DAGOBAH : Un système d'annotation sémantique de données tabulaires indépendant du contexte

Yoan Chabot, Thomas Labbé, Jixiong Liu, Raphaël Troncy

▶ **To cite this version:**

Yoan Chabot, Thomas Labbé, Jixiong Liu, Raphaël Troncy. DAGOBAH : Un système d'annotation sémantique de données tabulaires indépendant du contexte. 31es Journées francophones d'Ingénierie des Connaissances, Sébastien Ferré, Jun 2020, Angers, France. hal-02888088

HAL Id: hal-02888088
https://hal.science/hal-02888088

Submitted on 7 Jul 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

DAGOBAH : Un système d'annotation sémantique de données tabulaires indépendant du contexte

Yoan Chabot[1], Thomas Labbé[1], Jixiong Liu[1,2], Raphaël Troncy[2]

[1] Orange Labs, France
yoan.chabot|thomas.labbe@orange.com

[2] EURECOM, Sophia Antipolis, France
raphael.troncy@eurecom.fr

Résumé : Cet article présente le système DAGOBAH permettant d'annoter sémantiquement des tables à l'aide d'entités Wikidata et DBPedia. Le système proposé annote les cellules et les colonnes d'une table et identifie des relations entre ces colonnes. Pour cela, un processus allant du pré-traitement des tables jusqu'à l'enrichissement d'un graphe de connaissances existant en utilisant uniquement les informations contenues dans la table est utilisé. Pour répondre au problème spécifique du typage des colonnes des tables, deux techniques sont introduites pour sélectionner des types suffisamment représentatifs tout en restant à un niveau de spécificité porteur d'informations. Les types ainsi identifiés sont ensuite utilisés dans un processus de désambiguisation des cellules. Le système a été évalué lors du challenge SemTab2019 (Jiménez-Ruiz *et al.*, 2020) de la conférence ISWC 2019 où il a obtenu des résultats prometteurs (Chabot *et al.*, 2019b).

Mots-clés : données tabulaires, graphe de connaissances, annotation sémantique, plongement, DAGOBAH, SemTab2019.

1 Introduction

L'annotation de tables à l'aide de graphes de connaissances est un problème crucial et complexe. Un grand nombre de gisements de données internes aux entreprises ainsi qu'une part importante des données du Web sont représentés sous forme de tables. La capacité à annoter ces dernières en utilisant un graphe de connaissances (encyclopédiques comme DBPedia et Wikidata ou contenant des informations d'entreprises) permet l'intégration de nouveaux services basés sur la sémantique. Cela ouvre notamment la voie à des outils plus efficaces pour l'interrogation (e.g. "aller au delà de simples mots-clefs" pour la recherche de jeux de données (Chapman *et al.*, 2019)), la manipulation et le traitement de corpus de tables hétérogènes (Chabot *et al.*, 2019a). Les données