maxime Gobert et Rabeb Abida.

Ces articles expliquent principalement l'utilisation d'outils de méthodes liées au STI pour faire correspondre les valeurs de tableaux (HTML, CSV, …) avec l'open data.

Lors de nos recherches, nous avons pu trouver un certain nombre d'articles via les mots clés suivants : « Schema Inference Open Data », « Tables Matching », « Linked Open Data », « Knowledge base », « Semantic Web », « Knowledge Graph », « Tables Matching », « Web Table », « Table Annotation », « STI Tool ». D'autres mots clés ont été utilisés pour réduire le nombre d'articles trouvés par notre première recherche. Au fur et à mesure notre liste de mots clés s'est adaptée pour répondre aux différentes sous-questions. Ce qui nous a permis d'arriver à une sélection de 57 articles.

Nos recherches ne se sont pas arrêtées aux titres ou aux abstracts, nous avons aussi fait des recherches dans les textes des articles. Nous avons décidé de procéder de cette façon puisqu'il est possible que certains mots-clés n'apparaissent pas dans les titres ou abstract de l'article. Nos mots-clés finaux sont les suivants : « Semantic Table Interpretation », « STI tool », « Dataset Matching », « HTML Table », « Open Data » et « Knowledge Graph ».

2.2.3 Critères d'inclusion et d'exclusion

Au niveau de la sélection de nos articles, différents critères d'inclusion et d'exclusion ont été utilisés.

Ceux-ci sont représenté dans le tableau suivant (Figure 1) :

Critères d'inclusion	Critères d'exclusion
I1 - Article qui répond directement à notre question de recherche	E1 - Article datant d'avant 2005
I2 - Article axé sur le contexte	E2 - Article sans Abstract
I3 - Article axé sur des méthodes STI	E3 - Article rédigé dans une autre langue que le français et l'anglais
I4 - Article axé sur des outils STI	

Tableau 1 : Critères d'inclusion et d'exclusion

Recherchant des articles récents, nous avons décidé de ne reprendre que des articles qui ne datent pas d'avant la date de publication des spécifications du RDF 1.0 en 2004 [12] **(E1)**.

Tout article ne contenant pas d'abstract a été exclu car sans ce dernier il nous est impossible de nous faire une idée concrète de la pertinence de l'article **(E2)**. Nous avons décidé de ne pas reprendre des articles qui ne sont pas rédigé en français ou en anglais **(E3)**.

Nous avons inclus tous les articles qui peuvent répondre directement à notre question de recherche **(I1)**. Pour donner du contexte à notre recherche nous avons décidé de lire des articles axés sur l'open data et le STI à haut niveau pour bien comprendre ce que cela représente ainsi que le contexte **(I2)**. Pour répondre à nos sous-questions de recherche nous avons décidé d'inclure des articles liés aux méthodes d'approche STI ainsi que les outils les utilisant **(I3, I4)**.

2.2.4 Sélection des articles

Nous avons cherché les articles scientifiques qui nous intéressait. Nous ne nous sommes pas basés sur des critères particuliers comme par exemple un auteur bien spécifique, nous avons choisi les articles que l'on jugeait les plus pertinents. Pour ce faire, nous avons utilisé nos critères d'exclusion pour ne récupérer que les articles qui pouvait nous intéresser. Ensuite, nous avons écarté certains articles qui ne nous permettait pas de répondre à nos sous questions de recherche. Nous avons finalement retenu 50 articles.

2.2.5 Traitement des articles

Une fois la sélection terminée, nous avons résumé les articles et récolté les informations qui nous intéressaient, c'est-à-dire des informations qui permettent de répondre à notre question de recherche ou à nos sous-questions. Pour résumer, nous avons décidé de reprendre l'abstract et la conclusion et d'expliquer brièvement le corps de l'article.

Les résultats des recherches ont été repris globalement, c'est-à-dire, repris de l'observation faite ainsi que de son interprétation. Les références vers d'autres articles ont également été reprises dans nos résumés, d'une part pour permettre un effet boule de neige dans notre recherche, d'autre part pour nous permettre de relier les articles entre eux. Pour finir, les informations importantes liées à la question de recherche et aux sous questions ont été extraites.

Chaque article n'a été lu que par un de nous deux. Pour éviter de lire le même article que l'autre, l'utilisation de l'outil Trello nous a permis de savoir où en était l'autre dans son résumé et savoir quel article il avait déjà lu. Les résumés des articles étaient accessibles via un OneDrive.

Via ces différents résumés, nous avons pu effectuer une analyse commune avec les informations récoltées par chacun. Chaque résumé contient le titre de l'article, l'abstract, une synthèse des parties importantes de l'article. Cette analyse nous a permis d'établir notre propre jugement et d'en formuler une conclusion.

2.2.6 Comparaison des articles

Les articles lus sont principalement des rapports de recherche. Le Tableau 2 représente un récapitulatif du type d'article rencontré.

Type d'article	Nombre d'articles
Revues ou enquêtes	5
Recherches pratiques sur des méthodes théoriques	20
Articles sur la recherche et la création ou présentation d'un outil pratique	14
Recherches théoriques	11
Total	50

Tableau 2 Récapitulatif des articles rencontrés par type

Chacun de ces articles a ensuite été relié à la sous-question qui leur correspond. Il a fallu ensuite résumer et analyser ces articles de manière critique pour pouvoir les utiliser.

Le Tableau 3 ci-dessous reprend les références de l'ensemble des articles avec les sous-questions qui leurs correspondent.

Sous-question	Références	Nombre d'articles
Contexte	[1], [2], [3], [4], [5], [6], [7], [8], [9], [10]	10
QR1	[13], [14], [15], [16], [17], [18], [19], [20], [21]	9
QR2	[2], [22], [23], [24] , [25] [26], [27], [28], [29], [30], [31], [32], [33], [34], [35], [36], [37], [38], [39], [40], [41]	21
QR3	[42], [43], [44], [45], [46], [47], [48], [49], [50], [51]	10
Total		50

Tableau 3 Récapitulatif des articles rencontrés par sous-question de recherche

2.3 Conclusion

Au niveau de la méthode de recherche, ce travail suit les principes de la revue systématique [11]. Tout d'abord, il a fallu établir nos motivations ainsi que nos questions de recherche. Une fois celles-ci établies, nous avons cherché les articles pouvant y répondre et nous nous sommes basés sur plusieurs critères pour les sélectionner. Une fois fait, nous avons établi le contexte de l'état de l'art pour permettre une compréhension globale des principaux axes de celui-ci.

3 Chapitre 3 : Comment fonctionne le principe de STI dans la comparaison de schémas open data ? (QR1)

3.1 Introduction

Les données se trouvant dans des tableaux HTML ou bien dans des fichiers de données (fichier CSV ou XML par exemple) sont souvent une mine d'or d'information. Il est donc important de pouvoir les exploiter pour pouvoir par exemple étendre les KB inter-domaines comme DBpedia [6] ou encore Wikidata [13].

Mais l'interprétation de ces tableaux peut être très compliquée à réaliser. Par manque de contexte il peut être difficile de savoir à quoi correspondent les valeurs des différentes colonnes d'un tableau. Par exemple, il est possible de recevoir des tableaux qui n'ont pas de noms de colonnes explicites et il est donc difficile de contextualiser les données reçues [14].

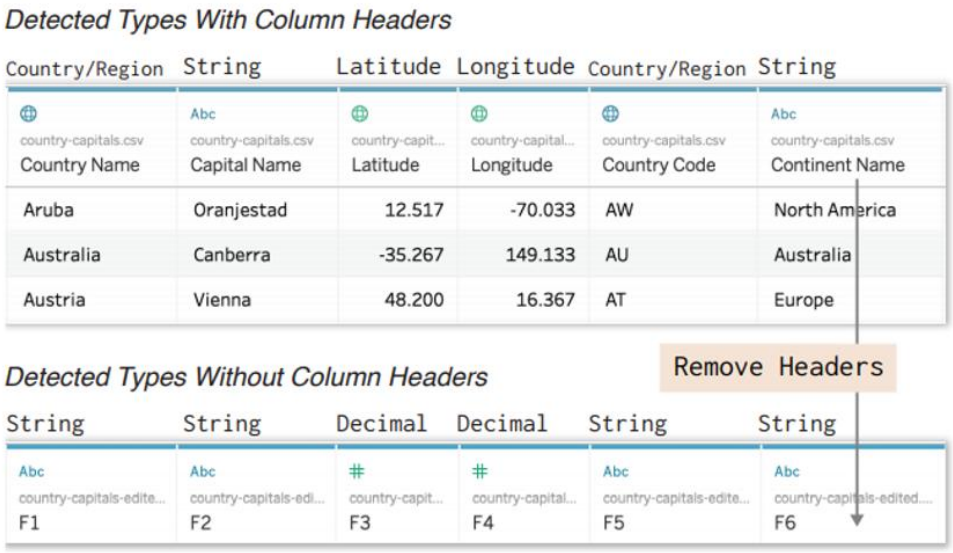


Figure 8 : Types de données détectés pour une table des capitales de pays, avec et sans entête. Image reproduite [14]

Sur la figure 8, chaque tableau est rempli avec le même CSV, dans le premier tableau nous avons les noms de colonnes correspondants aux contextes de chaque colonne (Country Name, Capital Name…). Dans le deuxième tableau, nous n'avons que des noms de colonne correspondant à une lettre et un chiffre. C'est un moyen utilisé par les systèmes SAP pour stocker les informations [14]. SAP est un fournisseur de logiciels ERP, il s'agit d'application commerciales utilisées pour gérer des grandes et moyennes entreprises [52]. Si le deuxième tableau est utilisé, il va être très difficile d'y donner du contexte et de comprendre quel est la valeur réelle des cellules dans notre monde.

Pour répondre à ce problème d'interprétation sémantique, un concept est à chaque fois respecté. Ce concept s'appelle le *Semantic Table Interpretation (STI).*

3.2 Fonctionnement du STI

Ce concept met en œuvre l'utilisation de KG et/ou de KB (par exemple DBpedia). Ce dernier est divisé en trois parties [15]:

- Premièrement, il faut assigner un type sémantique (une classe d'un KG) à une colonne du tableau. Cette étape s'appelle *Column Type Annotation* (CTA).
- Deuxièmement, il faut faire correspondre une cellule du tableau à une entité d'un KG. Cette étape s'appelle *Cell Entity Annotation* (CEA).
- Troisièmement, il faut pouvoir faire correspondre une propriété du KG en une relation entre deux colonnes du tableau. Cette étape s'appelle *Column Predicate Annotation* (CPA).

Par exemple, en utilisant un KB tel que DBpedia et un fichier CSV et tel que :

Header1	Header2
Barack Obama	Hawai
Vladimir Putin	Léningrad
George W Bush	New Haven
Emmanuel Macron	Amiens

Figure 9: Tableau d'un fichier csv d'une liste de président et leur lieu de naissance

L'étape CTA, va définir le Header1 comme une classe DBpedia de type dbo:president. Dans certains cas, les chercheurs vont définir cette colonne comme colonne sujet de la table c'est-à-dire que cette colonne définie le contexte du tableau.

L'étape CEA, va définir Barack Obama comme une entité donc par exemple pour Barack Obama, il faut retrouver le dbr:Barack_Obama. Cette étape peut avoir son lot de difficulté puisque dans certains tableaux il peut y avoir des abréviations, des caractères spéciaux, etc. Il faut donc prévoir un ensemble de règles pour récupérer l'entité correspondante.

L'étape CPA, va faire correspondre la colonne *Header1* et la colonne *Header2*. Dans ce cas, il va falloir trouver un moyen pour faire comprendre que le *Header2* est le lieu de naissance des présidents correspondant au *Header1*.

L'application de ces trois étapes est différente d'un chercheur à l'autre, chacun utilise ses propres techniques pour pouvoir régler ce problème de sémantique. Mais la similitude de chacune des recherches est le respect de ces trois étapes.

3.3 Evaluation (Gold Standards)

Après avoir rempli les étapes ci-dessus, il faut pouvoir être capable d'évaluer la qualité des annotations sémantiques, c'est-à-dire savoir si ces annotations sont proches de la réalité. Les Gold Standards sont ce qui se fait de mieux à l'heure actuelle dans cette tâche. Les Gold Standards sont des systèmes de données qui permettent d'évaluer les performances des méthodes d'annotations. Ils sont constitués d'un ensemble de tables ainsi que leurs correspondances générées manuellement par l'homme dans une base de connaissance [16].

Les Gold Standard les plus utilisés sont [13] :

- T2Dv2
- Limaye
- Musicbrainz
- IMBD
- Taheryan 2015
- SemTab 2020

T2Dv2

Ce Gold standard correspond un jeu de données qui a été annoté de manière manuelle sur un ensemble de 779 Tables venant du Web Table Corpora. Le Web Table Corpora stocke un ensemble de 1.78 billions de pages HTML contenant elles-mêmes 233 millions de tables HTML [17].

Dans T2Dv2, il y a 237 tables qui correspondent au moins à une instance DBpedia. T2Dv2 va donc permettre de mesurer le taux de comparaison possible entre des tables et les KB [16]. Un ensemble de sujets sont repris comme des endroits, des personnes, ... En tout, 237 classes, 25 119 instances et 618 propriétés composent ce Gold Standard [21], [17].

Limaye

Ce jeu de données contient plus de 6000 tables venant de Wikipédia et du Web [13]. Chaque entité est annotée par un lien amenant à un article Wikipédia, les colonnes et leurs relations sont représentées via les concepts de YAGO.

Limaye possède un sous-ensemble appelé Limaye200 qui est composé de 200 tables qui ont été créées à partir de processus manuels et automatiques.

LimayeAll est une autre version de Limaye qui a été créée à partir d'un processus totalement automatisé. Cette version utilise Freebase [8] comme KG et contient 6310 tables [13].

MusicBrainz

Ce jeu de données est composé de tables annotées qui ont été reprises des pages Web de MusicBrainz. Le KG utilisé est Freebase. Chaque page contient une table avec une liste de musique par société de musique [13].

IMBD

Ce jeu de données est orienté sur les films. Il reprend un total de 7416 tables provenant des pages IMDB [13].

Taheryan

Taheryan 2016 est composé de deux jeux de données et qui ont été manuellement annotés. Ces jeux de données ont été créés par Teheriyan et al. [18]. Le premier jeu contient 29 tables concernant les travaux dans les musées et dans le deuxième, 15 tables concernant des armes.

SemTab

Ici, l'utilisation d'un *Framework* est utilisée pour faire une évaluation des données se trouvant dans des tableaux et les KG. Ce *Framework* possède plusieurs étapes d'évaluation et utilise des méthodes automatisées pour créer des jeux de données de référence [15]. En 2019, le KG cible était DBpedia, en 2020 c'était Wikidata. Voici un tableau récapitulatif des différents Gold Standards cités précédemment.

Ce sont donc des ensembles contenant des données sémantiques justes. Elles donnent une indication pour savoir la technique STI appliquée par le chercheur est correcte ou non.

GS		Tables	Colonnes	Lignes	Classes	Entités	Prédicats	KG
T2Dv2		234	1157	27996	39	-	154	DBpedia
Limaye		6522	-	-	747	142 737	90	Wikipedia and Yago
LimayeAll		6310	28547	135 978	-	227 046	-	Freebase
Limaye200		200	903	4144	615	-	361	Freebase
MusicBrainz		1406	9842	-	9842	93266	7030	Freebase
IMDB		7416	7416	-	7416	92321	-	Freebase
Taheriyan		29	2467	16 006	-	-	-	CIDOC-CRM EDM Model Schema.org
SemTab 2019	Round 1	64	320	9088	120	8418	116	DBpedia
	Round 2	11 924	59 620	298 100	14 780	463 796	6762	
	Round 3	2161	10 805	153 431	5752	406 827	7575	
	Round 4	817	3268	51 471	1732	107 352	2747	
SemTab 2020	Round 1	34 295	+-250353	+- 168 045	/	/	/	Wikidata
	Round 2	12,173	+- 83993	+- 55995	/	/	/	
	Round 3	62,614	+- 394468	+- 225410	/	/	/	
	Round 4	22,207	+- 466347	+- 77724	/	/	/	

Tableau 4 Comparaison des Gold Standard et SemTab, Tableau [13] enrichi à partir de [48]

3.4 Nouvelles technologies

L'approche STI met en œuvre des méthodes/techniques récentes qu'il est nécessaire d'expliquer pour mieux comprendre vers quoi cette approche pourrait évoluer à l'avenir, il s'agit des méthodes supervisées et des bases de connaissances imbriquées.

3.4.1 Méthodes supervisées

L'apprentissage automatique, *machine learning* en anglais, est une technique pouvant être mise en œuvre dans la méthode d'interprétation STI. On classe d'ailleurs dans le chapitre suivant ces méthodes en tant que méthodes qui exploitent des données d'entrainement pour l'annotation ou non, on parle de méthode supervisée/non-supervisée (Tableau 13).

3.4.2 Knowledge Graph Embeddings

Un graphe de connaissance imbriqué, *Knowledge Graph embedding* en anglais (KGE), fournit des techniques de représentation des connaissances qui peuvent être utilisées par des applications comme la complétion d'un KG en prédisant des informations manquantes.

On peut voir le KGE comme la solution pour incorporer les connaissances d'un KG dans une application du monde réel [19].

La motivation derrière les KGE est la préservation d'une information structurelle, par exemple la relation entre entités, et la représenter dans un espace vectoriel pour rendre sa manipulation plus facile [20]. Cette approche permet de transformer un KG en un espace vectoriel en conservant l'ensemble de ses informations, il existe deux groupes de méthodes : les modèles de translation et les modèles de correspondance sémantique [19].

- Le modèle de translation utilise des mesures basées sur la distance pour générer un score de similarité pour une paire entité-relation.
- Le modèle sémantique utilise des mesures basées sur la similarité entre objets.

Ils peuvent être particulièrement utiles pour Prédire des liens manquants, par exemple pour le triplet (Bruxelles, estCapitaleDe, ?) ou (?, estCapitaleDe, Belgique). Le KGE va déduire l'information manquante à partir du KG lui-même et ainsi l'enrichir.

3.5 Conclusion

Le *Semantic Table Interpretation* est un concept permettant l'interprétation de tableaux (HTML, CSV,etc.) qui met en œuvre l'utilisation de bases de connaissances, il est divisé en trois étapes : *Column Type Annotation* (CTA), *Cell Entity Annotation* (CEA) et *Column Predicate Annotation* (CPA). Il permet l'annotation des tableaux donnant une interprétation sémantique de ceux-ci. La qualité de ces annotations peut être évaluée au moyen des Gold Standard, il s'agit de systèmes de données permettant d'évaluer les performances des méthodes d'annotations. Les technologies récentes tel que le machine learning et les KGE ouvrent à une amélioration importante des approches STI.

4 Chapitre 4 : Quelles méthodes d'interprétation sémantique des tables et d'inférence de schémas peuvent être appliquées pour permettre une intégration dans le Web des LOD ? (QR2)

4.1 Introduction

Etant donné que les tables contiennent du texte et que nous comprenons la signification d'une phrase grâce à l'interprétation des mots et du contexte qui est fourni autour de celle-ci, nous pourrions interpréter les tables en utilisant les techniques du langage naturel.

Même si le Web offre une large variété de tables de qualité, elles sont généralement intégrées dans des composants HTML et leur description n'est disponible que dans le texte les entourant, surtout que les noms de colonnes ne sont pas toujours présents. Sans connaitre la sémantique des tables, il est donc très difficile de tirer parti du contenu. Typiquement les moteurs de recherche ne se basent que sur l'utilisation du texte en lui-même et non sur les relations possibles entre eux (un tableau ne sera lu que case par case) [27].

Tout d'abord, de manière naturelle, pour interpréter une table nous regardons le haut et la gauche de la table et les labels fournis pour chaque colonne ou/et pour chaque ligne, ensuite, nous recherchons le lien qu'il existe entre chacune des colonnes. Grâce au contexte, Il est aussi possible de comprendre ce que l'on peut trouver dans la table, par exemple grâce à une légende ou un texte additionnel. C'est également grâce à cette sémantique qu'il est possible d'effectuer des opérations entre les tables comme des jointures ou des unions. Quelles sont les méthodes existantes qui peuvent être utilisées pour automatiser cette interprétation ?

4.2 Base des méthodes

L'ensemble des méthodes qui seront présentées utilisent le même principe d'identification des colonnes, des cellules et des relations pour travailler avec l'annotation.

D'une manière générale, la décomposition des données d'un tableau se fait en plusieurs tâches correspondantes au principe fondamental du STI et à ses différentes étapes (CTA, CEA, CPA).

- Préparation des données (enlever des caractères spéciaux, tout mettre en minuscule, etc.).
- Assigner un label de colonne et/ou de ligne à partir d'une ontologie appropriée. Reconnaitre le concept sémantique qui décrit le mieux les données (CTA).

- Lier les valeurs de cellules à des entités (ou parfois à des littéraux) (CEA).
- Découvrir les relations sémantiques entre les colonnes de la table et ajouter les propriétés pour les représenter (CPA).
- Générer une représentation des données liées en levant l'ambiguïté du contenu des cellules en les liants aux entités existantes dans une base de connaissance.

La sortie d'une STI est sémantiquement annotée donnée tabulaire [22].

En d'autres termes, le flux de travail typique qu'on retrouvera dans ces méthodes et le suivant :

1- Retrouver les candidats en liant les composants de la table comme par exemple le label de colonne ou de cellule.
2- Construire un ensemble candidat ainsi que leurs caractéristiques et le model d'interdépendance sémantique entre les candidats et les composants de la table, c'est-à-dire les relations.
3- Sélectionner le meilleur candidat et appliquer la liaison.

Une préparation des données peut être nécessaire afin d'avoir une table régulière qui sera utilisée par le flux STI, c'est-à-dire une table avec une en-tête et dont le nombre de cellules est égal au produit du nombre de colonnes et de lignes [2], [23].

Colonne

Cellule

Nom	**Réalisateur**	**Sortie**	**Origine**
Star Wars, épisode I : La Menace fantôme	George Lucas	1999	Etats-Unis
La Soupe aux choux	Jean Girault	1981	France
Indiana Jones et le temple maudit	Steven Spielberg	1984	Etats-Unis

En-tête H

Lignes r_i

Tableau 5 Exemple d'un tableau régulier

4.3 Méthode probabiliste - Wang et al.

Wang et al. [24] identifient la clé de compréhension d'une table en deux parties. D'abord quel est le concept le plus probable qui contient un ensemble d'entités données et ensuite quel est le concept le plus probable qui a un ensemble d'attributs donnés. C'est-à-dire identifier l'entité d'une colonne de la table et ensuite les valeurs correspondantes à celle-ci. Ils appliquent cette méthode sur des tables HTML qui ont été filtrées au préalable pour ne garder que les tables potentiellement intéressantes, seules des tables ayant une entité unique seront sélectionnées pour cette étude [24].

4.3.1 Méthode d'interprétation

Probase

Pour appliquer cette méthode, ils vont utiliser Probase [25] qui est une base de connaissance dont la base a été construire avec les Hearst Patterns [36] dont l'idée principale est d'exploiter des patterns lexico-syntaxique pour détecter les relations is-a dans un texte [36]. Une construction isA est la représentation conceptuelle de généralisation/spécialisation utilisée pour simplifier la présentation [26].

Mais comme ceux-ci ne sont pas suffisants, Wang et al. [24] vont enrichir cette base de connaissance au moyen d'un nouveau pattern « Quel est le A de I », qui permettra d'ajouter plus d'attributs sur des entités, par exemple, « quelle est la capitale de la Belgique ? » Ici la capitale va être un attribut candidat pour l'entité Pays.

La base de données Probase est construite de la même façon que dans la méthode Venetis et al. [27] expliquée plus loin dans ce document, elle contient un index qui supporte la recherche et le classement des concepts candidats et renvoie des listes d'instances et d'attributs possibles [24].

Les deux scores essentiels fournis par Probase sont la plausibilité et l'ambiguïté. L'exemple de Microsoft et Apple comme nom d'entreprise est très parlant. Apple est le nom d'une entreprise mais signifie également une pomme en anglais, dans le cas d'un tableau d'entreprise, Apple aura un score d'ambiguïté bas et de plausibilité élevé [24].

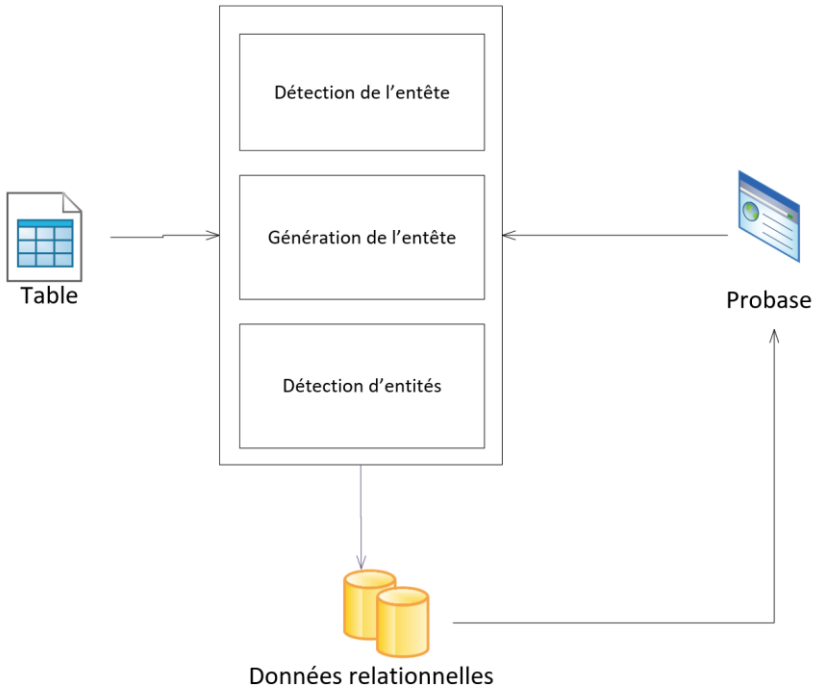


Figure 10 Schéma des étapes de la méthode de compréhension d'un tableau par Wang et al. [24]

Algorithme

La méthode est simple. D'abord ils cherchent les entités de la table en appliquant une fonction fournie par l'api de Probase.

> $\kappa_A(A)$: pour un ensemble d'attributs A, $\kappa_A(A)$ retourne une liste de triplets (c_i, A_i, sa_i) classés par ordre de score sa_i, où c_i est probablement un concept de A, $A_i \subseteq A$ sont des attributs du concept c_i, et sa_i est le score de probabilité donné au concept c pour A [24].

Soit le tableau suivant :

Nom	**Date de naissance**	**Parti politique**	**Taille**
Barack Obama	04/08/1931	Démocratique	1.87
Arnold Schwarzenegger	30/07/1947	Républicain	1.88
Hillary Clinton	26/10/1947	Démocratique	1.68

Tableau 6 Tableau d'exemple de politiciens américains

Nous avons A = {Nom, Date de naissance, Parti politique, Taille}.

On applique $\varkappa_A$ sur A :

$\varkappa_A(A)$ = (Présidents américain, {Date de naissance, Parti politique}, 0.90)
(politiciens, { Date de naissance, Parti politique}, 0.88)
(Joueurs de NBA, { Date de naissance, Taille}, 0.65)

Dans le cas où il n'y a pas de header, alors on utilisera la fonction suivante :

> $\varkappa_E(E)$: pour un ensemble d'entités E, $\varkappa_E(E)$ retourne une liste de triplets (c_i, E_i, se_i) classés par ordre de score se_i, où c_i est probablement un concept de E, $E_i \subseteq E$ sont des entités du concept c_i, et se_i est le score de probabilité donné a concept c pour E [24].

Soit pour notre exemple, soit E = {Nom, Barack Obama, Arnold Schwarzenegger, Hillary Clinton}

$\varkappa_E(E)$ = (politiciens, {Barack Obama, Arnold Schwarzenegger, Hillary Clinton}, 0.95})
(acteurs, {Arnold Schwarzenegger}, 0.5})

Dans le premier cas, on voit que la lecture par ligne n'est pas forcément fiable car le concept des présidents américains est noté comme le plus probable, tandis que dans la seconde fonction il s'agit du bon concept, celui de politiciens, qui est remonté.

Il s'agit donc ici de la détection de l'entête et dans le cas où elle n'existe pas, de la création de celle-ci. Ce n'est qu'ensuite que vient la notion du détecteur d'entités pour les lignes du tableau. Celui-ci fonctionne grâce aux scores générés en appliquant les deux fonctions précédentes en même temps pour obtenir les différents scores candidats et ne garder que les meilleurs candidats possibles pour chaque entité et ainsi obtenir le schéma final.

Dans notre exemple le schéma final sera : (Politiciens, {Date de naissance, parti politique}, 0.88)

4.3.2 Résultats

On démontre ici un moyen simple de comprendre les entités et les attributs des tableaux HTML afin d'en déterminer leur schéma grâce à d'une base de connaissance probabiliste.

4.4 Modèle d'annotation avec Yago – Limaye et al.

Limaye et al. [23] annotent les tableaux du Web avec des étiquettes pour les colonnes et pour les relations représentées par cette table. L'objectif est de choisir une étiquette unique par colonne ou relation, dans une ontologie ayant comme source Yago [37] [23].

Ensuite, ils proposent un modèle graphique qui permet de modéliser l'étiquetage :

- Des colonnes des tables avec des types.
- Des paires de colonnes avec des relations binaires.
- Des cellules du tableau avec des identifiants d'entités.

Le but est d'abord d'annoter chaque table en suivant le chemin suivant :

- Annoter les colonnes de la table à un ou plusieurs types du catalogue.
- Annoter les paires de colonnes avec une relation binaire du catalogue.
- Annoter les cellules de la table à des ID d'entités du catalogue.

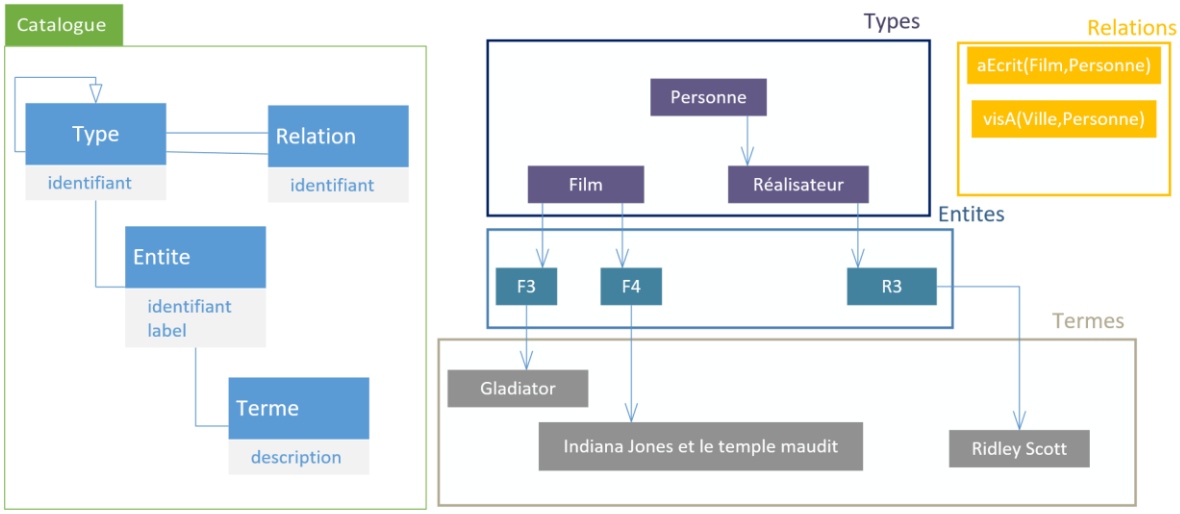


Figure 11 Illustration du modèle d'annotations

4.4.1 Méthode d'interprétation

Pour annoter les tables, Limaye et al. [23] utilisent un catalogue basé sur le catalogue YAGO [37] comprenant de hiérarchie de types avec des relations de sous-types et des entités qui sont des instances de types. Ils ne considèrent l'utilisation que de tables régulières, c'est-à-dire que le nombre de cellules est égal au produit du nombre de colonnes et de lignes.

Titre	**Réalisateur**
Star Wars, épisode I : La Menace fantôme	George Lucas
Gladiator	Ridley Scott
Indiana Jones et le temple maudit	Steven Spielberg

Tableau 7 Films et leurs réalisateurs , exemple repris dans la Figure 11

Dans le Tableau 7 ci-dessus :

- Titre est un Type « Film ».
- Réalisateur est un Type « Personne », qui possède un sous-type « Réalisateur ».
- Le label de la relation Titre-Réalisateur est « aEcrit (Film, Personne) ».
- Le label de l'entité F3 est « Gladiator ».

Chaque entité E peut avoir un ensemble de termes associés. Par exemple, la ville de New York possède les termes « New York, New York City et The Big Apple ».

Les composants des tables sont modélisés comme des nœuds dans le graphe et les interdépendances sont modélisées comme des facteurs. La tâche qui permet de définir l'inférence revient à rechercher une affectation de valeurs aux variables qui maximise la probabilité conjointe il s'agit d'un algorithme approximatif s'appuyant sur des modèles graphiques probabilistes. Un avantage de cette méthode est qu'elle aborde les trois tâches d'annotation simultanément.

4.4.2 Résultats

Ce modèle permet d'exploiter les tables des LOD sur le Web pour répondre à des requêtes relationnelles simples même si les tables sources n'ont pas de schéma uniforme. Ce travail, qui donne un modèle précis pour la tâche d'annotation, ouvre la voie pour augmenter catalogues avec des informations relationnelles dynamiques, cette approche peut conduire à de meilleures réponses aux requêtes relationnelles sur des tables non structurées.

4.5 Modèle d'annotations multiples – Venetis et al.

Comme Limaye et al. [23], Venetis et al. [27] annotent les tableaux du Web avec des étiquettes de colonnes et de relations. Cependant, contrairement à Limaye et al. [23], leur objectif est de choisir plusieurs étiquettes dans une ontologie [23].

De plus, YAGO ne comprenant qu'une petite fraction des étiquettes. Le but sera également de détecter les relations binaires à une plus grande échelle, c'est-à-dire grâce à une extraction issue directement du Web [27].

Venetis et al. [27] proposent une technique pour récupérer automatiquement la sémantique des tables, en ajoutant des annotations à un tableau décrivant les ensembles d'entités de la table. Celles-ci vont également décrire les relations binaires entre les colonnes au moyen d'un modèle reposant sur un calcul de probabilité maximum et l'identification des relations entre la colonne « sujet » et les autres colonnes en utilisant une base de données avec de patterns réguliers lexico-syntaxiques comme les Hearst pattern [36], [27], [28].

4.5.1 Méthode d'interprétation

L'idée clé de ce travail est de d'utiliser l'inférence sur chacun des composants individuels pour améliorer la qualité globale des étiquettes. Pour se faire, ils utilisent deux bases de données, la première qui contient les classes avec ses instances et l'autre qui contient les relations de triplets (argument1, prédicat, argument2), et ils utilisent également un algorithme de recherche de table basé sur la sémantique.

(1) La base de données isA consiste en des pairs (instance, classe) dont chaque paire possède un score. Son but est de produire les labels de colonnes.

(2) La base de données relationnelle comprend des triplets (argu1, predicate1 argu2) et permet d'obtenir le label de la symbolique de la relation comme par exemple « est la capitale de ».

Paramètres du problème

Les tables peuvent être semi-structurées avec très peu de métadonnées comme par exemple des tables sans en-tête ou sans nom. La qualité des tables de l'échantillon varie significativement.

Le but est de créer des annotations pour exposer la sémantique des tables plus explicitement. Pour cela on ajoute deux types d'annotations :

- Les **labels de colonnes** représentent un ensemble d'entités dans une colonne particulière. Par exemple, pour la table de le Tableau 8, on ajoute les annotations « Espèce d'arbre, arbre et plante » à la première colonne et l'annotation « est connu comme » pour décrire la relation représentée dans la table.
- Les **labels de relation** représentent la relation binaire d'une paire de colonne de la table. Par exemple, pour le Tableau 8, on ajoute l'annotation « est connu comme » pour décrire la relation représentée dans la table.

Noisetier	Corylus avellana
Frêne blanc	Fraxinus americana
Hêtre commun	Fagus sylvatica
Saule pleureur	Salix babylonica

Tableau 8 Exemple de table qui associe les arbres à leurs nom scientifique

Identifier la colonne sujet est important dans le contexte car le label associé à celui-ci offre une description précise de quelle table on parle et les relations reflètent les propriétés de la représentation. Cette technique ne demande pas de connaitre le sujet mais permet plus de précision dans les annotations.

L'algorithme de recherche de tables se base sur deux types de propriétés :

(1) Une propriété d'un ensemble d'instances ou d'entités, par exemple les films présentés au Festival de Canne.

(2) Une propriété d'une instance individuelle, par exemple la date d'anniversaire de Barack Obama.

La base de données enregistre les statistiques des cooccurrences et un modèle de lien à probabilité maximum est utilisé pour prédire les meilleurs concepts et relations.

Algorithme

L'algorithme le plus performant utilise une méthode « hybride » pour l'assignation des labels. Elle choisit le label sur base de deux listes générées par deux autres méthodes.

La première (**Model)** utilise la probabilité maximum comme dans la méthode précédente de Limaye et al. [23].

La Deuxième (**Majority)** requière qu'au moins un certain pourcentage des cellules de la colonne possède un label particulier, l'algorithme classe les labels sur base d'un rang en fonction de la majorité.

4.5.2 Résultats

Venetis et al. [27] ont décrit des algorithmes permettant de récupérer partiellement la sémantique de tableaux sur le Web. Étant donné que seule l'étendue du Web correspond à l'étendue des données structurées sur le Web, ils sont en mesure de récupérer la sémantique des données structurées sur le Web de manière efficace. Ils proposent une technique d'extraction d'informations ouvertes afin de récupérer une plus grande quantité de données.

4.6 Message Passing – Mulwad et al.

D'une manière générale, les trois méthodes précédentes ne tentent pas de lier les valeurs des cellules de la table. Wang et al. [24] n'identifient pas non plus les relations entre les colonnes, Venetis et al. [27] identifient uniquement les relations entre la colonne « sujet » et les autres colonnes de la table.

Mulwad et al. [29] décrivent un travail permettant l'inférence automatique des tableaux et de leur représentation en tant que données liées RDF pour les mettre à disposition des utilisateurs.

4.6.1 Méthode d'interprétation

Mulwad et al. [29] mettent également en œuvre un algorithme de passage de messages sémantiques qui utilise les connaissances LOD pour améliorer les schémas existants.

La grande nouvelle fonctionnalité de cette méthode est l'incorporation d'un savoir sémantique, utilisé dans un model graphique, qui sera passé à un algorithme. Dans cette méthode on distingue les constantes littérales comme des nombres et des mesures pour lesquels on ne génère ni annotation, ni entités candidates.

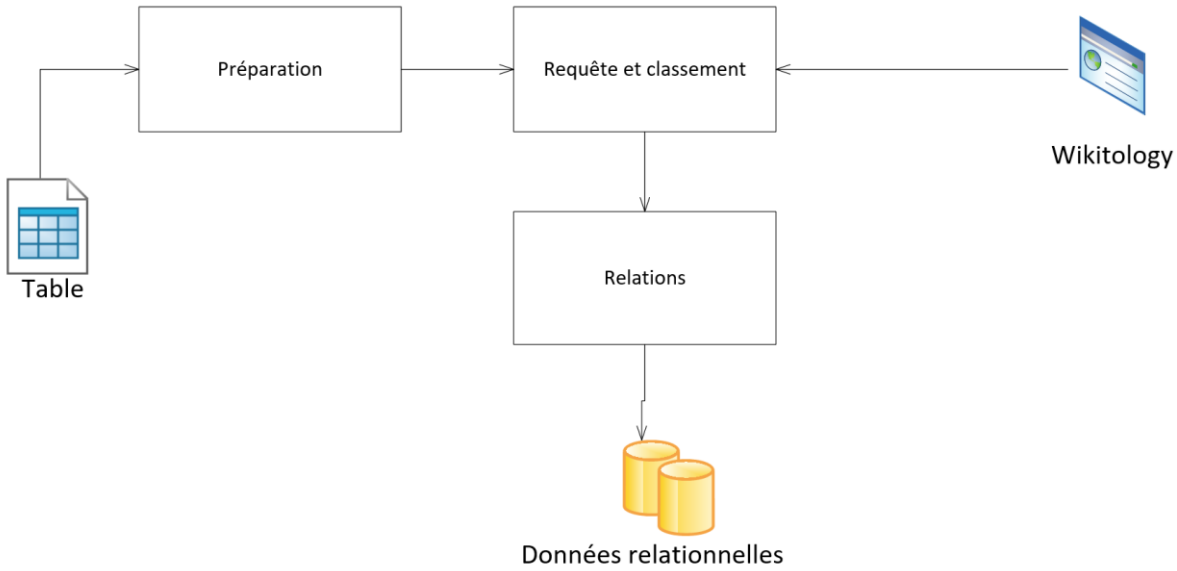


Figure 12 Schéma des étapes de la méthode de compréhension d'un tableau par Mulwad et al. [26]

Préparation

D'abord, la table passe par un processus de préparation des données. Ce processus fait partie de modules indépendants qui permettent de préparer les données pour le flux de travail comme par exemple développer les acronymes ou encore sélectionner un échantillon lorsque la table est trop volumineuse [29].

Requête et classement

Ensuite, on génère les ensembles d'entités candidates pour les cellules, les colonnes et les relations. Pour cela on utilise Wikitology, une base de connaissance combinant les informations de Wikipédia, DBpedia [38] (contient les classes et instances) et Yago (contient les types) [37].

City	**State**	**Mayor**	**Population**
Baltimore	MD	S.Rawlings-Blake	640000
Philadelphia	PA	M.Nutter	1500000
New York	NY	M.Bloomberg	8400000
Boston	MA	T.Menino	610000

Tableau 9 Table des villes américains, tableau reproduit de [29].

Exemple de requête pour le Tableau 9 :

Query String : *Baltimore* et le contexte data *City*, *MD*, *S.C.Ralwings-Blake* et *640 000*.
Wikitology retourne des listes d'entités classées pour *Baltimore*, incluant les entités *Baltimore*, *John_Baltimor*e et *Baltimore_Ravens*.
Un **entity ranker** évalue chaque entité candidate d'une cellule pour montrer la pertinence entre l'entité donnée (*John_baltimore*) et la cellule (*Baltimore*).
Les **relations candidates** obtenues entre la colonne *City* et la colonne *State* pourrait inclure *isPartOf*, *capitalCity*, *bornIn* ,etc.

Relations

Le but est de représenter la table comme un tout en assignant les valeurs des colonnes et des valeurs de cellules et identifier les relations entre les colonnes de la table.

On utilise des modèles de graphes probabilistes pour étudier les probabilités d'interaction entre les variables. Prenons une colonne City qui suggère que les cellules peuvent référer à des instances villes. Cependant dans les autres colonnes on réfère à des joueurs de basket, des entraineurs et des divisions, on peut donc déduire que les villes réfèrent à l'équipe elle-même. Il s'agit d'un exemple de métonymie pour laquelle une entité (l'équipe) est référencée par ses propriétés (localisation de son club). Cette interaction peut être capturée en insérant des arrêtes messagères entre les variables. Il s'agit d'un passage de message sémantique.

C1	C2	C3
R11	R21	R31
R12	R22	R32

Tableau 10 Exemple de table théorique

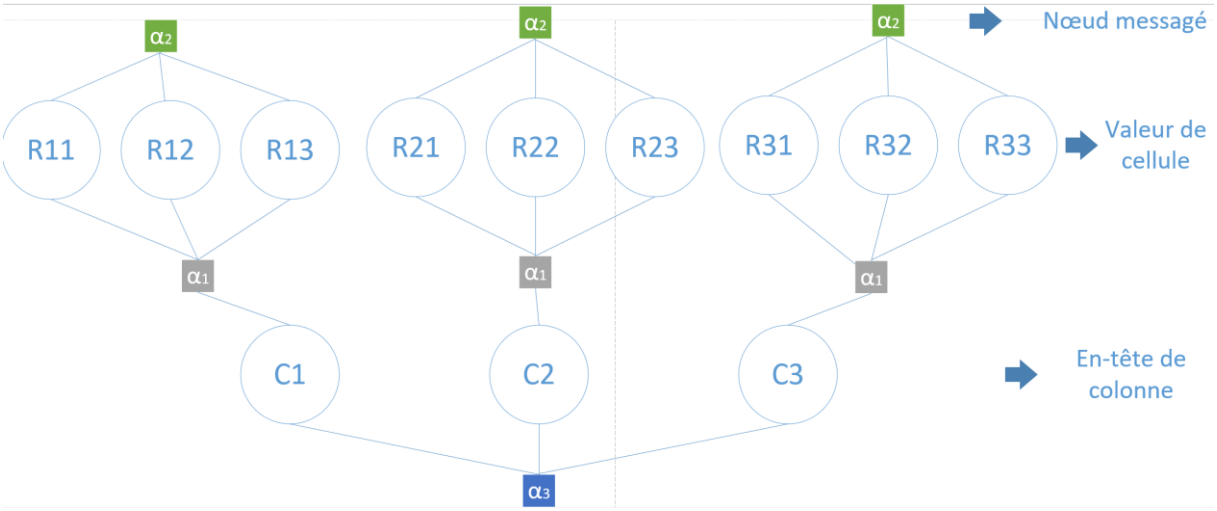


Figure 13 Représentation du Tableau 10 sous forme de graphe factoriel paramétré (image inspirée de [29])

Supposons que dans une itération, $\propto_1$ reçoivent les valeurs City, *Baltimore_Ravens*, Philadelphia et Boston. Le but de $\propto_1$ est de déterminer si toutes les valeurs sont correctes et dans le cas contraire identifier les aberrations. Dans ce cas $\propto_1$ identifie *Baltimore_Ravens* comme une aberration et envoie un message « CHANGE » to R11. Pour le reste c'est un message « NON CHANGE ». Le même principe est appliqué sur $\propto_2$ et $\propto_3$. L'algorithme mettra à jour les différentes valeurs ayant reçues ce message.

Une fois celles-ci mises à jour, elles envoient leurs nouvelles affectations aux nœuds de facteurs et le processus complet se répète. Idéalement, le processus doit être répété jusqu'à ce que les meilleures affectations possibles soient obtenues, c'est-à-dire jusqu'à ce qu'aucun nœud de variable ne reçoive un message de changement.

Cependant, cette convergence est rarement atteinte en pratique, c'est pourquoi dans l'implémentation actuelle, l'algorithme laisse ce cycle se répéter en fonction d'un seuil de changement de variable prédéfinit.

4.6.2 Résultats

Le cœur de cette méthode est donc un modèle graphique probabiliste qui capture un maximum de sémantique, y compris les relations entre les en-têtes de colonnes et entre les entités des lignes de la table. Elle décrit l'utilisation d'un nouvel algorithme de passage de message sémantique qui permet d'éviter des gros calculs de probabilité conjointe comme nous avons pu voir dans les méthodes précédentes [23], [24], [27].

4.7 Table Miner - Zhang

.

Les méthodes existantes ont traité le sujet de manière exhaustive pour la création des espaces candidats [23], [24], [27], [29] et la plupart du temps on retrouve soit des algorithmes qui décrivent des bases de connaissances, soit des algorithmes qui décrivent les composants de la table (header ou contenu de la table) [22].

Il n'est pas nécessaire de lire l'entièreté de la table, la lire partiellement peut augmenter l'efficacité des algorithmes, Limaye et al. [23] ont montré que la première phase peut couter jusqu'à 99% du temps total de calcul.

Cette méthode (TableMiner) a été conçue pour classer les colonnes et lever l'ambiguïté des cellules de manière plus efficace et efficiente vis-à-vis des limitations des œuvres existantes [30].

4.7.1 Méthode d'interprétation

TableMiner adopte une incrémentation en deux phases pour interpréter les colonnes et met en place l'utilisation du contexte existant à l'extérieur des tables [22], [28], [30].

- Les légendes des tableaux et le titre de la page Web peuvent mentionner des termes clés susceptibles de constituer le concept central d'un tableau.
- Une phase d'apprentissage « en avant » qui utilise une liaison incrémentale qui construit une interprétation sur une base itérative, ligne par ligne, jusqu'à ce que l'algorithme soit suffisamment confiant dans la classification des colonnes (cela est déterminé par la convergence).
- Une phase de mise à jour « en arrière » qui utilise les résultats initiaux de la première phrase pour contraindre et guider l'interprétation des données restantes.

Le contexte

Le contexte autour du tableau peut contenir des informations clés sur son contenu. Par exemple, les paragraphes autour de celui-ci peuvent décrire ce qu'il contient. Par ailleurs on retrouve de plus en plus d'annotations sémantiques dans le code HTML (*Rich snippet*) pour permettre d'améliorer l'accès au contenu par les moteurs de recherches [22].

La phase en avant

Soit l'Algorithme 1, nous avons :

C = Canditates, T = Table, i = lignes, j = colonnes, $E_{i,j}$ = set canditate pour la cellule $T_{i,j}$, C_j concepts candidats pour la cellules j

Algorithm 1 Forward learning

1: Input: T_j; $C_j \leftarrow \emptyset$
2: **for all** cell $T_{i,j}$ in T_j **do**
3: $prevC_j \leftarrow C_j$
4: $E_{i,j} \leftarrow$disambiguate($T_{i,j}$)
5: $C_j \leftarrow$updateclass(C_j, $E_{i,j}$)
6: **if** convergence(C_j, $prevC_j$) **then**
7: break
8: **end if**
9: **end for**

Algorithme 1 Phase en avant (image reproduite de [22])

La phase en avant se divise en trois parties : la recherche de candidat, la désambigüisation et la classification. Chaque itération va désambiguïser le contenu d'une cellule en comparant les candidats avec leur contexte. On recherche le texte de la cellule dans une KB et les entités candidates sont ainsi définies (**recherche de candidat**). On va ensuite définir un score de contexte qui va mesurer la similarité entre chaque entité candidate et le contexte de la cellule (**désambigüisation**). L'entité avec le score le plus élevé sera sélectionnée pour la cellule courante et utilisé pour mettre à jour le set Candidate de la colonne C_j. Un score sera également appliqué en fonction du contexte de la colonne (**classification**).

Du point de vue de la convergence de l'algorithme, TableMiner ne traite pas exhaustivement toutes les cellules d'une colonne. Au lieu de cela, il s'arrête automatiquement en détectant la convergence : à la fin de chaque itération, on compare C_j à sa version précédente. Si les scores ont peu changé, on considère la classification comme stable et on choisit le candidat de C_j avec le score le plus élevé pour annoter la colonne correspondante. Dans la majorité des cas TableMiner n'utilise que 50% des données dans les tables de plus de 20 lignes.

La phase en arrière

La phase en arrière est en deux parties, elle va d'abord reprendre les entités candidates C_j définies précédemment comme entités utilisées pour la désambiguïsation des cellules restantes dans la colonne. On va donc parcourir à nouveau les cellules existantes avec cet ensemble d'entités candidates. Du coup, il est possible d'avoir de nouveaux éléments ajoutés à C_j ou des modifications dans les scores existants (**première partie**). Si l'entité gagnante est modifiée dans cette phase alors on réitère la procédure en relançant la première phrase en utilisant le nouveau C_j créé dans la phrase actuelle (**deuxième partie**). Dans la majorité des cas, l'algorithme complet se termine en une seule itération, dans les autres cas on observe une convergence rapide et les itérations suivantes

sélectionnent à nouveau le pool d'entités candidatés déjà traitées dans la phase de mise à jour de la première itération.

Améliorations de la méthode

Pour compléter l'état de l'art de cette méthode, deux améliorations ont été mises en place.
La première, résulte de la résolution du problème de la phrase arrière en enlevant la deuxième partie de celle-ci, la pratique montre que cela améliore le temps de calcul pour un résultat équivalent et la précision obtenue est comparable dans les deux méthodes [28].

La seconde permet d'améliorer la précision en utilisant le contexte intérieur des tables, c'est-à-dire des annotations pouvant se trouver à l'intérieur de la table et aussi le temps de calcul grâce à l'amélioration des algorithmes précédemment cités [22] , [28], cette amélioration est apportée par TableMiner+ [30].

4.7.2 Résultats

Comparé aux méthodes vues précédemment, TableMiner utilise Freebase [8] et obtient les meilleures performances à la fois pour la classification et pour la désambiguïsation. TableMiner peut potentiellement fournir entre 24 et 60% d'économie de calcul par rapport aux par rapport aux méthodes exhaustives, en fonction des tâches [22].

TableMiner contribue à l'état de l'art en introduisant :

1) Une méthode générique d'interprétation de tableaux pouvant être adaptée à n'importe quelle base de connaissances ;
2) Un modèle générique d'utilisation de divers contextes de tableaux dans cette tâche, le premier qui utilise les balises sémantiques des pages Web comme caractéristiques.
3) Une méthode automatique de détermination d'un échantillon de données pour amorcer l'interprétation des tableaux.

L'utilisation de données partielles dans l'interprétation des tableaux détermine un nouveau challenge ; définir un échantillon des données pour chaque tâche et déterminer une taille arbitraire de celui-ci pour qu'il soit suffisamment efficace. Le but ici est de définir les données optimales à utiliser pour l'échantillonnage pour permettre un maximum de précision.

4.8 T2KMatch – Ritze et al.

Actuellement nous n'avons pas relevé le problème des valeurs manquantes dans la recherche d'entités et de relations et la standardisation de performances et de la précision de la comparaison entre les tables et les bases de connaissances. Cette méthode met en évidence ce problème et présente le T2D Gold Standard Evaluation (Gold Standards) pour mesurer et comparer les performances des systèmes de comparaison entre une table et une base de connaissance. Le but est donc de pouvoir remplir des données manquantes dans des bases de connaissance comme DBpedia via des tableaux HTML [31].

Le T2D est composé de deux parties :

The Schema-level gold standard : Il est composé de 1748 tables dont 762 peuvent être comparé avec les classes DBPedia et 7983 colonnes qui correspondent aux propriétés de DBpedia.

The Entity-level Gold Standard : Il contient un sous-ensemble de 233 tables du Schema Level Gold standard dont les lignes peuvent être comparées aux entités DBpedia.

4.8.1 Méthode d'interprétation

La méthode T2K MATCH est un algorithme de comparaison qui se base sur une comparaison d'instance (entités) et de schéma (propriétés). Le modèle de données utilisé est composé de tables décrivant des entités qui contiennent des attributs sous forme de chaine de caractère ou des nombres.

Avant d'utiliser cet algorithme, les tableaux doivent être préparés. On enlève les caractères spéciaux, tout est mis en minuscule, les mesures et les entités sont normalisées, etc. Vient ensuite la détection de l'en-tête du tableau et enfin l'algorithme de comparaison qui se compose de quatre parties.

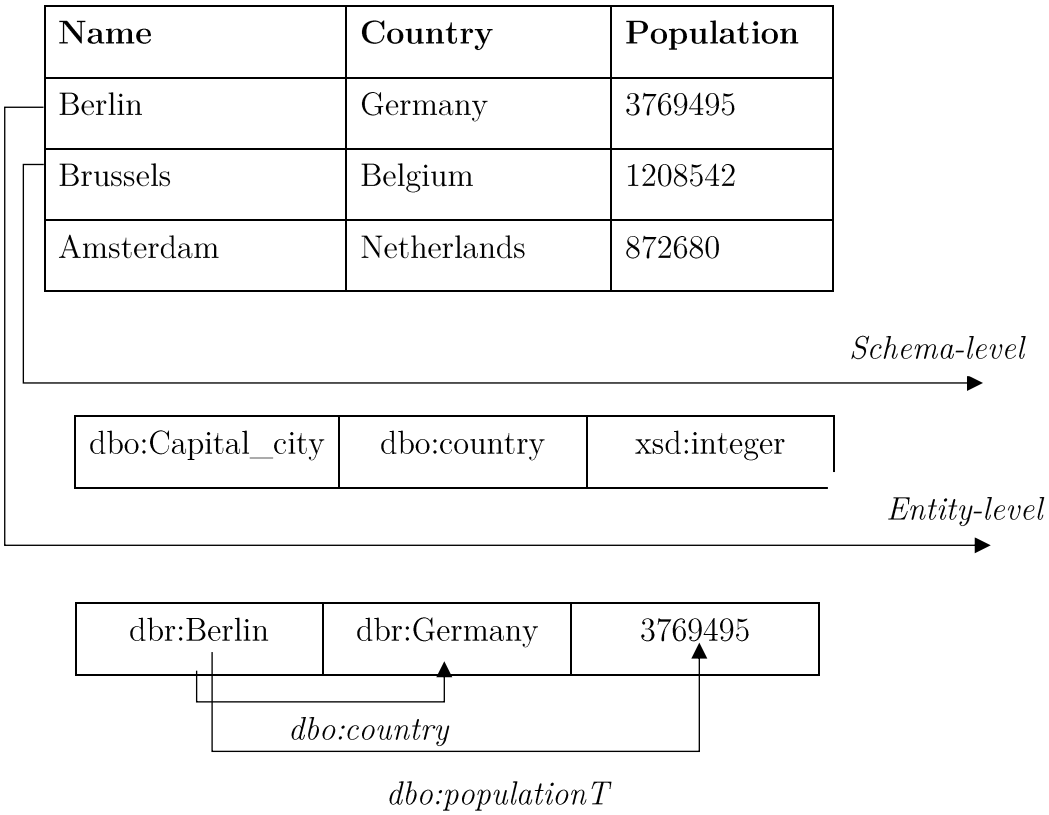


Figure 14 Exemple de table annotée grâce à DBpedia

Sélection des candidats

Le but est de déterminer un ensemble de candidates depuis DBpedia pour chaque entité de la table. On commence par trouver le label DBpedia qui peut correspondre à l'entité. Les candidats trouvés sont classés via un ordre de similarité et seulement le top k est gardé (k est définit comme paramètre de l'algorithme). Ensuite, il faut regarder la distribution des classes par entités et ne reprendre que les classes les plus fréquemment utilisées, les candidats ne participants pas à ces classes sont éliminés.

Comparaison par valeur

Les valeurs de entités candidates et les valeurs des entités de la table sont comparées mais uniquement les valeurs ayant les mêmes types de données. En cas de valeurs multiples, une comparaison avec toutes les combinaisons est faite et la meilleure est choisie.

Comparaison par propriété

On exploite les similarités du schéma en faisant des comparaisons entre les attributs de la table et les propriétés des entités candidates. Il s'agit ici de repérer les relations possibles entre les propriétés DBpedia et les colonnes de la table pour une entité déterminée. On va ainsi pouvoir créer un score de comparaison entre les entités

candidates, on choisira ainsi la classe DBpedia qui contient le score de similarité le plus élevé et on éliminera les entités qui ne lui correspondent pas.

Comparaison itérative

A la première itération, on répète l'étape de comparaison par propriété car la liste de candidats a été changée et donc la matrice de similarité est devenue plus petite. On compare ensuite les scores de similarités entre les comparaisons par valeur et les comparaisons par propriété. Ces similarités sont ensuite agrégées pour chaque entité/candidat et le candidat qui possède le plus haut score est choisi pour son entité.

L'algorithme se termine à partir du moment où il n'y a plus de similarités. Une fois terminé, on choisit la paire qui contient la similarité la plus grande entre les attributs/propriétés et les entités/candidats.

4.8.2 Résultats

Au niveau des résultats on relève un constat intéressant qui est l'évolution des propriétés pouvant amener un manque de précision. En effet, la catégorie "personne" comporte des variations des valeurs des attributs, comme par exemple le poids d'un athlète qui peut changer au cours de sa carrière, ou encore, les dates de naissance ou de décès des personnes historiques sont souvent différentes d'une source à l'autre.

Un autre constat est lié au concept d'attributs d'entité : les tableaux sur les espèces utilisent souvent des noms de ressources incohérents, parfois le nom latin, sinon le nom anglais. Certaines propriétés sont également difficiles à distinguer, c'est-à-dire que les aéroports des États-Unis ont presque toujours le même code IATA et FAA. Si nous rencontrons un tableau qui ne contient que des aéroports des États-Unis, le choix de la bonne propriété est purement aléatoire car les deux auront le même score de similarité.

Un second constat de ce concept découle de la nature de notre cas d'utilisation : si les sources de DBpedia qui correspondent aux entités du tableau HTML n'ont pas de valeurs de propriétés, nous ne pouvons pas calculer de similarité et ne mettrons donc pas en correspondance ces colonnes. Il en va de même dans l'autre sens. Si le seul attribut correspondant d'une table HTML est l'attribut de l'étiquette de l'entité, tout ce que nous pouvons faire est d'appliquer la correspondance des chaînes de caractères et de déterminer la classe la plus fréquente.

De plus, ces tableaux ne sont pas directement utiles pour notre cas d'utilisation car ils ne nous fournissent pas de valeurs manquantes. Cependant, elles pourraient être utilisées pour des tâches de complétion d'ensembles (c'est-à-dire l'ajout de ressources aux classes) ou pour la création de nouvelles propriétés.

4.9 Méthode d'apprentissage – Pham et al.

Pham et al. [33] présentent une nouvelle approche qui se base sur un domaine indépendant afin d'automatiser les annotations sémantique au moyen d'une approche qui utilise l'apprentissage automatique. Cette méthode recherche une solution à deux problèmes, le premier qui concerne qu'un même label peut présenter des données sous différemment formats. Le second est que les données peuvent être très similaires pour des labels différents comme par exemple des températures et des âges de personnes.

Contrairement à Venetis et al. [27] où il faut charger une grande quantité de données qui doivent être en ligne, le modèle d'apprentissage automatique n'est pas spécifique à un domaine particulier il peut donc être utilisé avec n'importe quel domaine. De même que pour Limaye et al. [23] ou Mulwad et al. [29] où l'utilisateur doit fournir des informations supplémentaires au domaine. Pham et al. Mettent en avant un processus automatique et indépendant à la base de connaissance.

4.9.1 Méthode d'interprétation

Métriques de similarité et classificateurs

Pham et al. [33] utilisent les noms de colonnes comme attributs et les utilisent pour comparer les similarités entre eux. A cela, ils vont également utiliser les similarités entre les valeurs et leurs types respectifs et définir un ensemble de métriques de similarité comme la similarité *TF-IDF cosine ou* la similarité *Jaccard* [39]. Ils travaillent à la fois sur tout type de colonne.

C'est à partir de ces métriques de similarités qu'on définira le type sémantique.

Annotation du type sémantique (CTA)

Pham et al. [33] prennent en entrée de leur algorithme un attribut non labélisé et l'ensemble des attributs labélisés du domaine. La sortie correspond à un ensemble de top-k labels sémantiques correspondants à l'attribut non-labélisé. L'algorithme se base sur les différentes métriques de similarités. Cela permet de déterminer un label même s'il n'existe pas de correspondance sémantique directe dans la base de connaissance ciblée.

Algorithme

Soit un ensemble d'attributs {a1, a2, … an}, nous calculons des vecteurs multidimensionnels $f_{i,j}$ ($i \neq j$).

Chaque dimension k correspond à une métrique de similarité, ainsi f(k) représente la similarité entre les attributs a_i et a_j sous la valeur de la métrique k.

Pendant l'entrainement, on labélise chaque vecteur $f_{i,j}$ par True ou False, où True signifie que pour les attributs a_i et a_j on le même type sémantique (même classe d'un KG) et vice-versa.

Pour configurer un nouveau domaine, on stocke l'ensemble d'attributs labélisées {a1, a2, … an} qu'on utilisera pour les comparer avec les nouveaux attributs. Cet algorithme est représenté par la Figure 15.

Par exemple, soit un nouvel attribut a_0, l'algorithme calcule f_{0j} pour tout j ($j \neq 0$) et utilise le classificateur pour labéliser chaque f_{0j} à True ou False. Si le label f_{0j} est True alors c'est qu'il s'agit du même type sémantique que a_j et donc le type sémantique de a_0 [33].

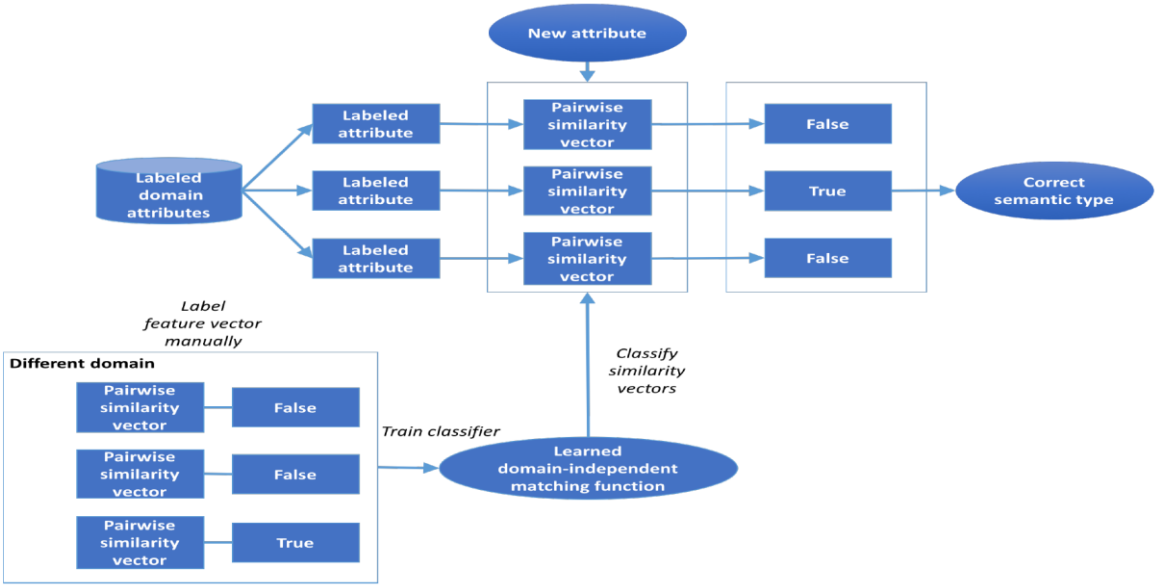


Figure 15 Algorithme du système de labélisation sémantique (image reproduite de [33])

4.9.2 Résultats

Cette approche présente l'utilisation d'un modèle d'apprentissage automatique et permet de l'utiliser dans n'importe quel domaine. Pham et al. [33] permettent alors de pouvoir utiliser les LOD afin d'enrichir ce modèle d'apprentissage et ainsi pouvoir définir un type sémantique jusqu'alors encore inconnu dans la base de connaissance.

4.10 Méthode d'imbrication – Efthymiou, et al.

Efhtymiou, et al. [32] se concentre uniquement sur l'interprétation des lignes de la table en fonction des entités d'un KB, contrairement à Limaye et al. [23] qui traitent les cellules individuellement. Comme Ritze et al. [31], la majorité des tables contiennent une colonne dont les valeurs servent de nom à décrire les entités [31], [32], elle définit le label de la colonne comme la colonne la plus à gauche avec le nombre maximum de valeurs distinctes [32].

Du point de vue de l'extraction des relations, la méthode s'inspire de Venetis et al. [27] (bases de données isA et de relations) et exploite les informations d'un KB ainsi que les relations déjà identifiée.

Elle répond à plusieurs problématiques, comme dans le cas où une cellule peut être la représentation de plusieurs entités, il est nécessaire de comparer cette cellule mais aussi sa description ou d'autres colonnes pour la faire correspondre à la bonne entité. Il en va de même lorsque le nom des entités entre une table et un KB sont différents, par exemple un pays qui change de nom (Bimarnie/Myanmar) [32]. Il est clair que la qualité d'une entité dépendra également du contexte comme le démontre le TableMiner+ [30].

4.10.1 Méthode d'interprétation

Méthode basée sur la recherche (Lookup-based)

La méthode *Lookup-based* exploite les colonnes des tables reconnues comme entités, elle détecte essentiellement les entités candidates en utilisant les informations minimales dans les tables raffinées (basées sur les termes fréquents dans les descriptions) ou enrichies (en exploitant les relations avec d'autres entités).

Cette méthode essaie de comparer les informations pauvres des entités des tables aux riches informations offertes par ces entités dans un KB. Elle se compose de plusieurs algorithmes.

Refined Lookup : Pour chaque label de colonne de cellule, on enregistre le top 5 des types les plus fréquents et comme les annotations de chaque ligne. Par exemple pour « Apple », les types acceptables seraient « entreprise » ou « fruit ».

Factbase Lookup : Identifie et exploite les termes fréquents dans la description d'une entité comme les relations entre elles. Cet algorithme construit un mécanisme d'indexation d'un KB avec des ID, des descriptions textuelles et des index générés *FactBase*. *Factbase* offre un service de recherche pour retrouver les entités avec un label spécifique ou des paires attribut/valeur.

En scannant la table on extrait les mots les plus fréquents utilisés dans la valeur rdfs:description. On extrait aussi les relations binaires entre les entités décrites sur la ligne et les entités mentionnées dans la même ligne. On crée ainsi des pairs candidates attribut/valeur pour n'importe quelle cellule de la ligne.

```
   Data: Table T
   Result: Annotated table T'
 1 T' ← T;
 2 allTypes ← ∅ ;                                          /* a multiset of types */
 3 descriptionTokens ← ∅ ;                                 /* a multiset of tokens */
   /* samplePhase                                                                    */
 4 labelColumn ← getLabelColumn(T);
 5 referenceColumns ← getReferenceColumns(T);
 6 for each row i of T do
 7 |   label ← T.i.labelColumn;
 8 |   results ← search(label);
 9 |   if results.size > 0 then
10 |   |   topResult ← results.get(0);
11 |   |   allTypes.addAll(topResult.getTypes());
12 |   |   tokens ← topResult.getDescriptionTokens();
13 |   |   descriptionTokens.addAll(tokens);
14 |   |   if results.size = 1 then
15 |   |   |   annotate(T'.i, topResult);
16 |   |   |   for each column j of referenceColumns do
17 |   |   |   |   v ← T.i.j;
18 |   |   |   |   if topResult.containsFact(a,v) then /* v is the value of a relation a */
19 |   |   |   |   |   candidateRelations.add(j,a);
20 acceptableTypes ← allTypes.get5MostFrequent();
21 descriptionTokens ← descriptionTokens.getMostFrequent();
22 for each column j of referenceColumns do
23 |   relations[j] ← candidateRelations.get(j).getFirst();
   /* annotation phase                                                               */
24 for each row i of T do
25 |   if isAnnotated(T'.i) then continue;
26 |   label ← T.i.labelColumn;
27 |   results ← search_strict(label, acceptableTypes, descriptionTokens);
28 |   if results.size > 0 then
29 |   |   topResult ← results.get(0);
30 |   |   annotate(T'.i, topResult);
31 |   |   continue ;                                      /* go to the next row */
32 |   for each column j in relations do
33 |   |   r ← relations[j];
34 |   |   results ← search_loose(label,r,T.i.j);
35 |   |   if results.size > 0 then
36 |   |   |   topResult ← results.get(0);
37 |   |   |   annotate(T'.i, topResult);
38 |   |   |   break ;                                     /* go to the next row */
```

Algorithme 2 FactBase Lookup (image reproduite de [32])

Méthode d'imbrication sémantique (Entity Embeddings)

La méthode *Entity Embeddings* exploite une représentation vectorielle du contexte de l'entité dans un KB afin de permettre d'identifier les sous-ensembles candidats les plus pertinents.

Cette approche est une approche de désambigüisation pour le texte. Cette technique est une instance de la famille des techniques appelées *global disambiguation techniques* qui considère que les entités qui apparaissent dans les phrases ou les paragraphes ont tendances à se former en des ensembles cohérents discutés dans le texte [32]. Comme l'exemple des villes ou équipe de basket, vu dans la méthode de Mulwad et al. [29], où une table représente des équipes de basket au lieu d'une liste de villes.

Ce méthode se base sur le Framework DoSeR [40] où les similarités sont calculées comme la similarité cosinus entre les représentations vectorielles, ces vecteurs appelés imbriqués sont une représentation de l'espace continu des entités dans le KB cible. Cela capture la structure du voisinage de chaque nœud. Dans DoSeR, les imbrications sont calculées en utilisant word2vec [53], un algorithme imbriqué pour le texte qui est connu pour sa performance et sa scalabilité.

De plus, la méthode offre deux phases de travail :

1. **Phase Offline**

On crée d'abord un formulaire d'index qui lie chaque entité e du KB à un ensemble de noms connus pour l'entité *m(e)*. Cela permet de récolter les propriétés connues dans la base de connaissances comme les noms communs des entités, exemple rfds:label, altLabel, etc.

Ensuite, on utilise word2vec comme suit : en donnant le KB cible, on calcule les entités imbriquées dans un document texte d, c'est-à-dire qu'on effectue une recherche aléatoire dans le voisinage de chaque entité de la KB où à chaque pas aléatoire on ajoute les URI des nœuds visités dans le document d. Il sera utilisé comme entrée de word2vec pour produire les imbrications pour chaque mot (URI de nœuds).

2. **Phase Online**

On utilise les imbrications et le formulaire d'index pour annoter les tables. On considère uniquement les colonnes avec les valeurs textuelles.

Premièrement, on crée un graphe où les sommets sont l'union de toutes les candidates entités *m(e)* obtenues depuis le formulaire d'index. Pour chaque paire de sommets tels

que les sommets ne soient pas candidats pour la même mention, on ajoute ensuite une arrête dirigée (liée v1 et v2).

Deuxièmement, nous créons une affectation pour chaque nœud en appliquant un PageRank pondéré algorithme [40] qui permet de calculer la pertinence de chaque nœud. Ils utilisent 50 itérations pour le PageRank et sélectionne les nœuds avec le score le plus élevé de l'ensemble de candidats pour chaque mention (*entity mention e*) [32].

4.10.2 Résultats

Les résultats utiliseront deux méthodes hydrides. L'une commençant par la méthode d'encastrement sémantique et ensuite la méthode Factbase. La seconde fera l'inverse. Cette méthode démontre une nouvelle référence et une approche hybride qui surpasse les méthodes individuelles jusqu'à 16% en F-score [32].

4.11 Meimei - Takaoka et al.

Takaoka, et al. [34] s'attaquent d'abord à la mise en place du concept de multi labélisation. En effet, dans le langage naturel, il est tout à fait possible d'avoir une colonne qui relie plusieurs concepts, par exemple une colonne « Entreprise » peut être représenté par le label « industrie » ou encore « firme ».

Dans les méthodes précédentes, on voit qu'habituellement on vient lier les cellules à son entité dans la base de connaissance [23], [24], [29], [30], c'est-à-dire les cellules des NE-colonnes mais il est plus couteux de faire ce lien avec les colonnes littérales.

D'ailleurs, Neumaier, et al. [35] ont proposé une méthode qui applique un algorithme de regroupement hiérarchique sur un KG pour construire une « extension » au graphe de connaissances qui va contenir des valeurs sur l'ensemble des valeurs possibles pour chaque concept candidat. Supposons la paire Ville-Température, l'ensemble des valeurs pourraient se situer dans l'intervalle [-80,57]. Cette méthode va analyser et comparer la distribution des valeurs numériques afin de définir une valeur statistique ce après quoi on retournera le top-k des meilleurs concepts candidats par rapport à celle-ci [2], [35].

En conséquence, comme on peut le voir dans le travail de Neumaier, et al. [35] qui cherchent à labéliser les colonnes contenant des valeurs numériques en les liant à un KG, ces méthodes sont souvent coûteuses en temps de calcul [34], [35].

Takaoka, et al. [34] proposent qu'il faut également l'interdépendances entre les colonnes des tables afin d'améliorer les performances prédictives.

4.11.1 Méthode d'interprétation

Leur méthode se base sur la mise en place d'un ensemble de fonctions composées de différents comportements variés.

a. **Colonne-Contenu** pour mesurer les similarités entre les cellules observées et les entités candidates.
b. **Colonne-Colonne** pour mesurer les similarités une entité candidate avec les entités courantes des autres colonnes.
c. **Titre-Colonne** pour mesurer la similarité entre l'en tête de la colonne et l'entité candidate d'une colonne.

Ils utiliseront des paramètres supervisés pour ces fonctions, où les inputs seront des cellules de colonnes et les outputs les entités candidates correspondante à cette colonne. Les deux types de comportements seront lié au type d'extraction, soit pour les colonnes-NE ou pour les colonnes littérales.

Approche supervisée

Etant donné l'utilisation des paramètres supervisés, on parle de l'utilisation d'une approche supervisée, il s'agit d'une tâche d'apprentissage qui permet à la méthode d'en apprendre d'avantage grâce aux exemples qu'on lui fournit [34]. C'est particulièrement intéressant dans le cas où le domaine du KG n'est pas équivalent au domaine de la table comme on a pu le voir avec Pham et al. [33].

Colonnes-NE

Comme dans la méthode de Limaye et al. [23], chaque cellule correspond à une entité candidate, ici il existe plusieurs candidats qu'on va déterminer et ensuite les transformer en vecteur embarqués dans un KG imbriqué. On calculera ensuite des statistiques.

Colonnes-littérales

Les littéraux peuvent être aussi bien des chiffres ou des textes. Ils vont donc utiliser des fonctionnalités textuelles, comme calculer la taille de la chaine de caractère ou l'occurrence d'une lettre, et des fonctions statistiques, comme calculer le minimum ou le maximum de la colonne.

Interdépendance

Comme illustré dans le Tableau 11, on peut voir de manière naturelle que la probabilité d'avoir la paire de colonne Taille/Température sera plus élevée que la paire Taille/Température.

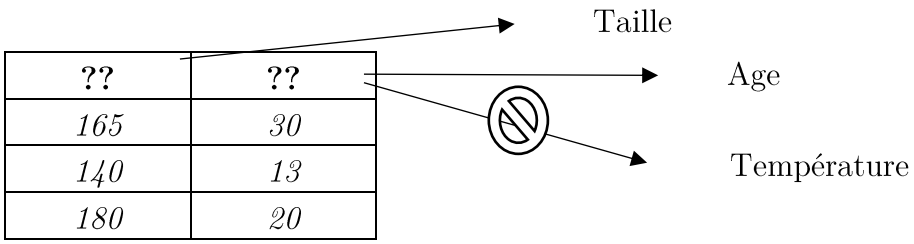


Tableau 11 Annotation sémantique avec interdépendance entre colonne (table inspirée de [34]).

Le Tableau 12 représente respectivement les résultats des annotations pour une méthode conventionnelle (Hybrid), une méthode sans interdépendance (w/o CID) et une méthode avec interdépendance entre les colonnes (w/ CID). La méthode proposée avec l'interdépendance entre colonne possède les meilleures performances pour chaque statistique.

Method	**MAP@5**	**nDCG@5**	**Sim@5**
Hybrid	*0.225*	*0.291*	*0.480*
Proposed (w/o CID)	*0.351*	*0.413*	*0.537*
Proposed (w/ CID)	***0.464***	***0.741***	***0.635***

Tableau 12 Résultats des annotations sur les tables (Table reproduite de [34])

4.11.2 Résultats

Cette méthode se basera sur l'utilisation du KG WordNet [54] qui est une base de données anglaise. Et l'utilisation de métriques de mesure de qualité comme MAP@k ou nDCG@k [41]. Comparé aux méthodes de Limaye et al. [23] ou de Pham et al. [33] , l'annotation des colonnes-NE sera plus précise et comparé à la méthode de Pham et al. [33] , l'annotation des colonnes-littérales le sera également. Cette méthode montrera également de meilleurs performances de calcul. Elle démontre en conséquence trois avantages aux approches existantes : Le support de différents types de données comme les données numériques, la précision et l'efficacité sur le temps de calcul.

4.12 Synthèse des méthodes

Ces méthodes permettent l'annotation et l'interprétation de différents tableaux peu importe leur contexte ou s'ils possèdent une en-tête, elles répondent à différents challenges et apportent des solutions parfois très différentes. Le schéma de comparaison est représenté par le Tableau 13.

Limaye et al. [23] annotent les tableaux avec des étiquettes de colonnes et de relations et se basent sur l'utilisation d'un algorithme probabiliste pour l'assignation des étiquettes. Tout comme Venetis et al. [27], ils utilisent une stratégie qui examine l'ensemble du contenu de la table tout en se basant sur l'application d'un modèle graphique probabiliste qui représente les interdépendances entre les éléments d'un tableau.

Par contre, à la différence que Venetis et al. [27] qui gèrent l'utilisation de plusieurs labels par ontologie, Limaye et al. [23] montrent que la phase de lecture de la table coute environ 80% du temps de calcul. Ce modèle a été déterminé par Mulwad et al. [29] et Zhang [22] comme trop couteux en temps de calcul au niveau du calcul de la distribution de la probabilité, c'est pourquoi ils proposent de nouvelles solutions.

Mulwad et al. [29] utilisent le même modèle graphique mais avec un algorithme de passage de message sémantique qui prend en compte les relations sémantiques entre les colonnes comme le fait Venetis et al. [27], mais en revanche, elle prend également en compte les entités au sein des lignes en plus des en-tête de colonne.

En plus d'également mettre en place cet algorithme de passage de message sémantique, Zhang [22], [28], [30] reprend les étapes telles que l'identification de la colonne Sujet, l'annotation des colonnes littérales et l'analyse d'un échantillon de données extrait de la table source.

Afin de pouvoir évaluer les performances et étudier le problème de remplissage des valeurs manquantes des bases de connaissances grâce à des tableaux HTML, Ritze et al. [31] présentent le premier Gold Standard public pour comparer les tableaux à des bases de connaissance. Les Gold Standards vont d'ailleurs s'intégrer dans les travaux futurs [30], [32].

Du point de vue du flux de travail STI, on y retrouve ses différentes étapes pour chaque méthode.

CTA

L'analyse des colonnes se distingue en trois cas. Le premier où l'on va rechercher le concept d'entité candidate afin d'identifier l'entité représentée par le tableau, ce cas est le concept clé que l'on retrouve dans chaque méthode [22], [23], [24], [27], [28], [29], [30], [31], [32], [34]. Ensuite, le sujet qui n'est pas spécialement représenté comme clé de la table car il peut contenir des valeurs en double comme dans Venetis et al. [27]. Cette colonne sujet est d'ailleurs parfois déterminée et déterminante dans la mise en place des différents algorithmes [22], [27], [28], [30], [32].

Il en est de même pour les littéraux, les liaisons de ces valeurs avec un KB sont d'ailleurs très couteuses en temps de calcul [22], [27], [28], [29], [30], [31], [32], [34]. On peut néanmoins retrouver des méthodes récentes implémentant des algorithmes plus performants [32], [33], [34].

CEA

Le lien permettant de lier une valeur de cellule à son entité candidate est aussi utilisé dans la grande majorité des méthodes [22], [23], [24], [28], [29], [30], [31], [32], [34]. Une cellule pouvant d'ailleurs parfois être représentée par plusieurs entités il est important d'étudier son contexte pour la définir correctement et ainsi améliorer la [22], [28], [30], [32]. On parle de comparaison *entity-level*.

CPA

L'analyse CPA permettant de lier une entité à ses propriétés et aussi utilisée dans la grande majorité des méthodes [22], [23], [27], [28] , [30], [31], [32], [34]. On parle ici de comparaison *schema-level*.

On peut remarquer qu'une partie des méthodes travaille avec un tableau qui a été au préalable préparé, c'est-à-dire que certaines valeurs ou champs sont filtrés, supprimés ou modifiés. Des règles de normalisations sont appliquées sur les données se trouvant dans les tables comme par exemple enlever les espaces additionnels ou les caractères spéciaux, tout mettre en minuscule, etc [24], [29], [31], [32].

Apprentissage automatique et KGE

Les méthodes récentes voient émerger l'utilisation de l'apprentissage automatique ainsi que l'*entity embedding* aux sein de leur implémentation [32], [33], [34]. Elles nécessitent des données d'entrainement ; grâce à cela elles peuvent être plus efficientes que les méthodes traditionnelles cependant la qualité des résultats reste très proche [31], [33] même si cela permet de pouvoir définir des types sémantiques qui jusque-là étaient encore inconnus dans la base de connaissance.

Méthode	Year	KB	Analyse CTA			Cellule-Entité CEA	Colonne-Colonne CPA	Préparation données	Apprentissage automatique
			Entité	Littéral	Sujet				
Limaye et al. [23]	2010	Yago	✓	×	×	✓	✓	×	×
Venetis et al. [27]	2011	isA Database	✓	✓	✓	×	✓	×	×
Wang et al. [24]	2012	Probase enrichie	✓	×	×	✓	×	✓	×
Mulwad et al. [29]	2013	Wikitology	✓	✓	×	✓	✓	✓	×
Zhang [22]	2014	Freebase	✓	✓	✓	✓	✓	×	×
Zhang [28]	2014		✓	✓	✓	✓	✓	×	×
Zhang [30]	2016		✓	✓	✓	✓	✓	×	×
Ritze et al. [31]	2015	DBpedia	✓	✓	×	✓	✓	✓	×
Pham et al. [33]	2016	Domaine indépendant	✓	✓	×	×	✓	×	✓
Efthymiou, et al. [32]	2017	DBpedia	✓	✓	✓	✓	✓	✓	×
Takeoka et al. [34]	2019	Wordnet	✓	✓	×	×	×	×	✓

Tableau 13 Comparaison des méthodes STI

4.13 Conclusion

En conclusion, ces méthodes répondent à différents challenges comme la comparaison entre des entités synonymes et/ou homonymes ou la désambigüisation des colonnes entités et littérales ou encore l'amélioration de la précision ou du temps de calcul.

Nous avons pu voir que Limaye et al. [23] font partie des pionniers en matière d'interprétation sémantique des tables.

Nous avons également pu observer que, dans la majorité des méthodes, les différentes étapes CPA, CTA, CEA sont respectées, même si les algorithmes sont parfois très différents, utilisation de modèles graphiques ou algorithmes itératifs par exemple.

Ritze et al. [31] présentent le premier Gold Standard public pour comparer les tableaux à des bases de connaissance, c'est à partir de là que les Gold Standards vont commencer à émerger dans les autres travaux [30], [32].

La préparation de données n'est pas considérée comme point essentiel dans la recherche même si la normalisation des tableaux peut apporter plus de précision.

Pour apporter une solution au problème des bases de connaissances incomplètes, l'apprentissage automatique et les bases de connaissances imbriquées sont des nouveaux outils qui sont de plus en plus retenus dans la recherche d'approches STI.

5 Chapitre 5 : Quelles méthodes d'interprétation sémantique des tables et d'inférence de schémas peuvent être appliquées pour permettre une intégration dans le Web des LOD ? (QR2)

5.1 Introduction

Il peut être difficile de comprendre et d'exploiter les données open data à leur plein potentiel. Certains outils ont donc été créés pour faciliter la vie des utilisateurs dans le déploiement de l'open data sur le web ou bien l'interprétation sémantique de ces derniers.

Dans cette section, quatre outils d'annotation sémantique vont être décrits et comparés. S'il existe plusieurs versions de ceux-ci, nous reprenons les deux dernières existantes. Pour la sélection de ces quatre outils, nous avons décidé de reprendre les outils qui sont arrivés dans le top 3 du défi SemTab 2020[7] qui a eu lieu en 2020 (MTab, LinkingPark, Dagobah) ainsi que MantisTable qui revient chaque année avec des résultats très corrects. De plus, un outil d'évaluation d'annotation va aussi être présenté. Le but de ce dernier est de permettre d'évaluer les annotations des outils d'annotation sémantique.

MTab est un outil d'annotation sémantique automatique. Lors des défis SemTab 2019/2020, il reçut les meilleurs résultats. Dans cette section, nous parlerons des deux versions de MTab. La version de 2019 se base sur le KG DBpedia et la version de 2020 se base sur le KG Wikidata.

Tout comme MTab, MantisTable est un outil d'annotation sémantique automatique. Il reçut de très bons résultats lors des défis SemTab 2019/2020. Dans cette section, nous parlerons des deux versions de MantisTable. La version de 2019 se base sur le KB DBpedia et la version de 2020 se base sur le KB Wikidata.

Contrairement aux autres outils, LinkingPark n'a participé qu'au SemTab 2020. Il n'est donc pas possible de comparer avec une version antérieure.

Dagobah est un outil d'annotation sémantique automatique. Lors du défi SemTab 2019, il se retrouve dans les derniers du classement, mais pour le défi SemTab 2020, il arrive à se hisser à la troisième place. Dans cette section, nous parlerons des deux versions de Dagobah (2019,2020).

[7] http://www.cs.ox.ac.uk/isg/challenges/sem-tab/2020/results.html

5.2 MTab 2019

L'outil MTab doit avoir rempli certaines préconditions avant de pouvoir mettre en place le principe de STI [42].

Ces préconditions sont les suivantes :

- Précondition 1 : Ils présument que le KG est complet et précis.
- Précondition 2 : Ils présument que le type d'une table est une table relationnelle verticale.
- Précondition 3 : Ils présument qu'il n'y a pas d'informations qui sont partagées entre les tables.
- Précondition 4 : Ils présument que les valeurs des colonnes ont les mêmes entités types et data types.
- Précondition 5 : Ils présument que la première ligne du tableau est la ligne d'en tête et que la première cellule de la colonne est l'en tête de la colonne.

Une fois que ces cinq préconditions sont validées, il est possible de résoudre les principes de CEA, CTA et CPA. Pour ce faire, l'outil sépare l'ensemble de ces principes en sept étapes.

5.2.1 Prétraitement

La première étape est assez classique et se retrouve dans la plupart des outils ou méthodes du STI. Il faut corriger l'ensemble des caractères qui risque de faire rater la correspondance avec les entités du KG. Il faut donc évincer les caractères spéciaux, les nombreux espaces, … Pour régler ce problème, l'équipe de MTab va utiliser l'outil Ftfy Tool[8] qui va permettre de régler toutes les erreurs d'écriture et les modèles Fasttext[9] qui vont permettre de prédire à quelle langue appartient les valeurs se trouvant dans les cellules (anglais, espagnol, français…). Ensuite, le modèle Duckling[10] est utilisé, ce dernier va permettre de faire une prédiction sur le type de donnée se trouvant dans chaque cellule (numérique, URL, email…).

Pour finir, les modèles SpaCy[11] sont utilisés pour prédire quel type d'entité se trouve dans chaque cellule. Des services de recherche tel que DBpedia Lookup, Wikidata Lookup sont utilisés pour retrouver des candidats intéressants correspondants à la cellule [42].

[8] https://pypi.org/project/ftfy/
[9] https://fasttext.cc/
[10] https://github.com/facebook/duckling
[11] https://spacy.io/usage/models

5.2.2 Traitement

La seconde étape permet d'estimer les entités candidates ressorties par les différents services de recherche lors de la première étape. Ici, un score de pertinence (de 0 à 1) est utilisé pour faire ressortir les entités les plus intéressantes.

La troisième étape s'intéresse surtout aux colonnes. Il faut pouvoir différencier les colonnes entités et les colonnes numériques. Un vote majoritaire est fait en utilisant les modèles Spacy et Duckling. Si le vote renvoie un tag *text* ou *entite* alors la colonne est de type entité sinon elle est numérique [42].

Pour les colonnes numériques, MTab va regrouper l'ensemble des valeurs dans une liste. Cette liste va être utilisée dans le Endpoint DBpedia pour récupérer un ensemble de candidat pouvant correspondre au type de la colonne numérique. Par exemple, dans un tableau comprenant des livres, nous avons des dates, l'outil MTab va donc essayer de comprendre à quoi correspond ces dates, et en utilisant le DBpedia Endpoint, il va retrouver une ontologie comme « PeriodicalLiterature » [42].

Pour les colonnes entités, un ensemble de probabilités est agrégé en un score (0 à 1) pour donner des scores au différents candidats de type entité.

La quatrième étape établit les relations entre les différents candidats (les colonnes). Pour cet outil, il y a deux types de relations possibles entité-entité ou entité-non entité.

Pour les relations entité-entité, il suffit de voir si les entités ont des relations communes en utilisant le DBpedia Endpoint (relations ou propriétés).

Pour les relations entité-non entité, une estimation est faite entre les entités candidates et les valeurs des cellules, ce qui donne des entités candidates avec des relations paires.

5.2.3 Post traitement

CEA

L'étape cinq est une ré-estimation des candidats de type entité. Il faut prendre en compte un ensemble de probabilités [42]:

- Les probabilités des entités candidates données par les services de recherche.
- Les probabilités des entités candidates compte tenu des probabilités de leur type.
- Les probabilités entre la valeur de la cellule et l'entité en utilisant l'algorithme de Levenshtein.
- Les probabilités des entités candidates en fonction des valeurs des cellules sur une même ligne.

Avec l'ensemble de ces probabilités, les entités correspondantes à chaque cellule peuvent être trouvés.

CPA

L'étape six correspond à la sélection des plus hautes probabilités des candidats pour établir leur relation via un vote majoritaire.

CTA

L'étape sept correspond à la sélection des plus hautes probabilités des candidats pour établir leur type via le vote majoritaire.

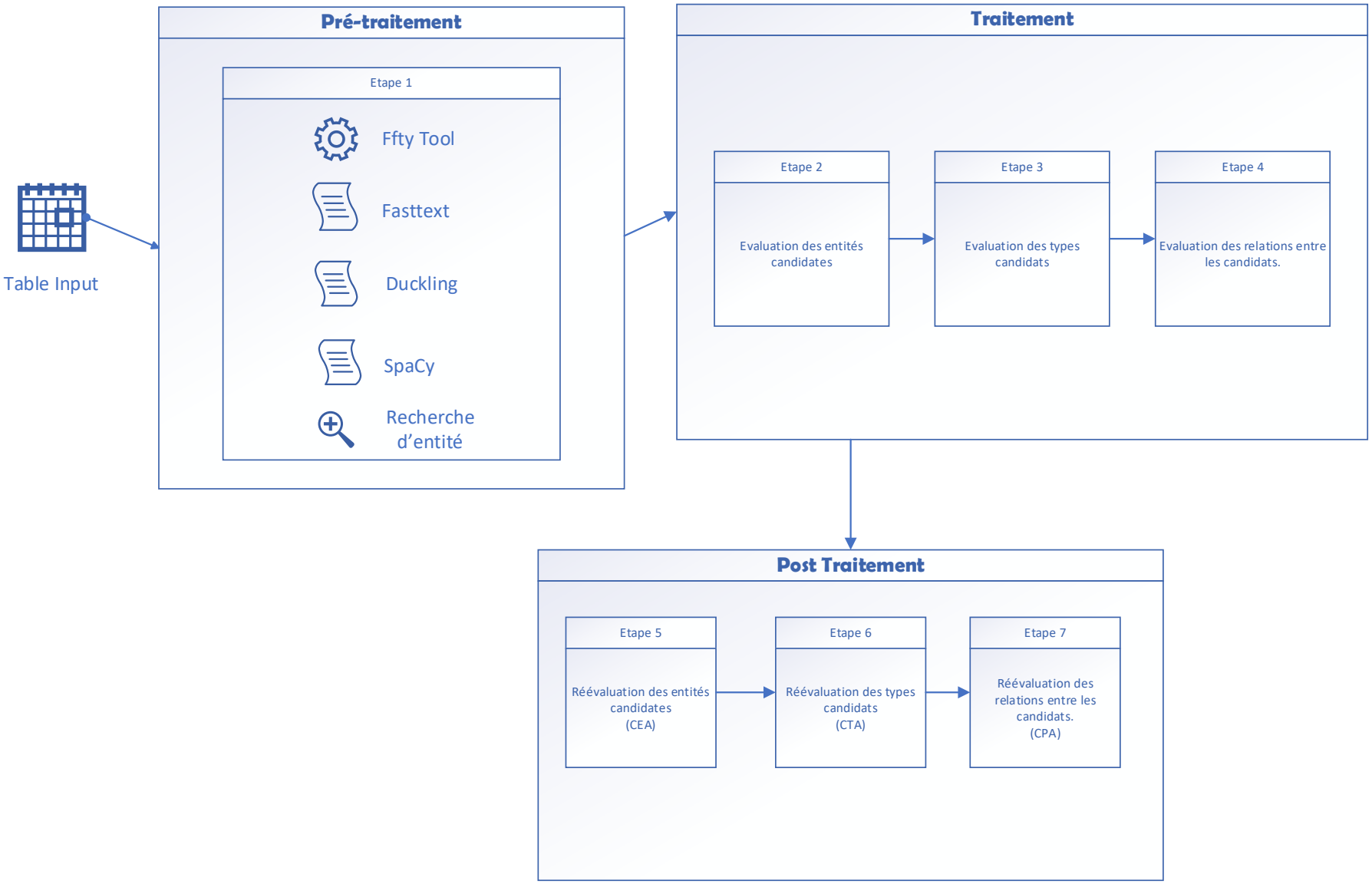


Figure 16 : Fonctionnement Mtab 201

5.3 MTab 2020

L'outil MTab remplit le même rôle que celui de 2019. Ici, le KB utilisé est Wikidata [43].

Par rapport à son homologue de 2019, les préconditions sont presque les mêmes, mais il y a quand même certaines différences. Les préconditions 1,2,4 et 5 du MTab 2019 se retrouvent dans celui de 2020. La précondition 3 ne s'y retrouve plus et est remplacée par celle-ci :

L'outil ne prend pas en compte l'analyse de colonne et suppose que la première colonne du tableau correspond à la colonne-sujet.

Au niveau de l'automatisation des tâches CEA, CTA, CPA, certaines différences sont aussi remarquées :

- Amélioration des performances des correspondances en utilisant les révisions historiques [12]disponibles sur Wikidata.
- La correction des fautes d'orthographe/des bruits se trouvant dans les cellules des tableaux est améliorée.
- Réduction du nombre de candidats pertinent en émettant l'hypothèse qu'une relation logique existe entre les cellules du tableau.

Le nombre d'étapes diffère aussi, le nombre passe de sept étapes à trois.

5.3.1 Prétraitement

Lors de la première étape, ils récupèrent l'ensemble des informations de Wikidata durant une période bien précise. Ensuite, ils récupèrent l'historique des révisions sous forme RDF (sujet – prédicat – objet) ; ce qui permet d'enrichir les déclarations de Wikidata. Ils utilisent une table d'hachage pour récupérer un ensemble d'entités contenant des étiquettes multilingues, des alias, des identificateurs [43]. Pour finir, ils indexent des déclarations d'articles (article – propriété – article).

[12] https://www.wikidata.org/wiki/Wikidata:History_Query_Service

5.3.2 Traitement

La deuxième étape consiste à retrouver les entités candidats des cellules. Pour ce faire, deux types de recherches sont faites.

La recherche par cellule

Pour rechercher l'entité correspondant à la cellule, une utilisation de l'algorithme de Levenshtein (recherche approximative) et d'une liste de score est mise en place. Avec ça, ils peuvent récupérer l'ensemble des entités pertinentes liées à la cellule.

La recherche via deux cellules

Pour enlever toutes ambiguïtés au niveau des cellules des tableaux, une relation est créée entre les cellules d'une même ligne. La première cellule utilisée se retrouve dans la colonne sujet du tableau, la seconde cellule est une cellule qui se situe sur la même ligne que la première cellule. Toutes les correspondances entre ces deux cellules vont former des listes de réponses et il ne sera gardé que les réponses contenant des déclarations valides [43].

On privilégie les réponses apportées par la recherche via deux cellules si cette recherche ne renvoie rien, alors les résultats des recherches par cellule sont pris.

Contrairement au MTab 2019, la correspondance entre les deux cellules se fait avant le post-traitement.

5.3.3 Post-traitement

La troisième étape, la correspondance entre le candidat de la colonne sujet et les autres cellules se situant sur la même ligne est faite. Un calcul est produit pour vérifier la similarité entre les candidats de la colonne-sujet et les candidats des autres colonnes.

CEA

Le candidat ayant la plus grande similarité par rapport à la valeur de la cellule est utilisé comme annotation CEA.

CPA

L'ensemble des propriétés des candidats est agrégé et un vote majoritaire est appliqué pour récupérer les annotations CPA.

CTA

Il faut reprendre les annotations CEA et rassembler l'ensemble de leur type pour ensuite faire un vote majoritaire pour récupérer les annotations CTA.

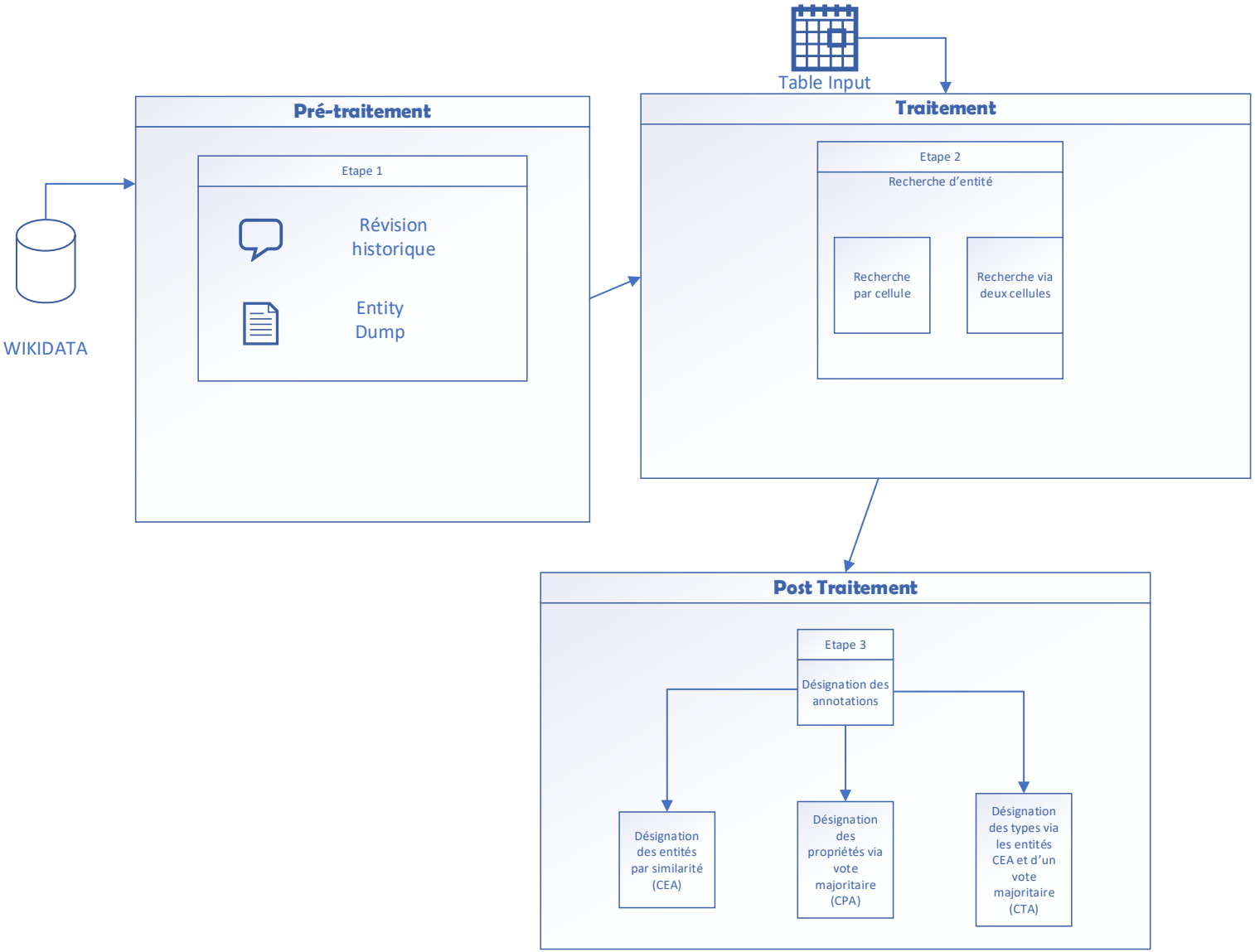


Figure 17 : Fonctionnement Mtab 2020

5.4 MantisTable 2019

L'outil MantisTable ne nécessite pas de remplir de préconditions avant de pouvoir mettre en place les principes de STI [44]. Cependant, deux points sont à relever.

Le premier est le fait que l'équipe a dû développer et adopter une nouvelle manière de charger les fichiers d'entrée, puisqu'à la base, l'outil MantisTable ne lit que les fichiers JSON mais le défi SemTab 2019 n'envoyait que des fichiers CSV.

Le second point est que l'étape *Column Analysis* a été supprimée puisque les colonnes des destinations sont déjà données.

5.4.1 Prétraitement

Cette étape est assez courante dans les méthodes/outils utilisant le principe de STI. Cette étape consiste à la préparation des données c'est-à-dire à la transformation des textes en minuscule, résolution des acronymes/abréviations, unités des mesures, etc. Pour faire ce genre de transformation, l'utilisation d'expressions régulières et Oxford English Dictionary[13] a été appliquée.

5.4.2 Traitement

La deuxième étape (*Column Analysis*) ressemble à peu près à l'étape numéro 3 de l'outil MTtab 2019. Il différencie les types de colonne entre entité ou littéral. Cela se fait via 16 expressions régulières qui permettent de définir des types d'expressions régulières (adresse, URL, code couleur hexadécimal…). Si la colonne dépasse une certaine limite d'occurrence, alors la colonne est définie comme littérale sinon c'est une entité.

Ensuite, il reprend les colonnes entité et cherche la colonne qui pourrait représenter le mieux le sujet de la table (la colonne sujet). Pour ce faire, un calcul reprenant le nombre moyen de mots pour chaque cellule, le nombre de cellules vides dans la colonne, le nombre de cellules avec une valeur unique et une distance à partir de la première colonne entité sont effectués. La colonne, qui a la plus haute valeur après ce calcul, est considérée comme la colonne-sujet.

CEA

La troisième étape (*Entity linking*) est aussi utilisée dans le MTab 2019 mais est faite différemment. Plutôt que faire cellule par cellule, l'équipe fait des liaisons pour l'ensemble des cellules d'une ligne pour essayer d'établir le lien entre les cellules et les entités du KG. S'il y a plus d'une entité de retourner, l'équipe va utiliser la distance de Wagner Fischer et reprendre celle avec la distance d'édition la plus faible. Pour les littéraux, ils

[13] https://www.oed.com/

utilisent un simple algorithme de correspondance qui se base sur différents types de données (dates, nombres et chaines de caractères).

CPA

La quatrième étape (*Predicate Annotation*) a pour but de trouver une relation (prédicat) entre la colonne-sujet et les autres colonnes pour comprendre le contexte général de la table. Ici, il faut donc reprendre les prédicats sortis par la troisième étape et reprendre celui qui revient le plus souvent.

CTA

La cinquième étape (*Concept and Datatype Annotation*) a pour but de faire correspondre l'entête des colonnes avec des éléments sémantiques tels que les concepts ou les types de données. Le but est donc de récupérer le RDF : type des entités candidates de la colonne et ensuite de voir si des liens sont possibles entre eux. Si oui, il faut récupérer le RDF : type qui revient le plus de fois dans la colonne et qui possède le plus de lien avec les autres RDF : type de la colonne [44].

5.4.3 Post traitement

Ici, il n'y a pas de post-traitement puisqu'il n'y a pas de réévaluation ou de réestimation au niveau des taches CEA, CTA et CPA.

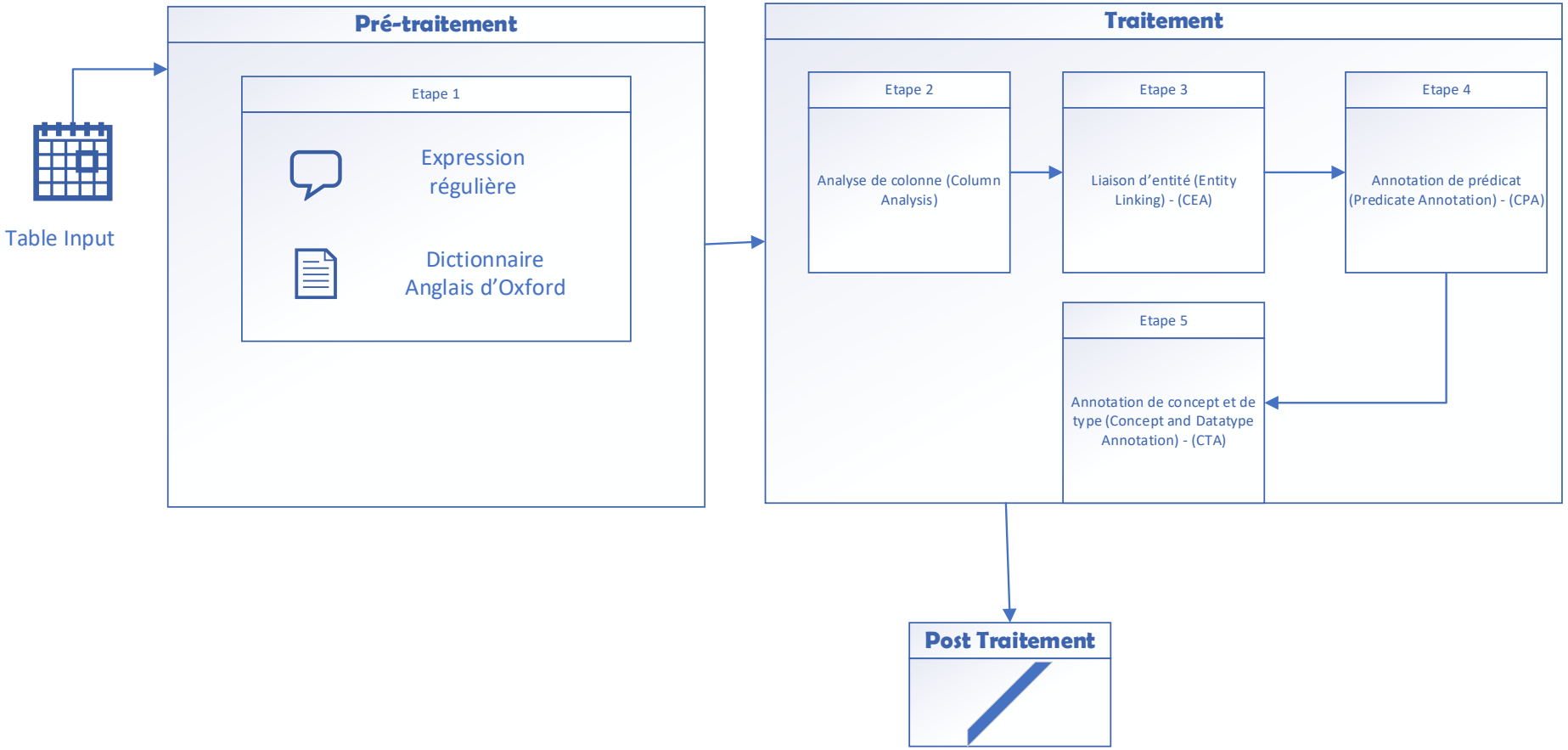


Figure 18 : Fonctionnement MTab 2019

5.5 MantisTable 2020

L'outil MantisTable remplit le même rôle que celui de 2019. Ici, le KB utilisé est Wikidata [45].

Aucune précondition n'est utilisée ici.

Contrairement au MantisTable 2019, il n'utilise pas directement de requête SPARQL. MantisTable va utiliser un autre outil open source appelé LamAPI[14]. LamAPI permet plusieurs choses :

- Correspondance de label : Il utilise la recherche élastique pour retrouver l'entité qui correspond le mieux au texte de la cellule.
- Correspondance de prédicat : Il permet de renvoyer l'ensemble des prédicats entre deux entités.
- Correspondance d'objet/concept : Il permet de renvoyer l'ensemble des objets/concepts lié à ces entités (dbo, foaf…).

La correspondance par label utilise des fichiers JSON qui ont été précalculés. Concernant les autres correspondances, l'outil Redis a été utilisé. Comme moteur de recherche, une recherche élastique supportant « HTTP GZIP » a été mise en place pour de meilleures performances.

Le fonctionnement complet de MantisTable 2020 comprend 8 étapes.

5.5.1 Prétraitement

Cette étape est assez similaire par rapport à celui de 2019 : un traitement est appliqué sur chaque cellule du tableau pour enlever les caractères spéciaux, parenthèses, les espaces en trop…

5.5.2 Traitement

La deuxième étape (*Column Analysis and Subject Detection*) est aussi très similaire à celle de 2019. Le but étant de différencier les entités des littéraux en utilisant les expressions régulières. Pour retrouver la colonne-sujet, ils utilisent des scores basés sur le contenu des colonnes comme pour l'outil de 2019.

La troisième étape (*Data Retrieval*) ne se retrouve pas dans l'outil de 2019. Pour chaque cellule qui a été normalisée, ils vont utiliser la correspondance de label de l'outil

[14] https://bitbucket.org/disco_unimib/lamapi

LamAPI. Si, par exemple, la donnée « Jurassic world » est insérée, une liste de candidats va sortir [45]:

- Q3512046 (" Jurassic World" - Movie);
- Q18615494 (" Jurassic World" - Comic Strip);
- ...

CEA

La quatrième étape est l'étape CEA : pour chaque cellule (entité), un score va être calculé en utilisant l'algorithme de distance de Levenshtein. Plus le texte normalisé de la cellule et de l'entité est proche, plus le score sera élevé.

Pour les colonnes littérales, le traitement est différent si la cellule est de type chaine de caractère, numérique ou bien date.

Pour les types numérique et date, une formule spécialement créée par l'équipe MantisTable a été mise en place. Les dates sont vues comme des valeurs numériques avec ce format YYYYMMDDHHmmSS [45].

Pour les chaines de caractères qui ne sont pas longues, le calcul de score fonctionne comme avec les entités. Pour les chaines de caractères qui sont longues, le calcul se fait via la distance de Jaccard parce qu'il offre un score de similarité plus fiable [45].

CPA

La cinquième étape est l'étape CPA, c'est un processus assez rapide où il faut récupérer l'ensemble des prédicats provenant des couples de candidats. Le candidat ayant l'occurrence la plus élevée sera classé premier et ainsi de suite. Cela permet une diminution du nombre de candidats pour chaque cellule.

CTA

La sixième étape est l'étape CTA, l'outil LamAPI va rechercher l'ensemble des concepts liés aux candidats ressortant de l'étape CPA. Un principe d'occurrence (score de 0 à 1) est encore effectué et ne sera gardé que ceux avec une occurrence supérieure à 0.95. Ensuite, le concept contenant le plus de connexion dans le KG Wikidata est repris.

5.5.3 Post traitement

La septième étape (*Revision*) reprend l'ensemble des données fourni par l'étape CTA et permet de corriger les différentes entités obtenues dans l'étape CEA. Les prédicats sont aussi réarrangés. Maintenant, chaque entité est cohérente avec la colonne et la ligne à laquelle elle appartient [45]. La dernière étape (Export) va permettre d'exporter l'ensemble des résultats obtenus par les étapes précédentes. Cela va faciliter l'évaluation des résultats pour chaque étape.

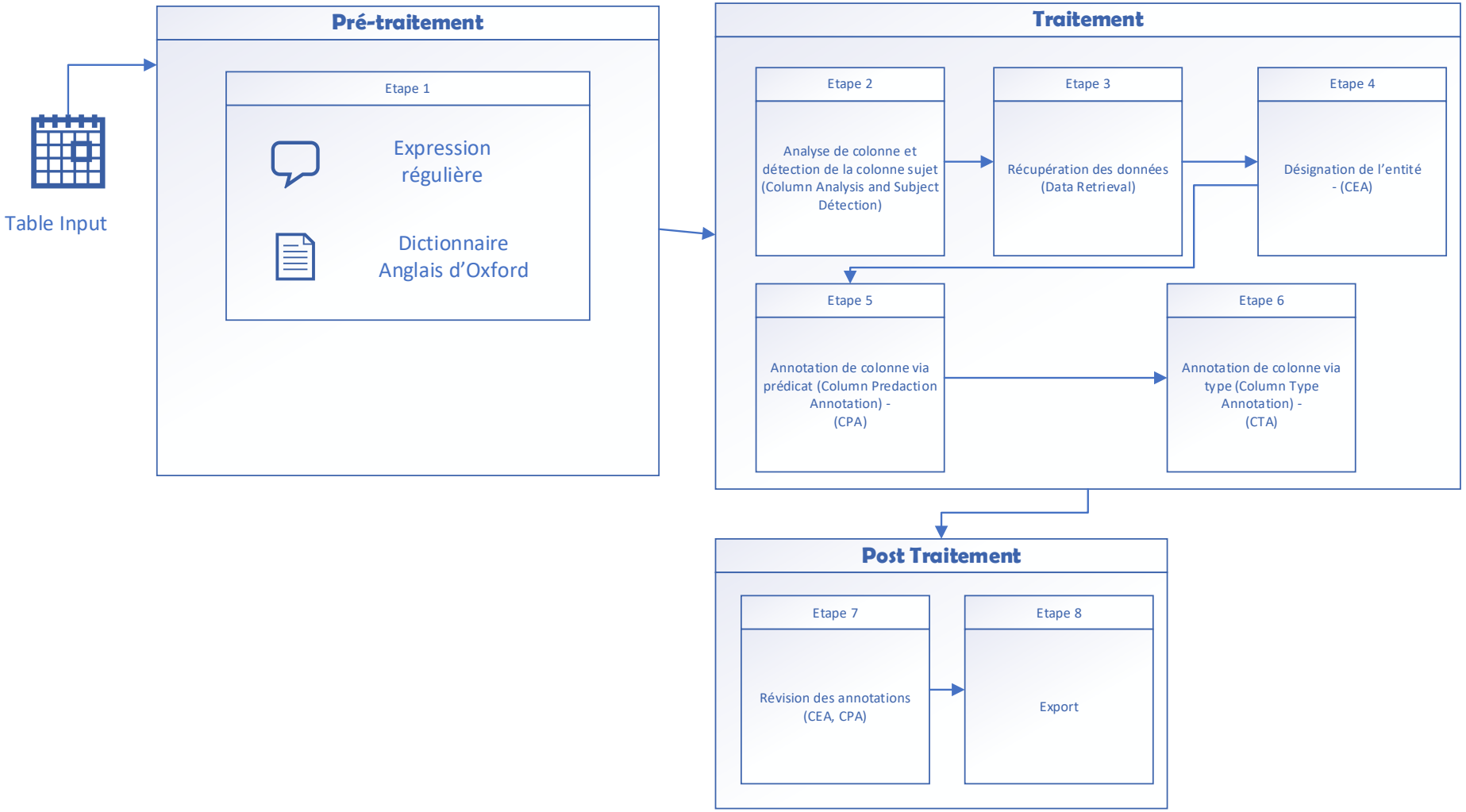


Figure 19 : Fonctionnement MTab 2020

5.6 LinkingPark 2020

Cet outil n'a besoin d'aucune précondition.

LinkingPark est divisé en trois procédures [46]:

- *Entity Linker* qui va générer les candidats et évincer les ambiguïtés des entités.
- *Property Link*er qui va retrouver les propriétés liées aux entités.
- *Type Inference* qui va récupérer le type le plus récurrent provenant des résultats de l'*Entity Linker*.

5.6.1 Prétraitement

C'est dans le processus *Entity Linker* que se trouve le prétraitement. Premièrement, l'API Wikidata MediaWiki va être configurée pour ne renvoyer qu'au maximum 50 candidats.

Deuxièmement, comme pour les autres outils, un correcteur de texte va être appliqué. Ce correcteur est différent des autres outils. Les autres outils utilisent des modèles, des expressions régulières pour rendre le mot le plus compréhensible possible mais, dans le cas de LinkingPark, le correcteur va vérifier toutes les chaines de caractères qui se rapprochent de la chaine de caractères originale et ne reprend que les titres des candidats qu'il aura retrouvés. Cela veut dire que la génération des candidats se fait en même temps que le traitement de texte.

Troisièmement, pour améliorer le renvoi des candidats, une recherche élastique est appliquée sur tous les titres des entités Wikidata [46].

5.6.2 Traitement

CEA

La seconde partie de l'*Entity Linker* concerne la désambiguïsation des entités. Elle se base un algorithme de classification itératif (ICA). Le but étant de ressortir l'entité la plus exacte possible pour chaque cellule. Pour ce faire, il faut que les entités d'une même colonne possèdent des types similaires et que les entités sur une même ligne possèdent des relations entre elles. Pour réussir cette étape, un modèle contenant une phase « *coarse-grained* » est utilisé [46]. Cette dernière va permettre d'enlever les candidats possédant des types incompatibles.

La deuxième phase « *fine-grained* » va sélectionner le meilleur candidat. Cette étape va trancher entre des cas ambigus plus complexes. Par exemple, pour le pays « France », il y a des cellules avec Brest, Paris, etc. Ces cellules correspondent à des villes mais il existe plusieurs villes du nom de Brest dans le monde (Ville de Brest en Biélorussie). Dans ce cas, il faut être sûr de reprendre la ville Brest correspondant à la France. Dans cette phase, il faut donc prendre en compte des valeurs telles que Pays = France.

CPA

Une fois que l'*Entity Linker* a fini son travail, c'est au *Property linker* (CPA) de rentrer en jeu. Cette troisième étape a pour but de retrouver les propriétés des différentes cellules du tableau. Lorsqu'il est possible de trouver des entités pour une colonne, deux correspondances sont effectuées : premièrement, une correspondance directe est utilisée mais si celle-ci ne marche pas, alors, une correspondance via une distance d'édition (comme Levenshtein) utilisant les valeurs des propriétés entités est appliquée. Ensuite, pour chaque colonne, un vote est fait pour reprendre la propriété qui a le plus de chance d'être la propriété de la colonne. Pour les valeurs numériques, un ensemble de caractéristiques est défini par type (la taille d'un être humain, ...). Pour chaque type reçu, l'ensemble des caractéristiques est vérifié pour savoir laquelle correspond le mieux à la colonne numérique. Par rapport aux autres, ils prennent en compte le fait que, sur Wikidata, certaines entités peuvent avoir des propriétés manquantes. Il faut donc étendre les scores de classement en reprenant les propriétés d'entités similaires.

CTA

Une fois le Property Linker terminé, le Type inference (CTA) est mis en place. Pour cette troisième étape, il faut reprendre les résultats de l'entity linker et en reprendre l'ensemble des types via une requête SPARQL. Ensuite, pour définir le type final, il faut reprendre celui qui revient le plus souvent. S'il y a égalité, il prend l'ontologie du type en compte pour rendre la recherche de type encore plus spécifique.

5.6.3 Post-traitement

Ici, il n'y a pas de post-traitement puisqu'il n'y a pas de réévaluation au niveau des taches CEA, CTA et CPA.

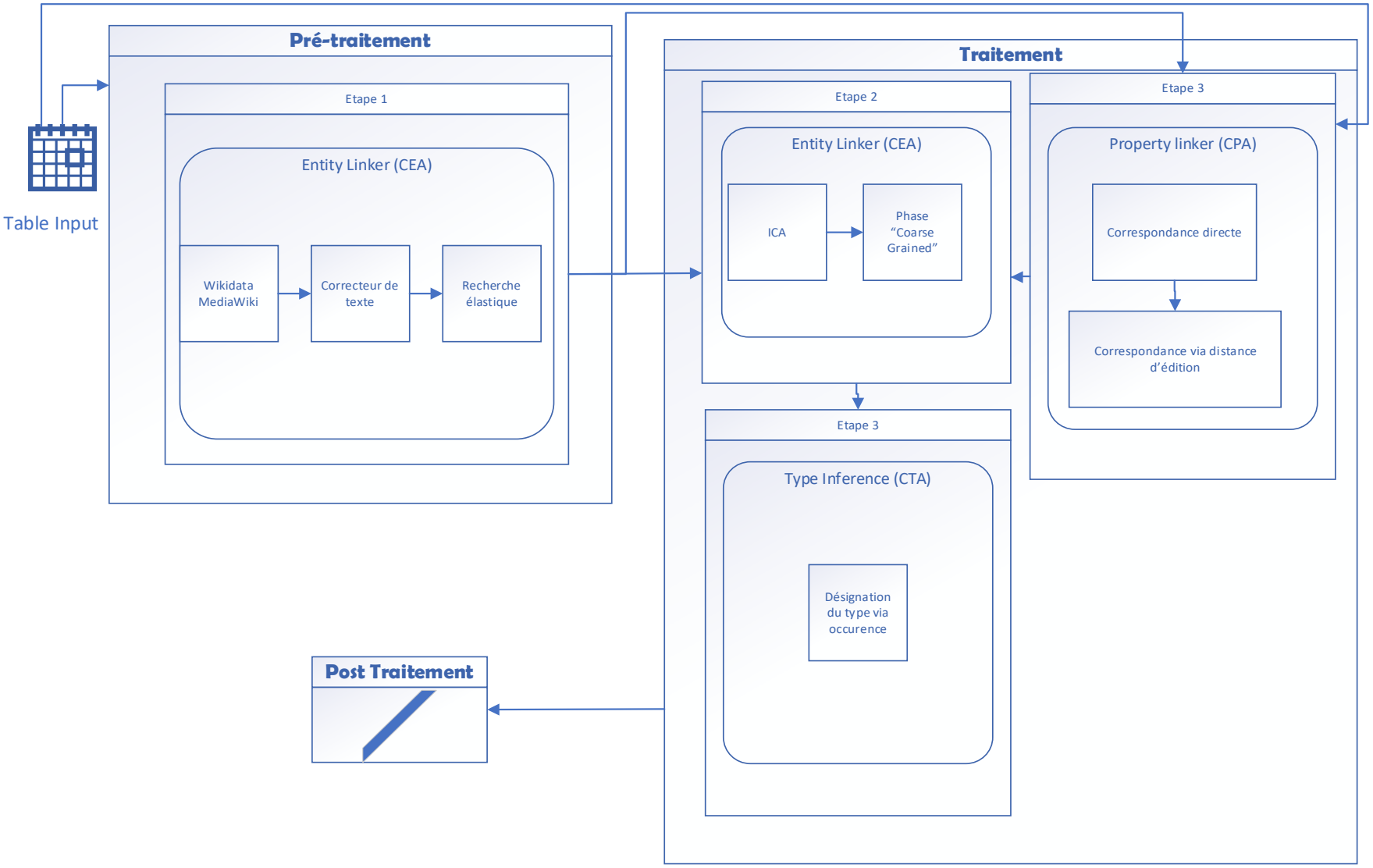


Figure 20 : Fonctionnement LinkingPark

5.7 Dagobah 2019

L'outil MantisTable ne nécessite pas de remplir de préconditions avant de pouvoir mettre en place les principes de STI [47].

Le KB utilisé pour cette version est DBpedia.

5.7.1 Prétraitement

La première étape de l'outil est l'utilisation du DWTC-Extractor[15]. Cet outil va renvoyer des informations importantes sur le tableau. La première information qu'il renvoie correspond à l'orientation du tableau (vertical ou horizontal). Pour ce faire, il va se baser sur 11 types (String, float, date…).

Ensuite, un score va être calculé pour chaque colonne et chaque ligne. Les moyennes de chaque colonne et de chaque ligne sont comparées entre-elles. En fonction du résultat obtenu, une table est définie comme horizontale ou bien verticale.

La seconde information concerne la précision des en-têtes de colonnes. L'outil se base sur le fait que l'en-tête contient une chaine de caractère et qu'il ne partage pas le même type que les cellules de sa colonne. Via ces deux informations, l'outil peut savoir si l'en-tête en question est assez précis pour représenter la colonne.

La troisième information concerne la détection de la colonne clé (sujet). Pour définir cette colonne, l'équipe se base sur le fait que cette dernière contient beaucoup de valeurs uniques et qu'elle se trouve du côté gauche du tableau.

Ces informations sont suffisantes pour commencer le traitement STI [47]. Mais avant ça, un correcteur de texte est appliqué pour résoudre les problèmes de caractères spéciaux, parenthèses…

5.7.2 Traitement

Deux approches sont exploitées pour remplir les tâches CEA, CTA, CPA. La première est appelée approche de base, ou *Baseline Approach* en anglais, et la seconde approche imbriquée, ou *Embedding Approach* en anglais.

1) Baseline Approach

La deuxième étape de l'outil concerne la recherche d'entités candidates. L'équipe utilise plusieurs services de recherche pour récupérer les entités candidates qui les intéressent (Wikidata API, Wikidata Cirrus Search Engine, DBpedia API). Pour reprendre les entités candidates qui sont les plus pertinentes ainsi que leur type, ils vont reprendre celles qui apparaissent le plus souvent dans les recherches. Pour pouvoir faire correspondre les résultats des différents services de recherche, une correspondance est

[15] https://github.com/JulianEberius/dwtc-extractor

faite entre les résultats obtenus par Wikidata et les entités de DBpedia (via SPARQL) [47].

- CTA

  La troisième étape s'occupe de récupérer le type de la colonne. Pour ce faire, ils ne vont reprendre que les types qui reviennent au moins 70% du temps dans la colonne. Ensuite, un calcul reprenant les entités candidates est effectué : le pourcentage d'occurrence du type de la colonne ainsi que le nombre de fois que le type apparait. Avec ces trois informations, le type de la colonne est obtenu.

- CEA

  Grace à l'étape CTA, il est possible de choisir l'entité dans la liste des entités candidates ressortie dans l'étape CEA. Cette quatrième étape concerne la résolution de l'ambiguïté des entités. Pour ce faire, ils vérifient si le premier candidat de la liste possède le type ressorti à l'étape CTA : si oui, alors l'entité candidate est choisie comme entité de la cellule ; si non, ils sélectionnent l'entité candidate qui correspond au type ressorti à l'étape CTA.

2) Embedding Approach

Cette approche prend en compte que les entités d'une même colonne ont des caractéristiques sémantiques communes et forment donc des groupes cohérents.

L'approche imbriquée se fait de la manière suivante [47]:

Pour la deuxième étape, ils utilisent le *pre-trained Wikidata embedding* [50] mais celui-ci n'ayant pas toutes les informations (alias, label…), il faut donc y rajouter une recherche élastique.

La stratégie utilisée pour récupérer les entités candidates est l'utilisation de REGEX ainsi que de l'algorithme de Levenshtein. Pour qu'un candidat soit repris, il doit avoir un pourcentage Levenshtein supérieur à 75% et/ou l'entité candidate doit contenir tous les mots se trouvant dans la cellule.

Ensuite, une stratégie de recherche par grille est appliquée. Celle-ci va calculer la précision du regroupement des candidats et en ressortir le groupement avec le taux de précision le plus élevé.

- CTA

  La troisième étape est de récupérer les types ayant un score d'occurrence élevé par groupement de candidats. Ensuite, ils récupèrent le type ayant le plus de spécificité par rapport à DBpedia (sous classes, owl:Thing).

- CEA

  Si l'entité candidate possède un type correspondant au résultat sorti durant l'étape CTA alors un score de 1.5 lui est attribué. S'il possède un type parent au résultat de l'étape CTA, alors un score de 1.0 lui est donné. Avec ce score, un nouvel algorithme est produit. Ce dernier va ressortir l'entité recherchée.

- CPA

  Que ce soit pour l'approche de base ou l'approche imbriquée, la méthode utilisée pour le CPA est toujours la même. Pour cette cinquième étape, soit une technique de recherche est utilisée sur les en-têtes des colonnes (cela ne donne pas une bonne précision) soit une recherche de relation entre les instances de deux colonnes avec un vote majoritaire est appliquée (cela offre une meilleure précision).

5.7.3 Post-traitement

Ici, il n'y a pas de post-traitement puisqu'il n'y a pas de réévaluation au niveau des taches CEA, CTA et CPA.

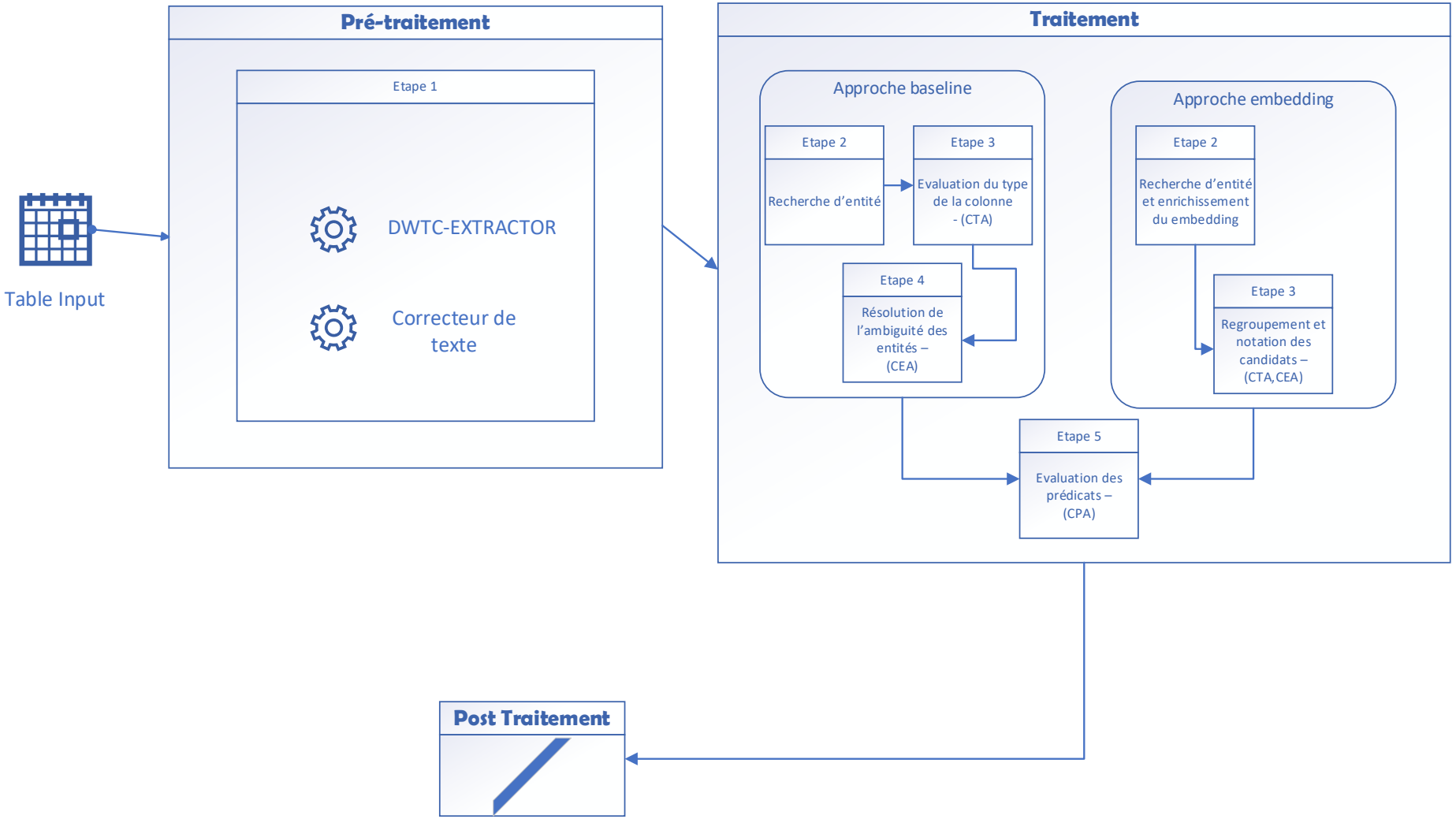


Figure 21 : Fonctionnement Dagobah 2019

5.8 Dagobah 2020

L'outil Dagobah remplit le même rôle que celui de 2019. Ici, le KB utilisé est Wikidata.

Aucune précondition n'est utilisée ici.

Par rapport à Dagobah 2019, l'approche de base a été abandonnée pour laisser entièrement la place à l'approche imbriquée [49].

5.8.1 Prétraitement

Pas de prétraitement pour cette version de l'outil.

5.8.2 Traitement

La première étape consiste à retrouver les entités candidates reliées aux différentes cellules. Cette étape correspond à la première étape de l'approche imbriquée de Dagobah 2019. Au niveau du REGEX, ils restent sur le principe que l'alias doit contenir l'ensemble des mots et peu importe l'ordre dans lequel ils sont. Pour l'algorithme de Levenshtein, le ratio passe à 65% (au lieu de 75% par rapport à l'année passée). Le ratio a été réduit pour être sûr d'avoir au moins un candidat par cellule. Un nombre maximum de 50 candidats est retenu par cellule [49] .

Ils ont décidé de créer leur propre service de recherche puisque Wikidata API a trop de restrictions (connexion simultanée, la taille des ensembles de résultats, temps des requêtes…). Ce nouveau service de recherche comprend le Wikidata Toolkit. Seules les entités qui possèdent un QID et les documents qui possèdent un PID sont utilisés. La recherche a été faite en python et possède des performances assez pauvres. Leur recherche a tourné pendant 11 jours pour le Round 3. Une optimisation sera possible en intégrant la librairie Cython Levenshtein[16] qui réduira considérablement le temps de recherche [49] .

La deuxième étape a pour but de donner un score de confiance aux différents candidats sortant de l'étape 1. Le but est de calculer la ratio Levenshtein le plus élevé entre la cellule et l'ensemble des labels des candidats. Pour calculer ce ratio, il faut aussi récupérer l'ensemble des cellules se trouvant sur la même ligne que la cellule où est appliquée la formule [49] . Par exemple, si une cellule contient un texte « University College Cork » et que l'entité candidate est « Q1574185 » alors son score sera de 1 puisque « Q1574185 » possède exactement le nom de cette université en label et toutes les cellules se situant sur la même ligne possèdent des informations liées à « Q1574185 ».

[16] https://pypi.org/project/python-Levenshtein/

CPA

La troisième étape est de retrouver les relations entre les colonnes. Le tableau est divisé en deux parties, la partie « tête » qui contient les entités candidates de la colonne « tête » et la partie « queue » qui contient les entités candidates de la colonne « queue ». Pour calculer le score, il a fallu prendre en compte les types des valeurs:

- Si des identifiants d'entités sont comparés, alors le score est égal à 1 si les identifiants sont les mêmes sinon il est égal à 0.
- Si des valeurs numériques sont comparées, alors le score sera égal à 1 – (num1-num2) / (num1+num2).
- Si des chaines de caractères sont comparées, l'algorithme de Levenshtein est utilisé.
- Si des dates sont comparées, alors, le score sera égal à 1 si des dates correspondent même si elles ont des formats différents.

La relation qui aura l'occurrence la plus élevée sera donc reprise et, s'il y a égalité, ce sera celle qui aura obtenu le score le plus élevé via les critères précédemment cités qui sera reprise.

CEA

La quatrième étape est donc de définir l'entité finale à la cellule correspondante. Pour cela, les scores obtenus sont récupérés à l'étape deux et un score de 1 y est ajouté si l'étape CPA arrive à retrouver les relations avec l'entité en question par rapport aux autres cellules. En augmentant le score de 1, le candidat a beaucoup plus de chance d'être choisi comme l'entité finale.

CTA

La cinquième étape consiste à récupérer le type des entités d'une même colonne. Pour ce faire, un vote majoritaire est fait sur toutes les entités d'une même colonne. Les types sont répartis en 3 niveaux :

Le niveau 0 correspond au type lié au prédicat « instance de » de l'entité CEA.

Le niveau 1 correspond au type lié au prédicat « sous-classe de » de l'entité CEA.

Le niveau 2 correspond au type de l'ancêtre de l'entité CEA.

Le type privilégié est donc le type direct. Un système de classement est aussi utilisé. Celui-ci respecte les principes d'annotations de Wikidata. Il existe donc 3 rangs : PREFERED-2, NORMAL-1, DEPRECATED-0.

Ensuite, tous les types des 3 niveaux sont récupérés et ils prennent les types qui reviennent le plus souvent. S'il y a une égalité, il faut d'abord prendre en compte l'occurrence, ensuite le niveau et finalement le rang.

Finalement, une relation est recherchée entre les différentes colonnes et s'il en existe plusieurs alors la plus spécifique est reprise. S'il n'en existe pas, c'est le type étant le plus proche de l'entité de la cellule qui est repris. S'il y a encore une égalité, c'est un choix aléatoire qui est fait.

5.8.3 Post-traitement

Ici, il n'y a pas de post-traitement puisqu'il n'y a pas de réévaluation au niveau des tâches CEA, CTA et CPA.

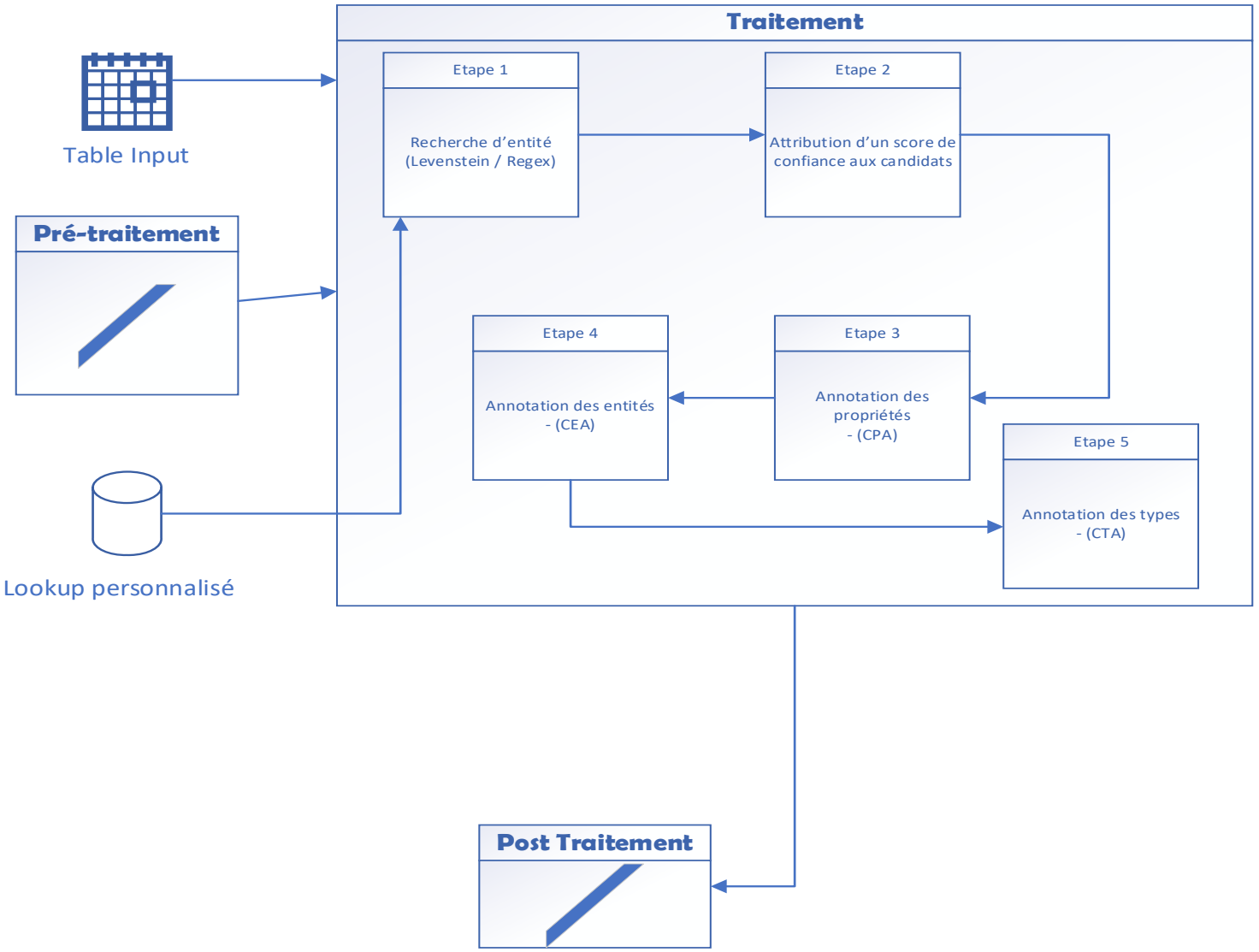


Figure 22 : Fonctionnement Dagobah 2020

5.9 Synthèse des différents outils

Chaque équipe a répondu au challenge SemTab via leurs propres méthodes. Ces équipes ont répondu aux étapes CEA, CTA, CPA à différents moments dans leur algorithme. Pour l'outil MTab, ces étapes sont réalisées durant le post-traitement. MantisTable 2020 le fait durant le traitement mais également durant le post-traitement. Dans le post-traitement, il va faire une réévaluation des étapes CEA et CPA. Le reste des outils répond aux différentes étapes durant la phase de traitement.

Chaque année, MTab reçoit la première place du défi SemTab. Il serait donc logique de penser qu'il serait plus intéressant de faire un post-traitement comme le leur pour être plus précis. Mais pour certains rounds, les autres outils présentés ci-dessous arrivent à l'emporter. La corrélation entre la précision des annotations et la position où les étapes CEA, CTA et CPA sont réalisées est insignifiante.

Outils	Prétraitement			Traitement			Post-traitement		
	CEA	CTA	CPA	CEA	CTA	CPA	CEA	CTA	CPA
Mtab2019	✗	✗	✗	✗	✗	✗	✓	✓	✓
Mtab2020	✗	✗	✗	✗	✗	✗	✓	✓	✓
MantisTable2019	✗	✗	✗	✓	✓	✓	✗	✗	✗
MantisTable2020	✗	✗	✗	✓	✓	✓	✓	✗	✓
LinkingPark	✗	✗	✗	✓	✓	✓	✗	✗	✗
Dagobah2019	✗	✗	✗	✓	✓	✓	✗	✗	✗
Dagobah2020	✗	✗	✗	✓	✓	✓	✗	✗	✗

Figure 23 : comparaison traitements & étapes STI

5.9.1 Prétraitement

La plupart des outils possèdent une étape de prétraitement sauf Dagobah 2020 qui utilise son propre service de recherche pour retrouver les candidats qui l'intéressent. L'objectif principal de cette étape est de corriger les fautes de frappe, évincer les caractères spéciaux, … Pour permettre de rendre le tableau d'entrée le plus correct possible et donc de retrouver les candidats qui ont le plus de chance de correspondre à ce tableau. La logique reste donc plus ou moins la même pour chaque outil. La différence se situe au niveau des modèles, des correcteurs qui ont été utilisés pour changer les inputs. Chaque correcteur a sa propre façon de fonctionner. Contrairement aux autres outils, l'outil MTab propose une recherche multilingue. Cela permet de couvrir un plus grand nombre de données.

5.9.2 Traitement

Au niveau du traitement, l'une des principales tâches de cette étape est la recherche de candidats. Pour y parvenir, les outils utilisent différentes méthodes telles que la recherche via un service, KG Endpoint, leur propre outil de recherche … En plus de ces derniers, le type de recherche diffère d'une version d'outil à l'autre et aussi d'un outil à l'autre.

L'outil MTab 2019 se base sur un score de pertinence pour récupérer les candidats souhaités, alors que MTab 2020 fait une recherche de cellule par Levenshtein ou bien une recherche via deux cellules en utilisant les relations entre des cellules d'une même ligne.

Pour MantisTable 2019, ce dernier va faire faire une correspondance entre les cellules et les entités via la distance Wagner-Fischer. C'est donc un autre algorithme qui est mis en place ici. L'algorithme de Levenshtein et Wagner-Fischer ont le même but mais sont programmés différemment. Via cette correspondance, MantisTable 2019 permet de répondre à l'étape CEA. Pour les étapes CTA et CPA, ils vont recevoir un ensemble de types et de prédicats pour ensuite décider reprendre ceux qui reviennent le plus souvent.

Pour MantisTable 2020 va utiliser l'outil LamAPI pour des raisons de performance, de recherche plus précise. Ici, l'avantage est que l'équipe MantisTable n'a pas eu besoin de développer d'outils/méthodes de recherche pour retrouver les candidats (cet outil utilise une recherche élastique et permet de renvoyer des candidats). Pour répondre à l'étape CEA, ils utilisent un principe de score utilisant la distance de Levenshtein (celui ayant le plus haut score est considéré comme l'entité). Comme pour MantisTable 2019, pour répondre aux étapes CTA et CPA, ils vont recevoir un ensemble de types et de prédicats pour ensuite décider reprendre ceux qui reviennent le plus souvent.

LinkingPark contrairement à MantisTable 2020 a créé ses propres recherches élastiques (phases « *coarse-grained* » et « *fine-grained* »). L'équipe LinkingPark a décidé d'utiliser ces recherches pour améliorer le nombre de candidats correctement attribués lors des générations de candidat. Via ces recherches, ils répondent à l'étape CEA. Pour l'étape CTA, ils vont recevoir un ensemble de types et de prédicats pour ensuite décider reprendre ceux qui reviennent le plus souvent. Concernant l'étape CPA, chaque ligne vote pour trouver la propriété qui convient le mieux.

Pour Dagobah 2019, ils ont décidé d'utiliser deux méthodes différentes (*baseline* et *embedding*) et de les comparer. Lors du round 1 du défi SemTab, les deux méthodes ont été comparées et c'est la méthode *baseline* qui a eu les meilleurs résultats. Cette dernière a donc été utilisée pour le reste des rounds. Pour répondre à l'étape CTA, l'approche *baseline* va reprendre le type qui revient le plus souvent et va ensuite utiliser cette information pour retrouver l'entité correspondant à la cellule ; ce qui répondra à l'étape

CEA. Contrairement à l'approche *embedding* qui se base sur des scores pour retrouver l'entité correspondant à la cellule (CEA) et pour l'étape CTA, l'outil va reprendre le type qui revient le plus de fois. Que ce soit la méthode *embedding* ou *baseline*, l'étape CPA est résolue de la même manière soit une recherche est utilisée pour récupérer la bonne propriété (mais peu précis) soit un vote majoritaire est effectué par colonne pour savoir quelle propriété est la bonne.

Pour Dagobah 2020, l'approche de base a été abandonnée par l'équipe Dagobah et ils ont décidé de reprendre une version améliorée de leur recherche imbriquée qui a donné des résultats très probants. Pour répondre à l'étape CPA, un score est établi via différentes métriques et le score le plus élevé est repris. L'étape CEA va se baser sur l'étape CPA et, via un système de score, va donner l'entité correspondante à la cellule. Pour l'étape CTA, un système de classement et d'occurrence est utilisé pour retrouver le type.

Via ces différents points, il est facile de remarquer que les étapes CEA, CTA, CPA sont liées entre elles. Certains outils se basent sur l'étape CEA pour répondre aux CTA et CPA mais d'autres outils utilisent l'étape CTA ou bien CPA pour répondre aux autres étapes.

Dans la plupart des cas, les outils vont essayer d'identifier les littéraux comme des dates, des numériques ou bien des strings pour retrouver plus facilement les valeurs des KG y correspondant. MantisTable est un des seuls outils avec Dagobah à identifier la colonne sujet d'un tableau. C'est une analyse importante pour notre mémoire puisque l'intégration d'ensembles de données se basent sur des colonnes-sujet.

5.9.3 Post-traitement

Au niveau du post traitement :

Pour MTab 2019, un ensemble de probabilités va être agrégé pour retrouver l'entité correspondant à la cellule (CEA). En fonction de ces probabilités, un vote majoritaire va être appliqué aux étapes CPA et CTA.

Pour MTab 2020, l'outil va calculer la similarité entre les cellules et les candidats. Et celui avec la plus grande similarité sera considéré comme l'entité correspondant à la cellule. Ensuite, un vote majoritaire va être mis en place pour répondre aux étapes CEA et CPA.

Pour MantisTable 2020, les étapes CEA et CPA vont être réestimées sur base du CTA. Ce qui va améliorer la précision des annotations.

Outils	Pré traitement	Traitement	Post Traitement
Mtab 2019	-Suppression des caractères spéciaux -Corriger les fautes de frappe (ex : espaces en trop) -Résoudre les erreurs d'écriture -Recherche multilingue	-Utilisation d'une recherche pour récupérer des candidats avec score de pertinence -Différencier les colonnes entités et numériques -Etablissement de relations entité-entité ou entité-non entité	-Agrégation de probabilité pour résoudre l'étape CEA -Mise en place d'un vote majoritaire (en fonction des probabilités) pour résoudre les étapes CPA et CTA
Mtab 2020	-Récupération de révision historique Wikidata -Utilisation de tables de hachage pour récupérer des entités multilingues	-Recherche de candidats par cellule. -Recherche de candidats via deux cellules.	-Calcul de similarité entre les candidats et les cellules (CEA) -Mise en place d'un vote majoritaire (CPA et CTA)
MantisTable 2019	-Résolution des textes en minuscule -Résolution des abréviations/acronymes -Correction des unités de mesure	-Différencier les colonnes entités et littéraux -Analyse de colonne pour retrouver la colonne-sujet -Mapping entre les cellules et les entités du KG (CEA) -Principe d'occurrence (CPA et CTA)	Pas de post-traitement

MantisTable 2020	-Résolution des textes en minuscule -Résolution des abréviations/acronymes -Correction des unités de mesure	-Différencier les colonnes entités et littéraux -Analyse de colonne pour retrouver la colonne-sujet -Recherche de candidat via LamAPI -Mise en place de scores (CEA) -Principe d'occurrence (CPA et CTA)	-Réévaluation des candidats (CEA et CPA)
LinkingPark 2020	-Résolution de l'orientation du tableau -Traitement de texte par recherche de candidats	-Choix du meilleur candidat via des phases « *coarse-grained* » et « *fine-grained* » (CEA) -Mise en place du principe d'occurrence (CTA) -Mise en place d'un vote majoritaire (CPA)	Pas de post-traitement
Dagobah 2019	-Renvoi d'informations importantes sur le tableau (orientation, en-tête…) -Recherche de la colonne clé (sujet)	-Deux types de recherches : *baseline* et *embedding* -Baseline : Recherches et SPARQL pour retrouver les candidats -Baseline : Calcul avec principe d'occurrence est mis en place (CTA) -Baseline : Evaluation CEA via CTA. -Embedding : Recherche de candidats via Wikidata Embedding	Pas de post-traitement

		-Embedding : Mise en place de score pour l'étape CEA -Embedding : Mise en place du principe d'occurrence pour l'étape CTA -Utilisation d'une recherche ou de vote majoritaire pour l'étape CPA	
Dagobah 2020	Pas de prétraitement	-Recherche de candidats via leur propre service de recherche -Attribution de score de confiance - Mise en place de score pour l'étape CPA - Mise en place de score pour l'étape CEA -Utilisation du principe d'occurrence et de distance pour l'étape CTA	Pas de post-traitement

Tableau 14 Synthèse des différents outils

5.9.4 Comparaison de résultats défi SemTab

Que ce soit pour le SemTab 2019 ou 2020, l'outil MTab sort grand gagnant. L'outil se retrouve très souvent premier durant chaque round. Sa méthode d'annotation peut être considérée comme la meilleure.

L'outil LinkingPark arrive second au classement pour 2020. Dans certains rounds, il a su prendre le pas sur l'outil MTab surtout pour l'étape CTA où il gagne sur chaque round sauf le dernier. Pour les autres étapes, LinkingPark arrive très souvent $2^{\text{ème}}$ ou $3^{\text{ème}}$.

L'outil Dagobah a eu la plus grande progression. Lors du SemTab 2019, il n'a pas su avoir des résultats très probants. Il était rarement dans le top 3 voir top 5 durant chaque round. Ceci est surement dû à l'approche *baseline* qui n'était pas suffisamment précise. Lors du SemTab 2020, ils ont utilisé l'approche *embbeding* et ont réussi à arriver troisième au classement. Il n'a pas su remporter un seul round mais il est resté assez constant avec des places de deuxième ou troisième.

MantisTable n'a pas su arriver dans le top 3 mais ce dernier a réussi à avoir de très bons résultats. Arrivant même à égalité avec MTab sur certains rounds. Lors du SemTab 2019, il a réussi à prendre quelques rounds à l'outil Dagobah.

Via l'ensemble de ces résultats, nous avons dû choisir quels outils nous allions utiliser durant notre mémoire. Nous avons décidé d'utiliser l'outil MantisTable 2019 et Mtab 2019. Nous voulions des outils utilisant exclusivement DBpedia parce que notre travail ne se base que sur ce KB (on ne reprend donc pas les outils de 2020). LinkingPark ne faisait pas partie du SemTab 2019 et n'a pas été repris. Nous avons repris l'outil Mtab 2019 parce que ce dernier remporte haut la main le SemTab 2019 ce qui fait de lui le meilleur candidat pour avoir les meilleures annotations. Nous avons aussi repris MantisTable 2019 parce que cette équipe possède un algorithme de détection de colonne sujet qui nous sera très utile pour la partie pratique de notre projet. Dagobah n'a pas été choisi puisque l'algorithme il n'est pas aussi performant que Mtab et il n'a pas un algorithme de détection de colonne tel que MantisTable.

Round	Taches	CEA			CTA			CPA		
	Outils	F1 score	Précision	Rank	F1 score	Précision	Rank	F1 score	Précision	Rank
Round 1	Mtab 2019	1.0	1.0	2	1.0	1.0	1	0.987	0.975	1
	Dagobah 2019	0.897	0.941	3	0.644	0.580	9	0.415	0.347	4
	MantisTable 2019	1.0	1.0	/	0.929	0.929	/	0.965	0.991	/
	Mtab 2020	0.987	0.988	1	0.885	0.884	2	0.971	0.991	1
	LinkingPark	0.987	0.988	2	0.926	0.926	1	0.967	0.978	2
	Dagobah 2020	0.922	0.944	6	0.834	0.854	6	0.914	0.962	7
	MantisTable 2020	0.982	0.989	3	0.746	0.753	9	0.888	0.942	8
	Mtab 2019	0.911	0.911	1	1.414	0.276	1	0.881	0.929	1
	Dagobah 2019	0.713	0.816	8	0.641	0.247	7	0.533	0.919	5
	MantisTable 2019	0.616	0.673	/	1.049	0.247	/	0.460	0.544	/

Round 2	Mtab 2020	0.995	0.995	1	0.984	0.984	2	0.997	0.997	1
	LinkingPark	0.993	0.993	3	0.984	0.985	1	0.993	0.994	2
	Dagobah 2020	0.993	0.993	2	0.983	0.983	3	0.992	0.994	3
	MantisTable 2020	0.991	0.993	4	0.966	0.973	4	0.961	0.966	6
Round 3	Mtab 2019	0.970	0.970	1	1.956	0.261	1	0.844	0.845	1
	Dagobah 2019	0.725	0.745	7	0.745	0.161	7	0.519	0.826	6
	MantisTable 2019	0.633	0.679	/	1.648	0.269		0.518	0.595	/
	Mtab 2020	0.991	0.992	1	0.976	0.976	2	0.995	0.955	1
	LinkingPark	0.986	0.986	2	0.978	0.979	1	0.985	0.988	3
	Dagobah 2020	0.985	0.985	3	0.974	0.974	3	0.993	0.994	2
	MantisTable 2020	0.974	0.979	4	0.958	0.965	4	0.941	0.957	5
	Mtab 2019	0.983	0.983	1	2.012	0.300	1	0.832	0.832	1

Round 4	Dagobah 2019	0.578	0.599	8	0.684	0.206	7	0.398	0.874	7
	MantisTable 2019	0.973	0.983	/	1.682	0.322	/	0.787	0.841	/
	Mtab 2020	0.993	0.993	1	0.981	0.982	1	0.997	0.997	1
	LinkingPark	0.985	0.985	2	0.953	0.953	4	0.985	0.988	5
	Dagobah 2020	0.984	0.985	3	0.972	0.972	3	0.995	0.995	3
	MantisTable 2020	0.812	0.985	9	0.725	0.989	8	0.803	0.988	7

Tableau 15 Tableau récapitulatif des résultats du défi Semtab par round

5.10 Conclusion

Cette section a permis d'y voir plus clair sur les différentes idées mises en place dans la création d'outils STI. Malgré des algorithmes et des méthodes de recherche différents, la plupart des outils ont une phase de prétraitement qui permet de rendre leurs données plus faciles à traiter, les phrases de traitement et post-traitement vont répondre aux étapes CPA, CTA, CEA.

Chaque phase a son importance. Par exemple, si le prétraitement est mal effectué, les données du tableau risquent d'être difficiles à retrouver sur le Web et cela aura donc un impact négatif sur les étapes CPA, CTA, CEA.

Via cette section, deux outils sont sortis du lot et vont donc être utilisés pour notre travail pratique.

Nous avons décidé de reprendre l'outil MTab qui est arrivé en premier pour les étapes de 2019/2020 pour le défi SemTab. Nous avons aussi repris l'outil MantisTable avec sa détection de colonne-sujet qui est un atout manquant de l'outil Mtab.

Via ces deux outils, il nous est possible d'apporter deux ajouts au domaine STI. Premièrement, nous allons être capable d'automatiser une des préconditions de l'outil MTab (la colonne sujet doit être la première colonne du tableau) et deuxièmement, il va être possible de comparer au niveau du schéma et des entités plusieurs jeux de données via l'outil MTab et notre propre algorithme.

6 Chapitre 6 : Création d'un outil d'intégration d'ensemble de données via STI

6.1 Introduction

Après avoir énuméré les différents outils/méthodes correspondants au STI, nous allons mettre en avant une ébauche d'une nouvelle solution d'interprétation efficiente et également un post-traitement pour permettre l'inférence de schémas entre différents tableaux.

Nous allons illustrer des cas pratiques grâce à notre propre algorithme d'intégration d'ensemble de données. Nous travaillerons sur un ou plusieurs tableaux au format CSV.

Premièrement, nous allons nous baser sur l'outil MantisTable 2019, il sera retravaillé afin de pouvoir utiliser uniquement les étapes de préparations de données et de détection de la colonne-sujet. Cette première étape va permettre d'identifier la colonne-sujet du tableau et un premier nettoyage des données en vue de son intégration dans l'outil suivant.

Ensuite nous utiliserons l'outil MTab 2019 pour créer nos annotations dans les différents tableaux. Enfin, nous utiliserons notre propre algorithme pour joindre les différentes tables grâce aux annotations définies précédemment.

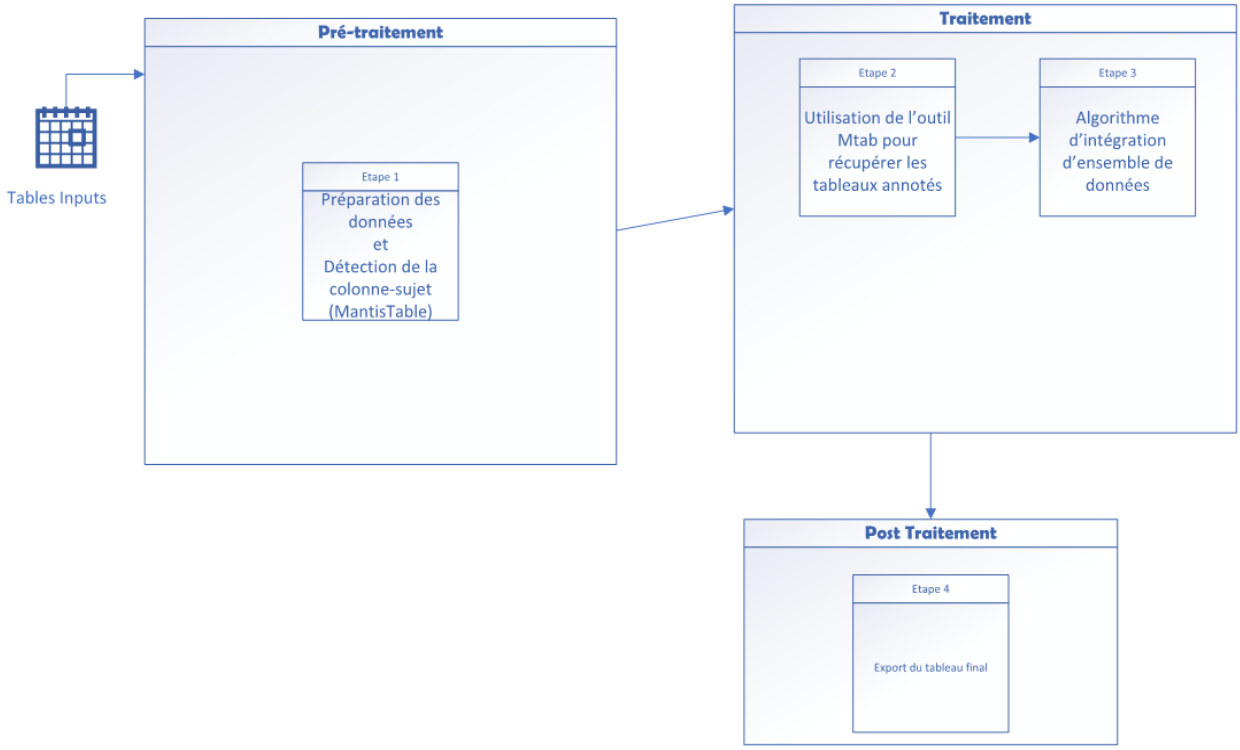


Figure 24 Architecture globale de la partie pratique

6.2 Prérequis

Quelle méthode/outil permettrait d'automatiser efficacement la comparaison de LOD afin d'en retirer les données communes aux différents schémas ? Pour répondre à cette question nous devons d'abord définir notre méthode de travail ainsi que les outils et langages utilisés.

Outils

Nous avons décidé d'utiliser deux outils existants avec notre propre algorithme d'intégration de données. Tout d'abord, nous allons utiliser les étapes de préparations de données et de détection de la colonne-sujet de l'outil MantisTable 2019. Une fois la colonne-sujet détectée, celle-ci va être automatiquement placée en tant que première colonne du tableau. C'est une étape importante puisque le second outil utilisé est l'outil MTab 2019. Ce dernier a comme précondition d'avoir la colonne-sujet comme première colonne du tableau. C'est MTab qui va nous permettre de créer les annotations nécessaires à l'utlilisation de notre propre algorithme de comparaison de tableaux.

Plusieurs raisons nous ont poussé à prendre ces outils :

- MantisTable

L'équipe qui l'a créé est très investie dans les travaux d'interprétation sémantique de tableaux. Ils ont participé aux challenges SemTab 2019 [42] et 2020[17] on peut donc s'attendre à de nouvelles versions de cet outil qui seront de plus en plus performantes. C'est aussi la seule équipe qui a proposé un algorithme de détection de colonne-sujet pour les défis SemTab. Que ce soit pour l'année 2019 ou 2020, ils ont toujours gardé leur algorithme de détection de colonne sujet.

- MTab

Que ce soit pour le défi SemTab 2019 ou 2020, cet outil a eu les meilleurs résultats[18] au niveau de l'application des principes STI (CEA, CTA, CPA). Il est donc logique de récupérer cet outil comme base pour récupérer les URI correspondant à chaque cellule du tableau.

Durant nos différentes lectures, nous avons été plus amenés à lire des requêtes SPARQL sur DBpedia, nous avons donc choisi ce KB qui nous semblait plus simple à comprendre et à utiliser. Nous nous sommes donc reposés sur les versions 2019 des outils parce qu'ils se concentrent surtout sur DBpedia.

[17] http://www.cs.ox.ac.uk/isg/challenges/sem-tab/2020/results.html

[18] http://www.cs.ox.ac.uk/isg/challenges/sem-tab/2019/results.html & http://www.cs.ox.ac.uk/isg/challenges/sem-tab/2020/results.html

Langage

Au niveau du langage utilisé pour notre méthode, nous avons choisi Python avec les librairies *Panda* et *Selenium*.

Python est un langage de programmation polyvalent car il permet une lecture et une écriture facile pour nos fichiers csv. Il possède également une syntaxe très simple à maitriser. D'ailleurs il a été utilisé dans les outils MantisTable et MTab. Etant donné que nous avons dû retravailler avec ces outils il était beaucoup plus judicieux de garder le même langage plutôt que de tout réécrire.

Panda est une librairie largement utilisée dans la science des données, de même que la bibliothèque *Selenium* qui va nous permettre de faire des requêtes HTTP et d'extraire des tableaux HTML provenant de l'outil MTab.

Méthode

Notre méthode d'approche STI se base sur quatre étapes essentielles :

- Etape 1 : Préparation des données et détection de la colonne-sujet (MantisTable).
- Etape 2 : Annoter et récupérer les tableaux (MTab).
- Etape 3 : Comparer les tableaux (Algorithme d'intégration d'ensemble de données).
- Etape 4 : Export du tableau de résultats.

Notre projet est disponible sur Github[19].

6.3 Etape 1 : Préparation des données et détection de la colonne-sujet (MantisTable)

MantisTable est un outil d'interprétation de tableaux open source qui se base un KG, DBpedia, pour retrouver ses annotations. Il dispose d'une interface graphique pour permettre l'analyse des résultats [51]. Nous utiliserons la version 3, celle-ci utilise durant le SemTab 2019, disponible sur Bitbucket[20].

Les seuls prérequis à l'utilisation de MantisTable sont Docker[21] et Node8+[22].

[19] https://github.com/Cabi-96/STI-Thesis/tree/master
[20] https://bitbucket.org/disco_unimib/mantistable-tool-3/src/master/
[21] https://www.docker.com/
[22] https://nodejs.org/

6.3.1 Import de la table

MantisTable nécessite l'import d'un fichier JSON. Comme nous travaillons avec des fichiers de type CSV, nous avons décidé d'utiliser un convertisseur en ligne csvjson[23] pour convertir directement notre fichier CSV en un fichier JSON utilisable par MantisTable.

Dans le formulaire d'import de fichier, nous avons désactivé le choix du Gold Standard car il n'est pas utile pour cette partie, en effet nous ne créons pas d'annotations ici.

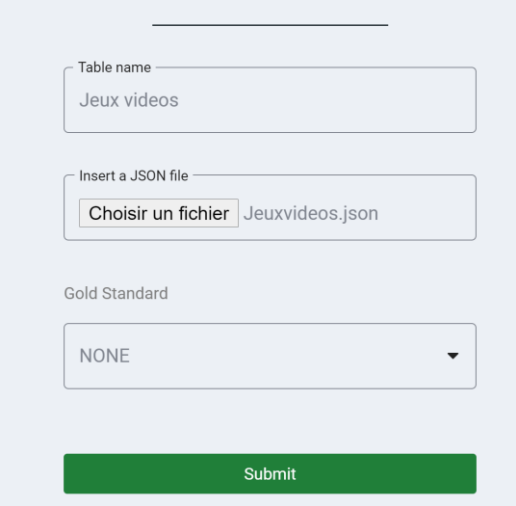


Figure 25 Formulaire d'insertion de fichier JSON dans MantisTable

Une fois le fichier importé, il est directement converti en table et celle-ci est disponible dans l'application.

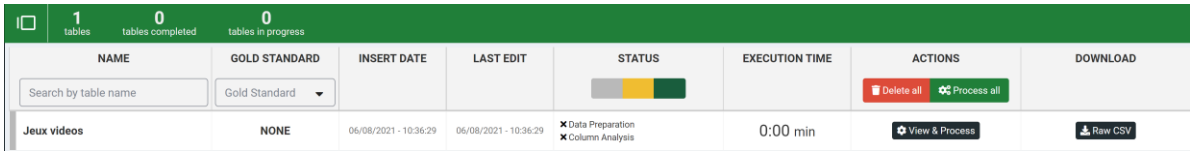


Figure 26 Liste des tables importées dans MantisTable

Pour compléter la liste des tables, nous avons rajouté une colonne « Download ». Celle-ci permet de récupérer la table directement convertie en CSV sans autres modifications.

Les autres colonnes restent inchangées par rapport à l'application initiale.

[23] https://csvjson.com/

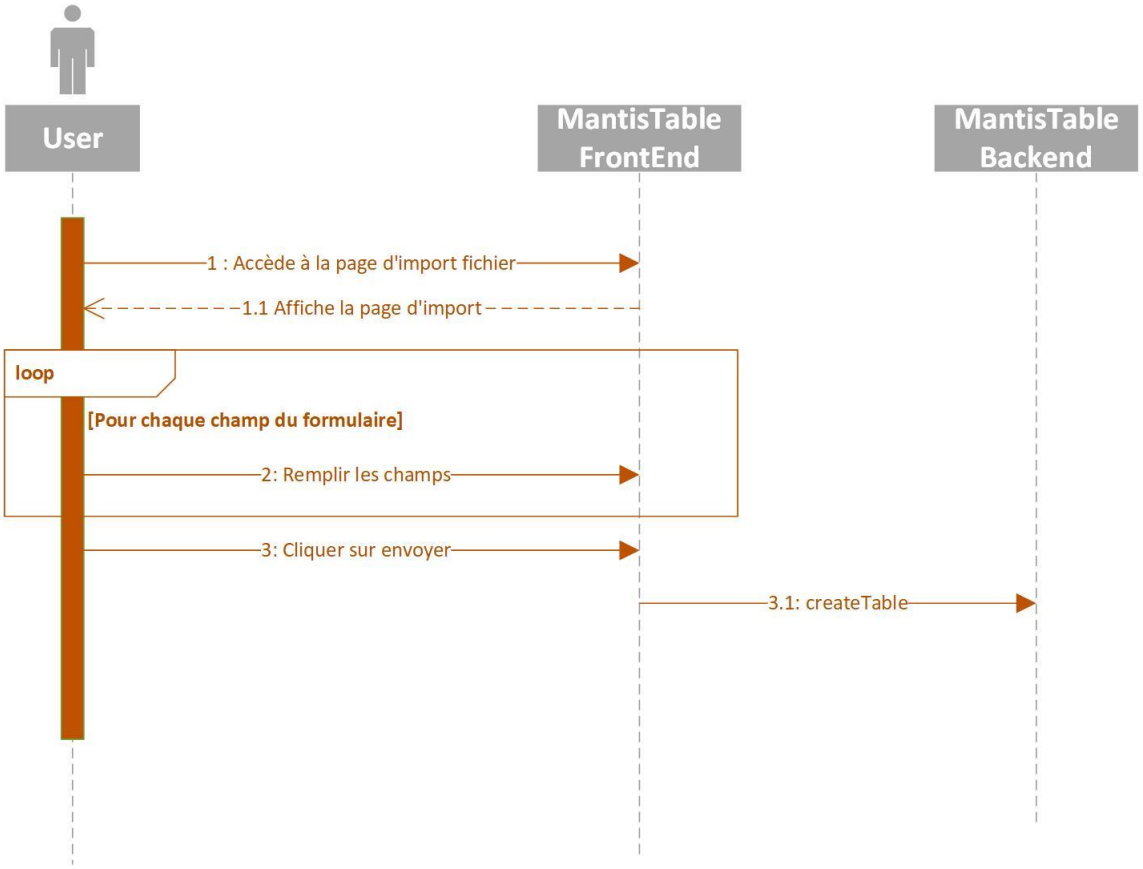


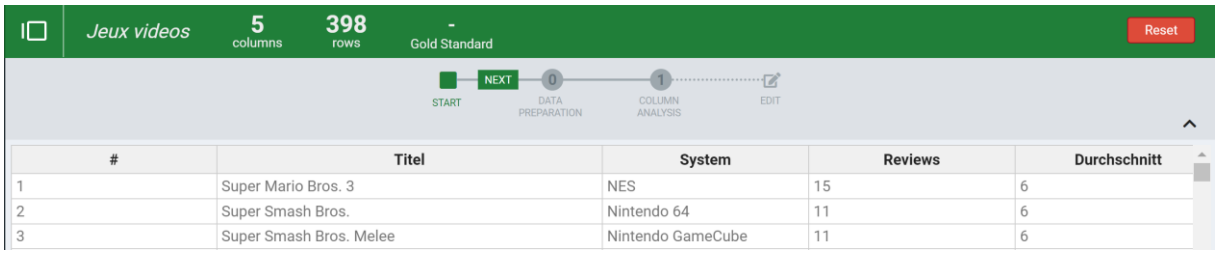
Figure 27 Diagramme de séquence d'import d'une table dans MantisTable

6.3.2 Traitement de la table

Nous avons réduit les étapes de traitement aux deux seules étapes qui nous sont nécessaires : la préparation de données et la détection de la colonne-sujet.

Préparation de données

Une fois la table importée, il est possible de la visualiser dans l'interface de l'outil. On récupère le nom de la table définit précédemment dans le formulaire et les totaux en haut de la page, c'est-à-dire le nombre de lignes et le nombre de colonnes de la table comme on peut le voir dans la Figure 28 Visualisation de la table brute importée dans MantisTableFigure 28.

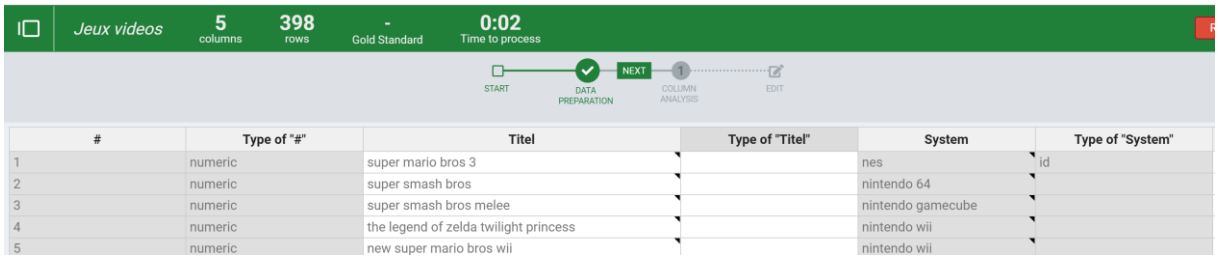


#	Titel	System	Reviews	Durchschnitt
1	Super Mario Bros. 3	NES	15	6
2	Super Smash Bros.	Nintendo 64	11	6
3	Super Smash Bros. Melee	Nintendo GameCube	11	6

Figure 28 Visualisation de la table brute importée dans MantisTable

En cliquant sur le bouton « NEXT », cela permet de lancer la première étape de préparation des données, c'est-à-dire transformer les textes en minuscule, résoudre les acronymes/abréviations, etc. grâce à l'utilisation d'expressions régulières.

Nous avons décidé de garder cette étape car elle permet plus de précision dans la détection de la colonne-sujet.



#	Type of "#"	Titel	Type of "Titel"	System	Type of "System"
1	numeric	super mario bros 3		nes	id
2	numeric	super smash bros		nintendo 64	
3	numeric	super smash bros melee		nintendo gamecube	
4	numeric	the legend of zelda twilight princess		nintendo wii	
5	numeric	new super mario bros wii		nintendo wii	

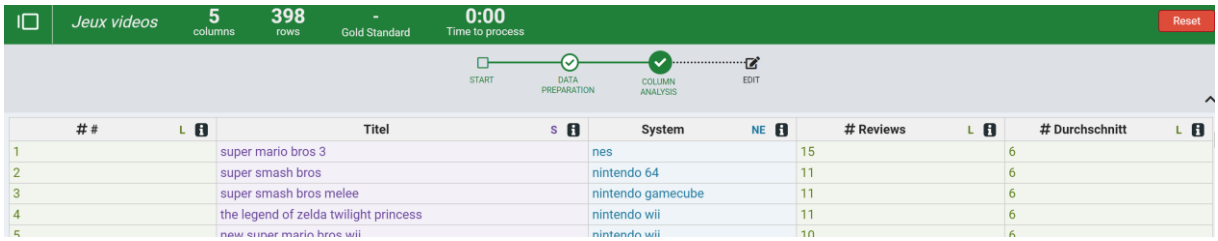
Figure 29 Visualisation de la table avec les types de cellules importée dans MantisTable

Les données sont donc nettoyées et une première annotation est disponible, il s'agit du typage des cellules, cela permet de savoir de quel type est la cellule (numérique, texte, id, etc.).

En cliquant à nouveau sur « NEXT », nous lançons l'étape suivante, c'est-à-dire la détection de la colonne-sujet.

Détection de la colonne-sujet

Une fois la détection de la colonne-sujet terminée, nous retrouvons notre table avec les annotations sur les différentes colonnes (NE-Column, S-Column, Literal).



# # L	Titel S	System NE	# Reviews L	# Durchschnitt L
1	super mario bros 3	nes	15	6
2	super smash bros	nintendo 64	11	6
3	super smash bros melee	nintendo gamecube	11	6
4	the legend of zelda twilight princess	nintendo wii	11	6
5	new super mario bros wii	nintendo wii	10	6

Figure 30 Visualisation de la table, avec ses annotations de colonne (littérale, sujet ou entité), importée dans MantisTable

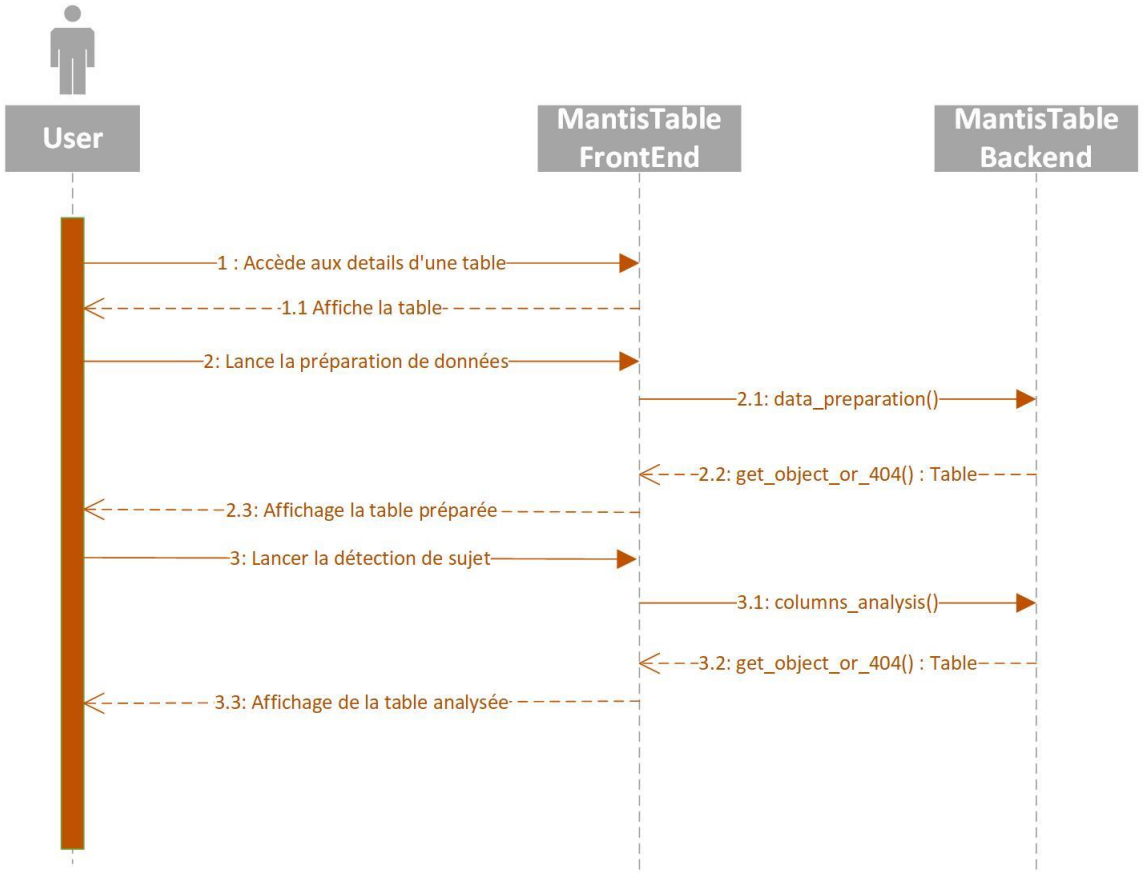


Figure 31 Diagramme de séquence de la préparation et de la détection de la colonne sujet dans MantisTable

6.3.3 Export de la table

Une fois l'étape de détection de sujet effectuée, nous pouvons revenir dans notre liste de tables. On s'aperçoit que dans la colonne « Download » (Figure 32) nous avons maintenant un bouton « S-Sorted CSV ». Il s'agit d'un bouton permettant l'export de la table directement en CSV avec la colonne-sujet en première colonne du fichier afin de pouvoir l'utiliser directement dans l'outil MTab.

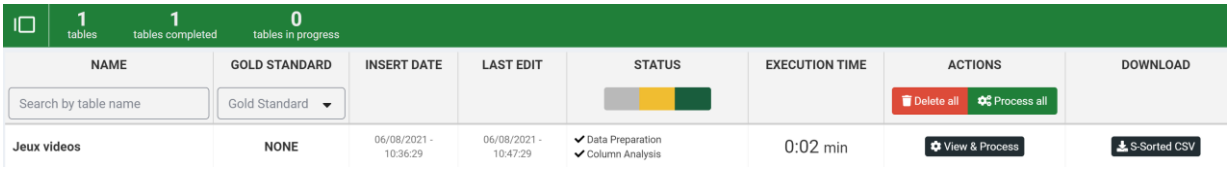


Figure 32 Liste des fichiers traités dans Matis Table avec le bouton S-Sorted CSV

Les signatures des deux méthodes d'export CSV que nous avons implémentées sont les suivantes :

- Download

```
def download_raw(request, table_id): HttpResponse
```

- S-Sorted Download

```
def download_csv(request, table_id): HttpResponse
```

6.4 Etape 2 : Utilisation de l'outil MTab pour récupérer les tableaux annotés

Notre algorithme est un algorithme Python automatisé qui se base sur la bibliothèque *Selenium*. Il va chercher directement le fichier CSV préparé par MantisTable dans un dossier spécifique. Une fois celui-ci récupéré, il accède directement à la page Web du site MTab pour y insérer les données et lancer le processus d'annotation. Une fois le processus terminé, nous récupérons notre table annotée sauf dans le cas où le temps d'attente maximal définit dans l'algorithme est atteint. Il n'est pas nécessaire ici de fournir une interface graphique car tout peut se faire en arrière-plan sans intervention nécessaire d'un utilisateur externe.

Méthode

Pour facilement illustrer l'étape d'annotation et d'extraction du tableau annoté, un diagramme de séquence a été créé (Figure 33). Ce dernier reprend l'ensemble des processus lancés (dans l'ordre chronologique) par le fichier « MtabExtractTable.py ».

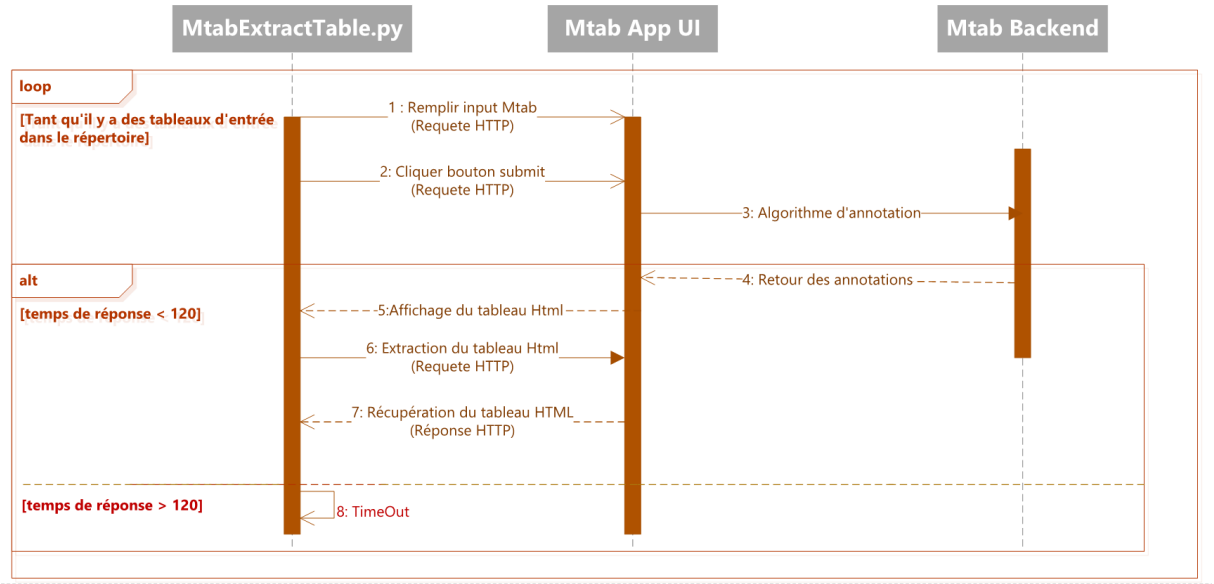


Figure 33 : Diagramme de Séquence MtabExtractTable.py

a. Remplir input MTab et envoyer (1 et 2).

Les processus « 1 : Remplir Input MTab » et « 2 : Cliquer bouton submit » se retrouvent dans la méthode ___interactePage du fichier « MtabExtractTable.py ».

Ils vont permettre d'insérer les tableaux d'entrée CSV qui ont été créé dans la première étape MantisTable.

b. Algorithme d'annotation (3).

Une fois l'input de MTab rempli et l'envoi de celui-ci, MTab App va lancer l'algorithme d'annotation.

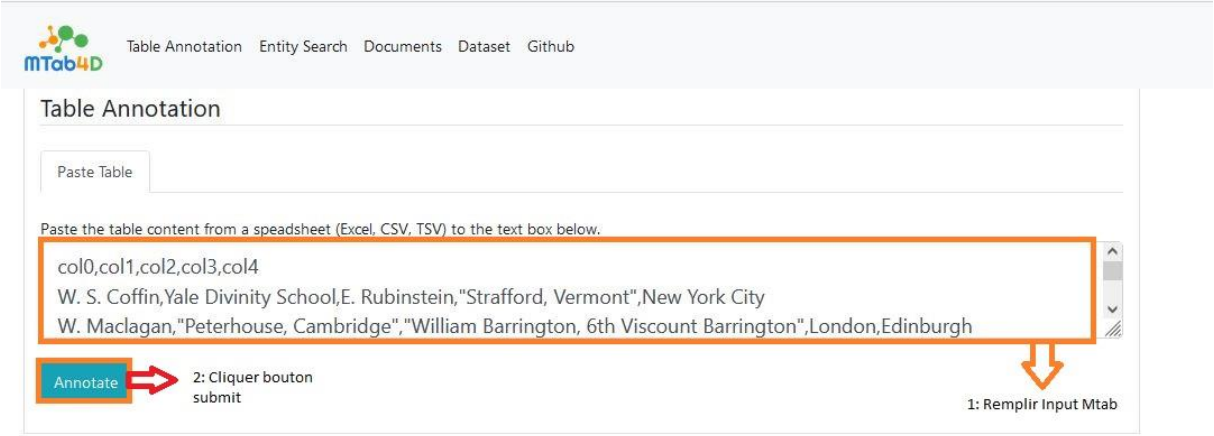


Figure 34 : Input & Buton submit MTab App

MTab va générer les entités candidates, établir les relations entre celles-ci et les colonnes/cellules de notre tableau. Pour cet outil, il y a deux types de relations possibles entité-entité ou entité-non entité.

Pour comprendre l'algorithme d'annotation veuillez-vous référer à la section outil MTab 2019.

c. Retour des annotations et affichage du tableau HTML (4 et 5)

Pour nos tests, nous avons décidé de paramétrer un timeout de deux minutes à l'outil pour annoter les tableaux d'entrée. S'il y a bien un retour avant les deux minutes, alors MTab va afficher un tableau contenant l'ensemble d'annotations.

d. Extraction et Récupération du tableau HTML (6 et 7)

Une fois que ce tableau est présent, l'algorithme va utiliser la bibliothèque *Selenium* pour retrouver et extraire le tableau HTML annoté de MTab. Le but de l'extraction du tableau HTML est de récupérer l'ensemble des URI du tableau.

S'il n'y a pas de réponse dans les deux minutes, alors l'algorithme passe au tableau d'entrée suivant. Ce temps de deux minutes ayant été arbitrairement choisi pour permettre de faire des tests rapides, l'utilisateur peut le changer en fonction de son besoin.

Figure 35 : Tableau d'annotations MTab App

Sortie

En sortie de notre algorithme, nous récupérons plusieurs tableaux (un tableau par entrée). Ce tableau est constitué d'un ensemble d'URIs.
Soit i = lignes, j = colonnes, T = Table initiale, U = table d'URI, nous avons que la cellule $T_{i,j}$ est représentée par l'annotation $U_{i,j}$.

Un exemple avec un tableau initial transformé en un tableau d'URI est respectivement représenté par la Figure 36 : Tableau initial et la Figure 37 : Tableau d'URI.

```
col0,col1,col2,col3
Giovanni Francesco Guidi di Bagno,Rome,C. family,1641-07-24
G. Doria,Palermo,G. A. Doria,1642-10-19
G. Murray,Chester Square,L. G. Murray,1860-02-16
Geoffrey,Normandy,Henry II of England,1212-12-12
Ferdinand III of Castile,Crown of Castile,Alfonso IX of Le¢n,1252-05-30
Erik Benzelius the younger,Linkoping,Erik Benzelius the elder,1743-09-23
E. t. Confessor,London,t. Unready,1066-01-05
E. W. Grinfield,Brighton,T. Grinfield,1864-07-09
E. F. Wilson,Salt Spring Island,D. Wilson,1915-05-11
D. F. Hudson,England,Father,2003-06-05
```

Figure 36 : Tableau initial

Ce tableau est ensuite passé dans l'application et Mtab et l'extraction HTML va rendre un tableau d'URI (Figure 37).

	Core Attribute	http://dbpedia.org/ontology/deathPlace	http://dbpedia.org/ontology/parent	http://dbpedia.org/ontology/deathDate
0	http://dbpedia.org/resource/Giovanni_Francesco_Guidi_di_Bagno	http://dbpedia.org/resource/Rome	http://dbpedia.org/resource/Colonna_family	http://dbpedia.org/resource/1641
1	http://dbpedia.org/resource/Giovanni_Doria	http://dbpedia.org/resource/Palermo	http://dbpedia.org/resource/Giovanni_Andrea_Doria	http://dbpedia.org/resource/1642
2	http://dbpedia.org/resource/George_Murray_(bishop_of_Rochester)	http://dbpedia.org/resource/Chester_Square	http://dbpedia.org/resource/Lord_George_Murray_(bishop)	http://dbpedia.org/resource/1860
3	http://dbpedia.org/resource/Geoffrey_(archbishop_of_York)	http://dbpedia.org/resource/Normandy	http://dbpedia.org/resource/Henry_II_of_England	http://dbpedia.org/resource/12-12-12
4	http://dbpedia.org/resource/Ferdinand_III_of_Castile	http://dbpedia.org/resource/Crown_of_Castile	http://dbpedia.org/resource/Alfonso_IX_of_Le%C3%B3n	http://dbpedia.org/resource/1252
5	http://dbpedia.org/resource/Erik_Benzelius_the_younger	http://dbpedia.org/resource/Link%C3%B6ping	http://dbpedia.org/resource/Erik_Benzelius_the_Elder	http://dbpedia.org/resource/1743
6	http://dbpedia.org/resource/Edward_the_Confessor	http://dbpedia.org/resource/London	http://dbpedia.org/resource/%C3%86thelred_the_Unready	http://dbpedia.org/resource/1066
7	http://dbpedia.org/resource/Edward_William_Grinfield	http://dbpedia.org/resource/Brighton	http://dbpedia.org/resource/Thomas_Grinfield	http://dbpedia.org/resource/1864
8	http://dbpedia.org/resource/Edward_Francis_Wilson	http://dbpedia.org/resource/Saltspring_Island	http://dbpedia.org/resource/Daniel_Wilson_(bishop)	http://dbpedia.org/resource/1915
9	http://dbpedia.org/resource/Donald_Foster_Hudson	http://dbpedia.org/resource/England	http://dbpedia.org/resource/Father	http://dbpedia.org/resource/2003

Figure 37 : Tableau d'URI

6.5 Etape 3 : Algorithme d'intégration d'ensemble de données

Cet algorithme prend en entrée deux ou plusieurs fichiers CSV correspondants aux annotations de deux ou plusieurs tableaux récupérés à la sortie de l'étape précédente (MTab). Le premier tableau en entrée est considéré comme le tableau-sujet, c'est-à-dire le tableau principal, et les autres tableaux sont considérés comme des tableaux

secondaires qui vont permettre de compléter ce premier tableau. En sortie, il renvoie un seul tableau qui correspond à la résultante des tableaux d'entrée.

Le traitement permet de comparer les tableaux d'entrée afin de rechercher l'inférence entre eux, cela peut être à la fois des unions et des jointures.

Cet algorithme est semi-automatique. C'est-à-dire qu'il a besoin de l'intervention de l'utilisateur pour fonctionner. Pour que l'utilisateur puisse intervenir, un ensemble de questions lui seront posées pour arriver au résultat le plus adéquat.

Tout d'abord, l'algorithme va laisser le choix à l'utilisateur d'ajouter des colonnes au tableau de résultats final. Les colonnes peuvent être ajoutées via recherche de similarités (Question1), par recherche de mot-clé (Question 2) et par recherche d'URI (Question 3). Initialement, ces colonnes ne possèdent pas encore de valeurs.

Finalement les valeurs seront ajoutées dans la phase d'insertion de données de l'algorithme.

6.5.1 Ajout de colonnes au tableau final par recherche de similarités (Question 1)

Cette question donne un choix à l'utilisateur qui permet à l'algorithme de savoir si les colonnes-sujet sont les mêmes entre les différents jeux de données. En fonction de la réponse de l'utilisateur, l'algorithme agira différemment.

Deux choix sont possibles pour cette question :

- Le premier choix permet à l'utilisateur de confirmer que les colonnes-sujet des deux ensembles de données sont les mêmes.
- Le second choix permet à l'utilisateur de confirmer que les colonnes-sujet sont différentes.

Pour orienter l'utilisateur sur la réponse à cette question, il faut reprendre l'ensemble des types sémantiques des colonnes-sujet des différents tableaux. Si les types de la colonne-sujet de la table-sujet correspond à 100% avec les types de la colonne-sujet d'une table secondaire, alors il est fort probable que les deux ensembles de données possèdent le même contexte. Par exemple, on pourrait avoir un premier ensemble de données qui possède dans sa colonne-sujet des présidents Américains et dans l'autre ensemble de données la colonne-sujet possède aussi des présidents Américains.

Il se peut aussi que les colonnes-sujet ne correspondent pas ou alors qu'elles ont quelques points en commun. L'algorithme va ressortir les éléments en commun et donner un conseil concernant les deux choix possibles. L'utilisateur devra donc taper 1 pour le premier choix et taper 2 pour le deuxième choix.

Choix 1

En tapant 1, l'utilisateur confirme donc que les colonnes-sujet sont les mêmes. L'algorithme va faire une union distincte de l'ensemble des tableaux. Il va d'abord rajouter les lignes des tableaux secondaires dans le tableau-sujet et ensuite, de la même façon, il va rajouter les différentes colonnes de ceux-ci.

Pour cela nous utilisons la méthode « append »[24] de la bibliothèque Panda appelée sur les différents tableaux. L'appel de cette méthode se fait de la manière suivante :

```
df = df1.append(df2, ignore_index=True, sort=False)
```

Figure 38 : Appel de la fonction "append"

Elle va permettre de rajouter des lignes et les colonnes manquantes au tableau-sujet en ajoutant tous les tableaux secondaires.

Par exemple :

	Core Attribute	http://dbpedia.org/ontology/deathPlace	http://dbpedia.org/ontology/parent	http://dbpedia.org/ontology/deathDate
0	http://dbpedia.org/resource/Giovanni_Francesco_Guidi_di_Bagno	http://dbpedia.org/resource/Rome	http://dbpedia.org/resource/Colonna_family	http://dbpedia.org/resource/1641
1	http://dbpedia.org/resource/Giovanni_Doria	http://dbpedia.org/resource/Palermo	http://dbpedia.org/resource/Giovanni_Andrea_Doria	http://dbpedia.org/resource/1642
2	http://dbpedia.org/resource/George_Murray_(bishop_of_Rochester)	http://dbpedia.org/resource/Chester_Square	http://dbpedia.org/resource/Lord_George_Murray_(bishop)	http://dbpedia.org/resource/1860
3	http://dbpedia.org/resource/Geoffrey_(archbishop_of_York)	http://dbpedia.org/resource/Normandy	http://dbpedia.org/resource/Henry_II_of_England	http://dbpedia.org/resource/12-12-12
4	http://dbpedia.org/resource/Ferdinand_III_of_Castile	http://dbpedia.org/resource/Crown_of_Castile	http://dbpedia.org/resource/Alfonso_IX_of_Le%C3%B3n	http://dbpedia.org/resource/1252
5	http://dbpedia.org/resource/Erik_Benzelius_the_younger	http://dbpedia.org/resource/Link%C3%B6ping	http://dbpedia.org/resource/Erik_Benzelius_the_Elder	http://dbpedia.org/resource/1743
6	http://dbpedia.org/resource/Edward_the_Confessor	http://dbpedia.org/resource/London	http://dbpedia.org/resource/%C3%86thelred_the_Unready	http://dbpedia.org/resource/1066
7	http://dbpedia.org/resource/Edward_William_Grinfield	http://dbpedia.org/resource/Brighton	http://dbpedia.org/resource/Thomas_Grinfield	http://dbpedia.org/resource/1864
8	http://dbpedia.org/resource/Edward_Francis_Wilson	http://dbpedia.org/resource/Saltspring_Island	http://dbpedia.org/resource/Daniel_Wilson_(bishop)	http://dbpedia.org/resource/1915
9	http://dbpedia.org/resource/Donald_Foster_Hudson	http://dbpedia.org/resource/England	http://dbpedia.org/resource/Father	http://dbpedia.org/resource/2003

	Core Attribute	http://dbpedia.org/ontology/birthDate	http://dbpedia.org/ontology/birthPlace
0	http://dbpedia.org/resource/Dick_Sheppard_(priest)	http://dbpedia.org/resource/1880	http://dbpedia.org/resource/Windsor
1	http://dbpedia.org/resource/Claus_Westermann	http://dbpedia.org/resource/1909	http://dbpedia.org/resource/Berlin
2	http://dbpedia.org/resource/Charles_Januarius_Acton	http://dbpedia.org/resource/1803	http://dbpedia.org/resource/Naples
3	http://dbpedia.org/resource/Carlo_Barberini	http://dbpedia.org/resource/1630	http://dbpedia.org/resource/Rome
4	http://dbpedia.org/resource/Cardinal_de_Bouillon	http://dbpedia.org/resource/1643	http://dbpedia.org/resource/France
5	http://dbpedia.org/resource/Camillo_Francesco_Maria_Pamphili	http://dbpedia.org/resource/1622	http://dbpedia.org/resource/Naples
6	http://dbpedia.org/resource/Benjamin_Hoadly	http://dbpedia.org/resource/1676	http://dbpedia.org/resource/Kent

Figure 39 : Ensemble de donnée Choix 1

Dans l'exemple de la Figure 40, les colonnes Core Attribut (colonne-sujet de la Figure 39) possèdent des types en commun.

```
['http://dbpedia.org/ontology/Cleric', 'http://dbpedia.org/ontology/Person']
['http://dbpedia.org/ontology/Cleric', 'http://dbpedia.org/ontology/ChristianBishop', 'http://dbpedia.org/ontology/Person']
Voici les éléments en communs :{'http://dbpedia.org/ontology/Person', 'http://dbpedia.org/ontology/Cleric'}
Tous les types de la liste sujet se retrouvent dans la liste cible. Nous suggérons donc de choisir le premier choix d'intégration de dataset.
Pour choisir le premier choix taper 1 sinon taper 2 :1
```

Figure 40 : Liste des types correspondant aux colonnes sujets du choix 1

[24] https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.append.html

Une fois que c'est fait, nous obtenons un jeu de données final mais certaines colonnes ont des valeurs « nan » puisque la méthode append permet juste de joindre deux jeux de données sans remplir les données manquantes.

6.5.1.1 Choix 2

Ici, l'utilisateur confirme donc que les colonnes-sujet sont différentes. Le premier tableau sera vu comme le tableau-sujet avec lequel l'algorithme va rechercher tous les liens possibles avec les tableaux secondaires et faire une liste de propositions.

Dans la Figure 41, on retrouve un exemple de tableaux n'ayant pas des types correspondants. On voit clairement que le premier tableau donne un type sémantique « Cleric » et le second donne un type « OfficeHolder ».

	Core Attribute	http://dbpedia.org/ontology/deathPlace	http://dbpedia.org/ontology/parent	http://dbpedia.org/ontology/deathDate
0	http://dbpedia.org/resource/Giovanni_Francesco_Guidi_di_Bagno	http://dbpedia.org/resource/Rome	http://dbpedia.org/resource/Colonna_family	http://dbpedia.org/resource/1641
1	http://dbpedia.org/resource/Giovanni_Doria	http://dbpedia.org/resource/Palermo	http://dbpedia.org/resource/Giovanni_Andrea_Doria	http://dbpedia.org/resource/1642
2	http://dbpedia.org/resource/George_Murray_(bishop_of_Rochester)	http://dbpedia.org/resource/Chester_Square	http://dbpedia.org/resource/Lord_George_Murray_(bishop)	http://dbpedia.org/resource/1860
3	http://dbpedia.org/resource/Geoffrey_(archbishop_of_York)	http://dbpedia.org/resource/Normandy	http://dbpedia.org/resource/Henry_II_of_England	http://dbpedia.org/resource/12-12-12
4	http://dbpedia.org/resource/Ferdinand_III_of_Castile	http://dbpedia.org/resource/Crown_of_Castile	http://dbpedia.org/resource/Alfonso_IX_of_Le%C3%83n	http://dbpedia.org/resource/1252
5	http://dbpedia.org/resource/Erik_Benzelius_the_younger	http://dbpedia.org/resource/Link%C3%B6ping	http://dbpedia.org/resource/Erik_Benzelius_the_Elder	http://dbpedia.org/resource/1743
6	http://dbpedia.org/resource/Edward_the_Confessor	http://dbpedia.org/resource/London	http://dbpedia.org/resource/%C3%86thelred_the_Unready	http://dbpedia.org/resource/1066
7	http://dbpedia.org/resource/Edward_William_Grinfield	http://dbpedia.org/resource/Brighton	http://dbpedia.org/resource/Thomas_Grinfield	http://dbpedia.org/resource/1864
8	http://dbpedia.org/resource/Edward_Francis_Wilson	http://dbpedia.org/resource/Saltspring_Island	http://dbpedia.org/resource/Daniel_Wilson_(bishop)	http://dbpedia.org/resource/1915
9	http://dbpedia.org/resource/Donald_Foster_Hudson	http://dbpedia.org/resource/England	http://dbpedia.org/resource/Father	http://dbpedia.org/resource/2003

	Core Attribute	http://dbpedia.org/ontology/birthDate	http://dbpedia.org/ontology/birthPlace	http://dbpedia.org/ontology/party
0	http://dbpedia.org/resource/Barack_Obama	http://dbpedia.org/resource/1961	http://dbpedia.org/resource/Honolulu	http://dbpedia.org/resource/Democratic_Party_(United_States)
1	http://dbpedia.org/resource/Benjamin_Harrison	http://dbpedia.org/resource/1833	http://dbpedia.org/resource/North_Bend,_Ohio	http://dbpedia.org/resource/Republican_Party_(United_States)
2	http://dbpedia.org/resource/Calvin_Coolidge	http://dbpedia.org/resource/1872	http://dbpedia.org/resource/Plymouth_Notch,_Vermont	http://dbpedia.org/resource/Republican_Party_(United_States)
3	http://dbpedia.org/resource/Harry_S._Truman	http://dbpedia.org/resource/1884	http://dbpedia.org/resource/Lamar	http://dbpedia.org/resource/Missouri
4	http://dbpedia.org/resource/Herbert_Hoover	http://dbpedia.org/resource/1874		http://dbpedia.org/resource/Iowa
5	http://dbpedia.org/resource/Lyndon_B._Johnson	http://dbpedia.org/resource/1908	http://dbpedia.org/resource/Stonewall	http://dbpedia.org/resource/Texas

Figure 41 : Exemple de jeux de données pour le choix 2

A partir de ces tableaux, l'algorithme va nous montrer les points communs entre les colonnes-sujet comme on le voit à la Figure 42.

```
['http://dbpedia.org/ontology/Cleric', 'http://dbpedia.org/ontology/Person']
['http://dbpedia.org/ontology/OfficeHolder', 'http://dbpedia.org/ontology/Person']
Voici les éléments en communs :{'http://dbpedia.org/ontology/Person'}
Tous les types de la liste sujet ne se retrouvent pas dans la liste cible. Nous suggérons donc de choisir le deuxième choix d'intégration de dataset.
Pour choisir le premier choix taper 1 sinon taper 2 :2
```

Figure 42 : Liste des types correspondant aux colonnes sujets du choix 2

Dans ce cas-ci, on voit que tous les types sémantiques ne sont pas retrouvés, le seul point commun mis en évidence est le type sémantique « Person ». En tapant 2, les lignes du second jeu de données ne seront pas ajoutées comme dans le choix 1, l'algorithme applique des requêtes SPARQL en prenant la colonne-sujet du tableau-sujet comme le sujet de notre requête afin de rechercher des points communs entre nos deux tableaux.

Recherche de similarités par requêtes SPARQL

D'abord, l'algorithme exécute un **premier type** de requête (cellule-sujet vers colonne) (Figure 43). Il s'agit d'une recherche de similarités entre les cellules de la colonne sujet du tableau-sujet (première colonne du tableau-sujet) et les ontologies des tableaux secondaires (colonne des autres tableaux).

On va créer une liste de propositions pour laquelle, à chaque fois que la requête renvoie une ontologie, on l'ajoute dans cette liste.

```
PREFIX dbr:  <http://dbpedia.org/resource/>
 select ?object where {
 { <http://dbpedia.org/resource/Giovanni_Francesco_Guidi_di_Bagno> <http://dbpedia.org/ontology/birthDate> ?object }
}
```

Figure 43 : Requête SPARQL pour retrouver des ontologies liées à la colonne sujet

Ensuite, l'algorithme exécute un **second type** de requête (cellule-sujet vers cellule) (Figure 44). Il s'agit d'une recherche de similarités entre les cellules de la colonne-sujet du tableau-sujet (première colonne du tableau-sujet) et les cellules des tableaux secondaires (cellules des autres tableaux) uniquement pour les colonnes qui ne se retrouvent pas dans la liste de proposition précédemment créée.

```
PREFIX dbr:  <http://dbpedia.org/resource/>
 select distinct ?predicate where {
 { <http://dbpedia.org/resource/Giovanni_Francesco_Guidi_di_Bagno> ?predicate <http://dbpedia.org/resource/Democratic_Party_(United_States)>}
}
```

Figure 44 : Requête SPARQL pour retrouver des ontologies liées à la colonne sujet

De même que pour le premier type de requête, si la requête renvoie une ontologie, on l'ajoute dans la liste de propositions.

Finalement, ces différentes requêtes SPARQL vont ressortir une liste de propositions. A partir de ces propositions, l'utilisateur choisit celles qu'il veut ajouter à la table-sujet.

```
Propositions:
0 http://dbpedia.org/ontology/birthDate
1 http://dbpedia.org/ontology/birthPlace
Sélectionner les propositions une par une en écrivant leurs numéros (-1 pour sortir de la question):
```

Figure 45 : Liste de proposition choix 2

6.5.2 Ajout de colonnes au tableau final par recherche de mot clé (Question 2)

La question numéro 2 va permettre à l'utilisateur d'ajouter des colonnes vides au tableau de résultats final. Pour cela, il lui suffit d'écrire un mot pour y retrouver son ontologie.

```
Question 2
Si vous avez une autre colonne à ajouter ecrivez le. Exemple : birthPlace. Si vous n'en avez plus, écrivez -1:birthplace
0 http://dbpedia.org/ontology/birthPlace
```

Figure 46 : Question 2 avec la réponse « birthplace » donnée par l'utilisateur

Pour un même mot, il est possible qu'il y ait plusieurs ontologies qui ressortent, il suffira à l'utilisateur de choisir celles qui sont intéressantes en y écrivant son numéro en commençant par 0 (Figure 46).

6.5.3 Ajout de colonnes au tableau final par URI (Question 3)

Si la première et la deuxième question n'ont pas permis de retrouver les colonnes souhaitées par l'utilisateur, il peut alors entrer lui-même le lien URI qui l'intéresse et celle-ci sera ajoutée en tant que colonne du tableau de résultats final.

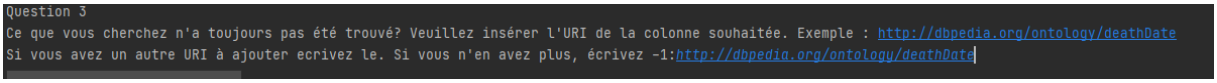


Figure 47 : Question 3 avec l'ajout d'un URI http://dbpedia.org/ontology/deathDate

Pour l'exemple de la Figure 47, cela va permettre d'ajouter l'ontologie « deathDate » si celle-ci n'est pas déjà dans le tableau final, c'est-à-dire ajouter une colonne vide « deathDate » dans notre tableau de résultats final.

6.5.4 Insertion des données

Une fois que tous les choix ont été effectués, on obtient un tableau de résultats initial qui comprend les colonnes sélectionnées dans les étapes précédentes ainsi que leurs valeurs de cellules respectives (Figure 48).

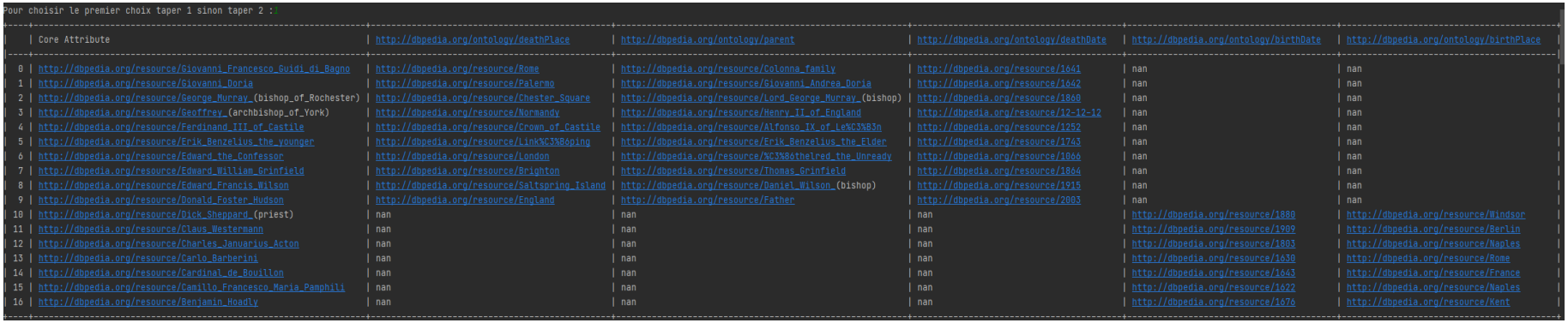
Ce tableau correspond à l'ajout des ensembles de données (lignes + colonnes). Pour remplir les valeurs non définies « nan », l'algorithme va exécuter des requêtes SPARQL (Figure 49).

Finalement, nous obtenons notre tableau de résultats final (Figure 50).

Pour l'exemple du choix 2, comme les colonnes-sujet ne sont pas les mêmes on peut voir qu'aucune ligne n'est rajoutée (pas d'union) dans notre tableau final (Figure 51) et que pour les requêtes SPARQL exécutées (Figure 52) nous aurons le résultat présenté à la Figure 53 .

Des valeurs « nan » sont encore présentes. C'est parce qu'il n'y a aucune information concernant cette ontologie pour cette personne dans DBpedia.

Pour l'explication de cette partie, nous avons présentés des résultats avec deux tableaux (un tableau-sujet et un tableau secondaire). Notre algorithme permet de travailler n'importe quel nombre de tableaux secondaires.



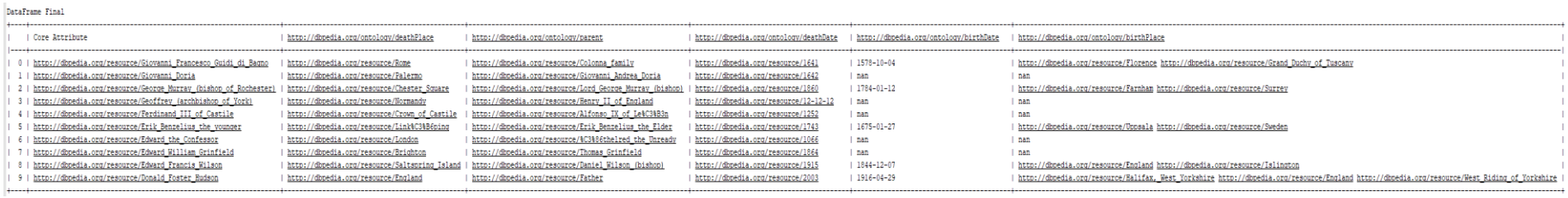
Pour choisir le premier choix taper 1 sinon taper 2 :

	Core Attribute	http://dbpedia.org/ontology/deathPlace	http://dbpedia.org/ontology/parent	http://dbpedia.org/ontology/deathDate	http://dbpedia.org/ontology/birthDate	http://dbpedia.org/ontology/birthPlace
0	http://dbpedia.org/resource/Giovanni_Francesco_Guidi_di_Bagno	http://dbpedia.org/resource/Rome	http://dbpedia.org/resource/Colonna_family	http://dbpedia.org/resource/1641	nan	nan
1	http://dbpedia.org/resource/Giovanni_Doria	http://dbpedia.org/resource/Palermo	http://dbpedia.org/resource/Giovanni_Andrea_Doria	http://dbpedia.org/resource/1642	nan	nan
2	http://dbpedia.org/resource/George_Murray_(bishop_of_Rochester)	http://dbpedia.org/resource/Chester_Square	http://dbpedia.org/resource/Lord_George_Murray_(bishop)	http://dbpedia.org/resource/1860	nan	nan
3	http://dbpedia.org/resource/Geoffrey_(archbishop_of_York)	http://dbpedia.org/resource/Normandy	http://dbpedia.org/resource/Henry_II_of_England	http://dbpedia.org/resource/12-12-12	nan	nan
4	http://dbpedia.org/resource/Ferdinand_III_of_Castile	http://dbpedia.org/resource/Crown_of_Castile	http://dbpedia.org/resource/Alfonso_IX_of_Le%C3%B3n	http://dbpedia.org/resource/1252	nan	nan
5	http://dbpedia.org/resource/Erik_Benzelius_the_younger	http://dbpedia.org/resource/Link%C3%B6ping	http://dbpedia.org/resource/Erik_Benzelius_the_Elder	http://dbpedia.org/resource/1743	nan	nan
6	http://dbpedia.org/resource/Edward_the_Confessor	http://dbpedia.org/resource/London	http://dbpedia.org/resource/%C3%86thelred_the_Unready	http://dbpedia.org/resource/1066	nan	nan
7	http://dbpedia.org/resource/Edward_William_Grinfield	http://dbpedia.org/resource/Brighton	http://dbpedia.org/resource/Thomas_Grinfield	http://dbpedia.org/resource/1864	nan	nan
8	http://dbpedia.org/resource/Edward_Francis_Wilson	http://dbpedia.org/resource/Saltspring_Island	http://dbpedia.org/resource/Daniel_Wilson_(bishop)	http://dbpedia.org/resource/1915	nan	nan
9	http://dbpedia.org/resource/Donald_Foster_Hudson	http://dbpedia.org/resource/England	http://dbpedia.org/resource/Father	http://dbpedia.org/resource/2003	nan	nan
10	http://dbpedia.org/resource/Dick_Sheppard_(priest)	nan	nan	nan	http://dbpedia.org/resource/1880	http://dbpedia.org/resource/Windsor
11	http://dbpedia.org/resource/Claus_Westermann	nan	nan	nan	http://dbpedia.org/resource/1909	http://dbpedia.org/resource/Berlin
12	http://dbpedia.org/resource/Charles_Januarius_Acton	nan	nan	nan	http://dbpedia.org/resource/1803	http://dbpedia.org/resource/Naples
13	http://dbpedia.org/resource/Carlo_Barberini	nan	nan	nan	http://dbpedia.org/resource/1630	http://dbpedia.org/resource/Rome
14	http://dbpedia.org/resource/Cardinal_de_Bouillon	nan	nan	nan	http://dbpedia.org/resource/1643	http://dbpedia.org/resource/France
15	http://dbpedia.org/resource/Camillo_Francesco_Maria_Pamphili	nan	nan	nan	http://dbpedia.org/resource/1622	http://dbpedia.org/resource/Naples
16	http://dbpedia.org/resource/Benjamin_Hoadly	nan	nan	nan	http://dbpedia.org/resource/1676	http://dbpedia.org/resource/Kent

Figure 48 : Ensemble de données final avant insertion choix 1

```
PREFIX dbr:  <http://dbpedia.org/resource/>
 select ?object where {
 { <http://dbpedia.org/resource/Giovanni_Francesco_Guidi_di_Bagno> <http://dbpedia.org/ontology/birthDate> ?object }
}
```

Figure 49 : Requête SPARQL d'insertion choix 1



DataFrame Final

	Core Attribute	http://dbpedia.org/ontology/deathPlace	http://dbpedia.org/ontology/parent	http://dbpedia.org/ontology/deathDate	http://dbpedia.org/ontology/birthDate	http://dbpedia.org/ontology/birthPlace
0	http://dbpedia.org/resource/Giovanni_Francesco_Guidi_di_Bagno	http://dbpedia.org/resource/Rome	http://dbpedia.org/resource/Colonna_family	http://dbpedia.org/resource/1641	1578-10-04	http://dbpedia.org/resource/Florence http://dbpedia.org/resource/Grand_Duchy_of_Tuscany
1	http://dbpedia.org/resource/Giovanni_Doria	http://dbpedia.org/resource/Palermo	http://dbpedia.org/resource/Giovanni_Andrea_Doria	http://dbpedia.org/resource/1642	nan	nan
2	http://dbpedia.org/resource/George_Murray_(bishop_of_Rochester)	http://dbpedia.org/resource/Chester_Square	http://dbpedia.org/resource/Lord_George_Murray_(bishop)	http://dbpedia.org/resource/1860	1784-01-12	http://dbpedia.org/resource/Farnham http://dbpedia.org/resource/Surrey
3	http://dbpedia.org/resource/Geoffrey_(archbishop_of_York)	http://dbpedia.org/resource/Normandy	http://dbpedia.org/resource/Henry_II_of_England	http://dbpedia.org/resource/12-12-12	nan	nan
4	http://dbpedia.org/resource/Ferdinand_III_of_Castile	http://dbpedia.org/resource/Crown_of_Castile	http://dbpedia.org/resource/Alfonso_IX_of_Le%C3%B3n	http://dbpedia.org/resource/1252	nan	nan
5	http://dbpedia.org/resource/Erik_Benzelius_the_younger	http://dbpedia.org/resource/Link%C3%B6ping	http://dbpedia.org/resource/Erik_Benzelius_the_Elder	http://dbpedia.org/resource/1743	1675-01-27	http://dbpedia.org/resource/Uppsala http://dbpedia.org/resource/Sweden
6	http://dbpedia.org/resource/Edward_the_Confessor	http://dbpedia.org/resource/London	http://dbpedia.org/resource/%C3%86thelred_the_Unready	http://dbpedia.org/resource/1066	nan	nan
7	http://dbpedia.org/resource/Edward_William_Grinfield	http://dbpedia.org/resource/Brighton	http://dbpedia.org/resource/Thomas_Grinfield	http://dbpedia.org/resource/1864	nan	nan
8	http://dbpedia.org/resource/Edward_Francis_Wilson	http://dbpedia.org/resource/Saltspring_Island	http://dbpedia.org/resource/Daniel_Wilson_(bishop)	http://dbpedia.org/resource/1915	1844-12-07	http://dbpedia.org/resource/England http://dbpedia.org/resource/Islington
9	http://dbpedia.org/resource/Donald_Foster_Hudson	http://dbpedia.org/resource/England	http://dbpedia.org/resource/Father	http://dbpedia.org/resource/2003	1916-04-29	http://dbpedia.org/resource/Halifax,_West_Yorkshire http://dbpedia.org/resource/England http://dbpedia.org/resource/West_Riding_of_Yorkshire

Figure 50 : Ensemble de données final après insertion choix 1

	Core Attribute	http://dbpedia.org/ontology/deathPlace	http://dbpedia.org/ontology/parent	http://dbpedia.org/ontology/deathDate	http://dbpedia.org/ontology/birthDate	http://dbpedia.org/ontology/birthPlace
0	http://dbpedia.org/resource/Giovanni_Francesco_Guidi_di_Bagno	http://dbpedia.org/resource/Rome	http://dbpedia.org/resource/Colonna_family	http://dbpedia.org/resource/1641	nan	nan
1	http://dbpedia.org/resource/Giovanni_Doria	http://dbpedia.org/resource/Palermo	http://dbpedia.org/resource/Giovanni_Andrea_Doria	http://dbpedia.org/resource/1642	nan	nan
2	http://dbpedia.org/resource/George_Murray_(bishop_of_Rochester)	http://dbpedia.org/resource/Chester_Square	http://dbpedia.org/resource/Lord_George_Murray_(bishop)	http://dbpedia.org/resource/1860	nan	nan
3	http://dbpedia.org/resource/Geoffrey_(archbishop_of_York)	http://dbpedia.org/resource/Normandy	http://dbpedia.org/resource/Henry_II_of_England	http://dbpedia.org/resource/12-12-12	nan	nan
4	http://dbpedia.org/resource/Ferdinand_III_of_Castile	http://dbpedia.org/resource/Crown_of_Castile	http://dbpedia.org/resource/Alfonso_IX_of_Le%C3%B3n	http://dbpedia.org/resource/1252	nan	nan
5	http://dbpedia.org/resource/Erik_Benzelius_the_younger	http://dbpedia.org/resource/Link%C3%B6ping	http://dbpedia.org/resource/Erik_Benzelius_the_Elder	http://dbpedia.org/resource/1743	nan	nan
6	http://dbpedia.org/resource/Edward_the_Confessor	http://dbpedia.org/resource/London	http://dbpedia.org/resource/%C3%86thelred_the_Unready	http://dbpedia.org/resource/1066	nan	nan
7	http://dbpedia.org/resource/Edward_William_Grinfield	http://dbpedia.org/resource/Brighton	http://dbpedia.org/resource/Thomas_Grinfield	http://dbpedia.org/resource/1864	nan	nan
8	http://dbpedia.org/resource/Edward_Francis_Wilson	http://dbpedia.org/resource/Saltspring_Island	http://dbpedia.org/resource/Daniel_Wilson_(bishop)	http://dbpedia.org/resource/1915	nan	nan
9	http://dbpedia.org/resource/Donald_Foster_Hudson	http://dbpedia.org/resource/England	http://dbpedia.org/resource/Father	http://dbpedia.org/resource/2003	nan	nan

Figure 51 : Ensemble de données final avant insertion choix 2

```
PREFIX dbr:  <http://dbpedia.org/resource/>
 select ?object where {
 { <http://dbpedia.org/resource/Giovanni_Francesco_Guidi_di_Bagno> <http://dbpedia.org/ontology/birthDate> ?object }
}
```

Figure 52 : Requête SPARQL d'insertion choix 2

DataFrame Final

	Core Attribute	http://dbpedia.org/ontology/deathPlace	http://dbpedia.org/ontology/parent	http://dbpedia.org/ontology/deathDate	http://dbpedia.org/ontology/birthDate	http://dbpedia.org/ontology/birthPlace
0	http://dbpedia.org/resource/Giovanni_Francesco_Guidi_di_Bagno	http://dbpedia.org/resource/Rome	http://dbpedia.org/resource/Colonna_family	http://dbpedia.org/resource/1641	1578-10-04	http://dbpedia.org/resource/Florence http://dbpedia.org/resource/Grand_Duchy_of_Tuscany
1	http://dbpedia.org/resource/Giovanni_Doria	http://dbpedia.org/resource/Palermo	http://dbpedia.org/resource/Giovanni_Andrea_Doria	http://dbpedia.org/resource/1642	nan	nan
2	http://dbpedia.org/resource/George_Murray_(bishop_of_Rochester)	http://dbpedia.org/resource/Chester_Square	http://dbpedia.org/resource/Lord_George_Murray_(bishop)	http://dbpedia.org/resource/1860	1784-01-12	http://dbpedia.org/resource/Farnham http://dbpedia.org/resource/Surrey
3	http://dbpedia.org/resource/Geoffrey_(archbishop_of_York)	http://dbpedia.org/resource/Normandy	http://dbpedia.org/resource/Henry_II_of_England	http://dbpedia.org/resource/12-12-12	nan	nan
4	http://dbpedia.org/resource/Ferdinand_III_of_Castile	http://dbpedia.org/resource/Crown_of_Castile	http://dbpedia.org/resource/Alfonso_IX_of_Le%C3%B3n	http://dbpedia.org/resource/1252	nan	nan
5	http://dbpedia.org/resource/Erik_Benzelius_the_younger	http://dbpedia.org/resource/Link%C3%B6ping	http://dbpedia.org/resource/Erik_Benzelius_the_Elder	http://dbpedia.org/resource/1743	1675-01-27	http://dbpedia.org/resource/Uppsala http://dbpedia.org/resource/Sweden
6	http://dbpedia.org/resource/Edward_the_Confessor	http://dbpedia.org/resource/London	http://dbpedia.org/resource/%C3%86thelred_the_Unready	http://dbpedia.org/resource/1066	nan	nan
7	http://dbpedia.org/resource/Edward_William_Grinfield	http://dbpedia.org/resource/Brighton	http://dbpedia.org/resource/Thomas_Grinfield	http://dbpedia.org/resource/1864	nan	nan
8	http://dbpedia.org/resource/Edward_Francis_Wilson	http://dbpedia.org/resource/Saltspring_Island	http://dbpedia.org/resource/Daniel_Wilson_(bishop)	http://dbpedia.org/resource/1915	1844-12-07	http://dbpedia.org/resource/England http://dbpedia.org/resource/Islington
9	http://dbpedia.org/resource/Donald_Foster_Hudson	http://dbpedia.org/resource/England	http://dbpedia.org/resource/Father	http://dbpedia.org/resource/2003	1916-04-29	http://dbpedia.org/resource/Halifax,_West_Yorkshire http://dbpedia.org/resource/England

Figure 53 : Ensemble de données final après insertion choix 2

6.6 Etape 4 : Export du tableau final

Une fois que le tableau est rempli, celui-ci peut être exporté en format Excel (XLSX) ou bien en CSV. Ce fichier s'appelle File Name.xlsx et peut être trouvé à la racine du projet.

6.7 Conclusion

Notre outil permet d'identifier la colonne via la détection de colonne sujet de l'outil MantisTable 2019 ; ensuite via l'outil Mtab 2019, nous sommes capables de récupérer l'ensemble des URI qui nous intéresse. Une fois les URI récupérés, il nous suffit de poser des questions à l'utilisateur et d'ensuite lancer nos requêtes SPARQL pour intégrer différents jeux de données entre eux. Nous n'avons pas pu produire énormément de tests mais, pour les cas que nous avions prévus, les résultats ont été très bons.

7 Chapitre 7 : Discussion

L'annotation des tableaux consiste à comprendre la structure et la signification du contenu en tenant compte du tableau en lui-même mais également du contexte comme sa légende par exemple. Cependant, il n'est pas toujours évident de l'interpréter en raison d'un manque d'informations comme le titre des colonnes ou encore d'une ambiguïté entre celles-ci.

D'une part, nous présentons l'Etat de l'art des approches STI permettant cette interprétation.

Les bases de connaissances grandissent de jour en jour ; être efficient signifie donc à la fois être précis mais aussi être rapide. L'approche STI se base sur l'étude de trois grands principes :

- L'étape d'assignation d'une propriété d'un KG en tant que relation entre deux colonnes du tableau (CPA).
- L'étape d'assignation d'une entité d'un KG à une cellule du tableau (CEA).
- L'étape d'assignation d'un type sémantique, c'est à dire une classe d'un KG, à une colonne du tableau (CTA).

On remarque que l'annotation du type de colonne constitue souvent la base des autres annotations [33]. Les méthodes plus traditionnelles font simplement la correspondance sur base de modèles graphiques [23], [29] ou des algorithmes itératifs [22], [28], [30], [31], [32]. Ces annotations se reposent sur des stratégies étudiant des métriques de similarité ou de vote majoritaire. Cependant, ces méthodes ne prennent pas en compte le fait que les bases de connaissances soient incomplètes et donc l'utilisation de ces trois grands axes seuls pour l'interprétation n'est pas suffisante.

Afin de pouvoir annoter les tableaux de manière plus efficace et d'apporter des algorithmes plus performants, on remarque que les méthodes récentes tendent à apporter de nouvelles fonctionnalités de plus en plus communes aux nouvelles méthodes pour améliorer cette rapidité et cette précision.

On parle ici de la préparation des données [44], des graphes de connaissances imbriqués [32] ou encore de l'utilisation de l'apprentissage automatique [33], [34].

D'autre part, nous présentons des outils récents qui mettent en place ces différentes approches.

Chaque outil à sa propre manière d'aborder le STI.

On remarque que les phases de prétraitement ont globalement le même objectif c'est-à-dire nettoyer les données en entrée et que chaque outil a sa propre manière de le faire. Certains utilisent des expressions régulières [49] et/ou des dictionnaires tels que celui d'Oxford [44] [45] ou bien des modèles de traitement de texte [42], etc. Au niveau des prétraitements, on remarque aussi qu'à part pour MantisTable, chaque outil change sa manière de prétraiter les données d'une version à l'autre.

La recherche d'entités est différente en fonction des outils : certains vont utiliser des lookups prédéfinis par les KB [42] [47] [46] [49], d'autres vont utiliser leur propre système de recherche ou bien reprendre un outil de recherche créé par une autre équipe [45]. On remarque qu'il existe différentes possibilités pour retrouver des entités candidates. La plupart des outils utilisent surtout le langage anglais mais il est aussi possible d'avoir du multi-langue [42].

Au niveau de la résolution des CTA, CPA, CEA, on remarque que certains principes reviennent régulièrement. Lorsqu'il y a plusieurs candidats pour une même cellule, des principes tels que le vote majoritaire, le calcul de l'occurrence ou encore de scores sont utilisés. Ces principes se retrouvent dans tous les outils. On remarque donc que la différence entre les outils se trouvent surtout dans leur manière de rechercher les entités candidates.

Chaque année, de nouvelles versions de ces outils sont déployées. La précision et la performance de ces outils sont comparées via le challenge SemTab.

Au niveau du travail pratique, nous avons réussi à créer un outil capable de comparer différents jeux de données entre eux et de les intégrer dans un même tableau de résultats. Cela donne une utilité supplémentaire au STI.

Via l'intervention de l'utilisateur, il est possible d'arriver à un résultat très complet. Lors de nos premiers tests, nous n'utilisions que l'outil MantisTable vu que celui-ci nous permettait de récupérer les colonnes-sujet et d'effectuer les tâches STI qui nous intéressaient. Mais l'outil prenait beaucoup de temps et sa façon d'exporter les tableaux ne convenait pas vraiment à ce que nous recherchions comme entrée pour notre algorithme de comparaison. Nous avons donc décidé de n'utiliser que la partie détection de colonne-sujet de MantisTable et l'outil MTab qui est plus performant et précis. Cette deuxième version a donné des résultats bien plus concluants que la première. Pour certains tableaux d'entrées, MantisTable pouvait prendre plusieurs minutes pour remplir les tâches STI alors qu'Mtab ne prenait que quelques secondes.

Au niveau de l'extraction de tableau HTML MTab, nous avons utilisé la bibliothèque Selenium qui nous semblait la plus simple d'utilisation. Nous n'avons pas eu le temps de

vérifier si d'autres bibliothèques plus performantes pouvaient être utilisées pour répondre à notre besoin.

Au niveau de notre intégration, nos requêtes permettent de couvrir les principaux concepts de DBpedia. Si des informations sont manquantes dans le tableau de résultats, cela veut dire que les données ne se trouvent tout simplement pas dans DBpedia ou pas dans le bon format. Nous n'avons travaillé que sur un seul KB, il est donc difficile d'avoir accès à toutes les informations liées à une entité. Il serait donc intéressant de pouvoir rajouter des KB telles que Wikidata. On pourrait mettre en place un concept [55] qui permettrait d'ajouter autant de KB que l'on souhaite. Le seul problème de cette approche, c'est que cela augmenterait le temps des processus STI qui est un problème retrouvé dans l'article ci-dessus.

Finalement, d'une part, notre outil apporte donc une preuve de concept pour un processus complet qui comprend le traitement de tableaux ainsi que leur comparaison en vue d'une éventuelle intégration dans une future application. D'autre part, l'état de l'art dirige vers la recherche et l'intégration de nouveaux concepts au sein de ce travail pratique comme l'apprentissage automatique ou encore les KGE.

8 Chapitre 8 : Menaces et limitations

8.1 Menaces et limitations de l'Etat de l'art

Il nous est impossible de garantir d'avoir sélectionné tous les articles pertinents, notre sélection a pu balayer un grand nombre d'articles mais il est clair que d'autres articles auraient pu nous permettre de compléter notre Etat de l'art en apportant d'autres informations.

Nous avons éliminé les articles redondants même si ceux–ci correspondaient bien à nos critères d'inclusion.

Cet état de l'art ayant été mené en équipe, il est évident que ce fut une force dans la quantité d'informations que nous avons pu sélectionner et traiter. Cependant, cela peut être également une faiblesse dans leur traitement car dépendant de notre propre vision des choses. Avec plus de temps, nous aurions pu améliorer les résumés et leur précision en lisant chacun chaque article sélectionné et en discutant à propos de celui-ci.

Les articles concernant les outils ne reprennent que les meilleurs outils de 2020, il aurait pu être intéressant d'également reprendre les meilleurs outils de 2019 pour avoir une meilleure vision de l'évolution entre les premiers outils de 2019 et 2020.

Avec plus de temps, nous aurions pu comparer les méthodes STI et les outils et faire un tableau de comparaison final entre les deux.

Au niveau du contexte, nous avons expliqué l'open data avec des articles qui datent de 2010. Même si ceux-ci sont complets et toujours valides, une utilisation d'articles plus récent permettrait d'avoir un regard plus neuf sur les LOD.

8.2 Menaces et limitations du travail pratique

Même si notre travail pratique a pu englober l'ensemble des processus liés au STI, nous avons repris en grande majorité du code existant afin de traiter l'interprétation des tableaux, il faut donc considérer que la probabilité de rencontrer un bug dans le traitement soit significative. D'ailleurs, le manque de temps et d'investissement dans le travail pratique nous a réduit à faire un minimum de cas de tests et de documentations.

Il est clair que notre méthode pratique est une amorce pour un travail beaucoup plus conséquent. Nous n'avons pas utilisé les Gold Standards afin de mesurer ou de comparer nos résultats vis-à-vis des outils existants.

Un point intéressant aurait été de reprendre la détection de colonne-sujet d'une des méthodes que l'on retrouve dans les méthodes de la QR2 [Chapitre 4].

Au niveau de l'extraction du tableau HTML, si le site MTab app décide de changer son HTML, il faudra changer l'algorithme d'extraction.

Nous sommes restés sur un seul KB, l'utilisation de plusieurs KB permettrait de compléter notre intégration de jeux de données.

Pour augmenter la vitesse des requêtes SPARQL, il aurait été préférable de télécharger DBpedia en local. Cela permettrait d'être moins dépendant du réseau Internet (ralentissement ou coupure du réseau par exemple) et donc, en plus d'avoir des requêtes, être plus rapide.

Notre travail est une preuve de concept, nous n'avons donc pas jugé utile de faire des tests unitaires.

Il serait bon de prévoir un convertisseur automatique de JSON à CSV pour la partie détection de colonne-sujet afin de pouvoir importer le format CSV directement.

9 Chapitre 9 : Perspectives d'améliorations

9.1 Orientations théoriques

Aux vues des dernières méthodes et avancées technologiques, nous sommes persuadés que la recherche doit s'étendre principalement sur des tableaux avec des entités inconnues pour évaluer de nouvelles méthodes d'apprentissage automatique, prendre en compte le manque de connaissance et d'étendre ce principe sur les autres tâches d'annotation (annotation des cellules et des propriétés).

Nous pensons que les futures approches STI devraient se pencher sur la modélisation neuronale du langage. En effet, les méthodes supervisées commencent à faire leurs preuves et, d'un autre côté, les bases de connaissances grossissent de jour en jour ; il serait donc opportun de se pencher sur l'intelligence artificielle et les modèles de réseaux de neurones et donc le Deep Learning dans l'approche STI.

De plus, les KGE sont des approches très récentes et très intéressantes dans l'utilisation des KG à partir des approches d'interprétation. Nous pensons que l'étude complète de cette approche avec le Deep Learning pourrait ouvrir vers de nouveaux concepts et améliorations de la recherche.

9.2 Améliorations du travail pratique

Au niveau des améliorations, plusieurs points peuvent être mis en avant :

Nous en sommes restés à une preuve de concept afin de montrer la faisabilité d'un procédé d'intégration et de comparaison. Nous avons donc utilisé des jeux de données très basiques mais, si ce travail doit être continué par d'autre personnes, ils pourront exploiter les puissantes bibliothèques que Python met à disposition pour accomplir des procédés plus complexes.

A l'heure actuelle, la préparation des données se fait deux fois. Une première fois lorsqu'il faut détecter la colonne-sujet en utilisant l'algorithme MantisTable et une fois de plus lors de l'utilisation de l'outil MTab. Il faudrait être capable de détecter quelle méthode de préparation de données est la plus efficiente (MTab ou MantisTable) et de placer celle-ci à la toute première étape du processus de traitement.

Nous n'avons utilisé que le KB DBpedia et les outils MantisTable/MTab 2019. Il serait intéressant aussi d'utiliser le KB Wikidata avec les nouvelles versions des outils MantisTable et MTab 2020.

Le fait de se baser sur l'extraction HTML via la bibliothèque *Selenium* était nécessaire, il serait intéressant de contacter l'équipe MTab en vue de pouvoir appeler directement leur API pour faciliter le processus de récupérer des annotations.

Un autre point à améliorer est la complexité de l'algorithme qui est de n^4.

Du point de vue utilisateur, il serait opportun d'utiliser une seule et même interface graphique et de pouvoir tout gérer directement à partir de celle-ci.

10 Chapitre 10 : Conclusion du mémoire

Dans ce travail, nous introduisons le problème de « comment automatiser efficacement l'interprétation sémantique tabulaire (STI) afin de pouvoir lier des jeux de données entre eux ? ». Nous avons étudié ce problème de deux manières différentes.

La première où nous parcourrons et comparons différentes méthodes STI. Ces méthodes montrent les approches classiques (itératives ou basée sur un modèle graphique) mais aussi les approches plus innovantes comme l'utilisation de l'apprentissage automatique et des graphes de connaissances imbriqués (KGE).

La seconde où nous étudions les meilleurs outils existants. Ceux-ci résultent de l'application de différents algorithmes pour permettre une annotation précise et rapide des tableaux.

Ensuite, nous avons mis en place notre propre méthode STI qui se base sur deux outils récents existants : MTab et MantisTable. Celle-ci permet une annotation efficiente de nos tableaux pour ensuite pouvoir les intégrer dans notre algorithme de comparaison afin de pouvoir lier ces jeux de données entre eux. Ce cas pratique nous montre une preuve de concept de l'automatisation, de l'interprétation et de la comparaison de jeux de données. Le tableau de résultats pouvant être utilisé dans le but d'une intégration ou encore d'une liaison entre les jeux de données initiaux.

Ce travail apporte donc plusieurs contributions. Il contribue à l'état de l'art en discutant des dernières avancées dans les approches STI et en dirigeant vers des concepts clés en vue de créer ou d'améliorer un outil d'interprétation. Ces concepts pouvant être amenés à être implémentés au sein du travail pratique qui a été fourni.

Les méthodes supervisées faisant leurs preuves, nous pensons également que les futures approches STI devraient étudier la modélisation neuronale du langage, c'est-à-dire intégrer le Deep Learning dans l'approche STI.

Une amélioration de notre travail pratique pourrait être envisagée comme la création d'un nouvel outil alliant les bénéfices de MTab, MantisTable et notre algorithme de comparaison dans une même interface. Ce nouvel outil devra intégrer les Gold Standards afin de pouvoir le comparer aux autres ainsi que l'utilisation de KG autre que DBpedia. Et pourquoi pas y intégrer des concepts de l'intelligence artificielle discuté ci-dessus.

11 Travaux cités

[1] O. Hartig et A. Langegger, «A Database Perspective onConsuming Linked Data on the Web,» pp. 1-10, Octobre 2010.

[2] M. Cremaschi, F. De Paoli, A. Rula et B. Spahiu, «A fully automated approach to a complete Semantic Table Interpretation,» pp. 1-32, avril 2020.

[3] M. Hauseblas et M. Karnstedt, «Understanding Linked Open Data as a Web-Scale Database,» pp. 1-7, Janvier 2010.

[4] B. L, «Open Data définition : qu'est-ce que c'est ? À quoi ça sert ?,» p. 1, 5 Novembre 2019.

[5] A. Zimmermann, «RDF 1.1: On Semantics of RDF Datasets,» 2014.

[6] J. Lehmann, R. Isele, M. Jakob, J. Anja , D. Kontokostas, P. N Mendes, S. Hellmaan, M. Morsey, P. Van Kleef, S. Auer et C. Bizer, «DBpedia – A Large-scale, MultilingualKnowledge Base Extracted from Wikipedia,» pp. 1-29, Janvier 2014.

[7] M. Färber, B. Ell, C. Menne et A. Rettinger, «A Comparative Survey of DBpedia, Freebase,OpenCyc, Wikidata, and YAGO,» pp. 1-26, Juillet 2015.

[8] K. D. Bollacker, C. Evans, P. Paritosh et T. Sturge, «Freebase: A collaboratively created graph database for structuring human knowledge,» *Proc. Sigmod.,* pp. 1247-1250, 2008.

[9] T. Pellissier Tanon, D. Vrandecic, S. Schaffert, T. Steiner et L. Pintscher, «From Freebase to Wikidata: The Great Migration,» pp. 1-10, 2016.

[10] A. S. Said, K. Rehan et S. Khaled, «A Survey of Semantic Analysis Approaches,» *Proceedings of the International Conference on Artificial Intelligence and Computer Vision (AICV2020),* pp. 61-70, 2020.

[11] E. Tacconelli, «Systematic reviews: CRD's guidance for undertaking reviews in health care,» pp. 226-266, Avril 2010.

[12] F. Manola et E. Miller, «RDF Primer,» *W3C recommendation, World Wide Web Consortium, 2,* 2004.

[13] M. Cremaschi, R. Avogadro, A. Slano et E. Jiménez-Ruiz, «STILTool: A Semantic Table Interpretation evaLuation Tool,» pp. 1-7, Novembre 2020.

[14] M. Hulsebos, K. Hu, M. Bakker, E. Zgraggen, A. Satyanarayan, T. Kraska, Ç. Demiralp et C. Hidalgo, «Sherlock: A Deep Learning Approach to Semantic Data Type Detection,» pp. 1-9, 2019.

[15] E. Jiménez-Ruiz, O. Hassanzadeh, V. Efthymiou, J. Chen, K. Srinivas et V. Cutrona, «Results of SemTab 2020,» pp. 1-9, 4 Janvier 2021.

[16] D. Ritze, O. Lehmberg et C. Bizer, «T2Dv2 Gold Standard for Matching Web Tables to DBpedia».

[17] D. Ritze, O. Lehmberg, R. Meusel et C. Bizer, «Web Data Commons - Web Table Corpora».

[18] M. Taheriyan, C. A. Knoblock, P. Szekely et J. Luis Ambite, «Leveraging Linked Data to Discover Semantic,» pp. 1-16, 23 Septembre 2016.

[19] S. Choudhary, T. Luthra, A. Mittal et R. Singh, «A Survey of Knowledge Graph Embedding and Their Applications,» 2021.

[20] A. Bordes, N. Usunier, A. Garcia-Duran et J. Weston, «Translating Embeddings for Modeling Multi-relational Data,» 2013.

[21] D. Ritze, O. Lehmberg et C. Bizer, «T2D Gold Standard for Matchnig Web Tables to DBpedia,» University Of Mannheim, [En ligne]. Available: http://webdatacommons.org/webtables/goldstandard.html. [Accès le 08 07 2021].

[22] Z. Zhang, «Towards Efficient and Effective Semantic Table Interpretation,» pp. 487-502, 2014.

[23] G. Limaye, S. Sarawagi et S. Chakrabarti, «Annotating and searching web tables using entities, types and relationships.,» *Proceedings of the VLDB Endowment 3(1-2),* p. 1338–1347, 2010.

[24] J. Wang, B. Shao, H. Wang et K. Q. Zhu, «Understanding tables on the web,» *Proceedings of the 31st International,* p. 141–155, 2012.

[25] W. Wu, H. Li, H. Wang et Q. K. Zhu, «Probase: a probabilistic taxonomy for text understanding.,» *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data,* p. 481–492, 2012.

[26] E. Yu, «Modelling Strategic Relationships for Process Reengineering PhD. Computer Science University of Toronto, Toronto,» 1995.

[27] P. Venetis, A. Halevy, J. Madhavan et M. Pasca, «Recovering semantics of tables on the web.,» *Proceedings of VLDB Endowment 4(9),* p. 528–538, 2011.

[28] Z. Zhang, «Learning with Partial Data for Semantic Table Interpretation,» *Knowledge Engineering and Knowledge Management,* pp. 607-6018, 2014.

[29] V. Mulwad, T. Finin et A. Joshi, «Semantic Message Passing for Generating Linked Data from Tables,» 2013.

[30] Z. Zhang, «Effective and Efficient Semantic Table Interpretation Using TableMiner+,» *Semantic Web,* 2016.

[31] D. Ritze, O. Lehmberg et C. Bizer, «Matching HTML Tables to DBpedia,» pp. 1-6, 2015.

[32] V. Efthymiou, O. Hassanzadeh, M. Rodriguez-Muro et V. Christophides, «Matching Web Tables with Knowledge Base Entities: From Entity Lookups to Entity Embeddings,» pp. 260-277, 2017.

[33] M. Pham, S. Alse, C. A. Knoblock et P. A. Szekely, «Semantic Labeling: A Domain-Independent Approach,» pp. 446-462, 2016.

[34] K. Takeoka, M. Oyamada, S. Nakadai et T. Okadome, «Meimei: An Efficient Probabilistic Approach for Semantically Annotating Tables,» vol. Proceedings of the AAAI Conference on Articial Intelligence 33, pp. 281-288, 2019.

[35] S. Neumaier, J. Umbrich, J. X. Parreira et A. Polleres, «Multi-level Semantic Labelling of Numerical Values,» pp. 428-445, 2016.

[36] M. M. Hearst, «Automatic acquisition of hyponyms from large text corpora. In: Proceedings of the 14th Conference on Computational Linguistics,» *Proceedings of the 14th Conference on Computational Linguistics - Volume 2.,* p. 539–545, 1992.

[37] F. M. Suchanek, G. Kasneci et G. Weikum, «YAGO: a Core of Semantic Knowledge.,» *16th International World Wide Web Conference,* p. 697–706, WWW2007.

[38] C. Bizer, J. Lehmann, G. Kobilarov et S. Auer, «A Crystallization Point for the Web of Data.,» *Web Semantics: Science, Services and Agents on the World Wide Web 7,* pp. 154-165, 2009.

[39] C. D. Manning, P. Raghavan, H. Schtextbackslashütze et E. Corporation, «Introduction to Information Retrieval,» *Inf. Retr,* vol. 13, 2018.

[40] S. Zwicklbauer, C. Seifert et M. Granitzer, «DoSeR - A Knowledge-Base-Agnostic Framework for Entity Disambiguation Using Semantic Embeddings,» pp. 182-198, 2016.

[41] C. L. Clarke, M. Kolla, G. V. Cormack, O. Vechtomova, A. Ashkan, S. Büttcher et I. MacKinnon, «Novelty and diversity in information retrieval evaluation.,» *Proceedings of the 31st Annual International ACM SIGIR 08,* pp. 659-666, 2008.

[42] N. Phuc, K. Natthawut , I. Ryutaro et T. Hideaki, MTab: Matching Tabular Data to Knowledge, 2019.

[43] N. Phuc, Y. Ikuya, K. Natthawut, I. Ryutaro et T. Hideaki, MTab4Wikidata at SemTab 2020: Tabular Data Annotation with Wikidata, 2020.

[44] M. Cremaschi, R. Avogadro et D. Chieregato, MantisTable: an Automatic Approach for the Semantic Table Interpretation, 2019.

[45] M. Cremaschi, R. Avogadro , A. Barazetti et D. Chieregato, MantisTable SE: an Efficient Approach for the Semantic Table Interpretation, 2020.

[46] S. Chen, A. Karaoglu, C. Negreanu, T. Ma, J.-G. Yao, J. Williams, A. Gordon et C.-Y. Lin, LinkingPark: An Integrated Approach for Semantic Table Interpretation, 2020.

[47] Y. Chabot, T. Labbé, J. Liu et R. Troncy, DAGOBAH: An End-to-End Context-Free Tabular Data Semantic Annotation System, 2019.

[48] E. Jiménez-Ruiz, O. Hassanzadeh, V. Efthymiou, J. Chen, K. Srinivas et V. Cutrona, «Results of SemTab 2020,» 2020.

[49] V.-P. Huynh, J. Liu, Y. Chabot, T. Labbé, P. Monnin et R. Troncy, DAGOBAH: Enhanced Scoring Algorithms for Scalable Annotations of Tabular Data, 2020.

[50] X. Han, S. Cao, X. Lyu et Y. Lin, «OpenKE: An Open Toolkit for Knowledge Embedding,» pp. 139-144, 2018.

[51] M. Cremaschi, A. Rula , A. Siano et F. De Paoli, «MantisTable: A Tool for Creating Semantic Annotations on Tabular Data,» pp. 18-23, 2019.

[52] S. Markandeya et K. Roy, «ERP and SAP Overview,» 2014.

[53] T. Mikolov, K. Chen, G. S. Corrado et J. Dean, «Efficient Estimation of Word Representations in Vector Space,» *Proceedings of Workshop at ICLR,* 2013.

[54] G. A. Miller, «Wordnet: A lexical database for english. Commun. ACM 38(11),» pp. 39-41, 1995.

[55] T. Knap, «Towards Odalic, a Semantic Table Interpretation Tool in,» Mai 2020.

[56] B. Moreau, N. Terpolilli et P. Serrano-Alvarado, «SemanticBot: Intégration Semi-Automatique de Données auWeb des Données,» pp. 1-5, 24 Janvier 2020.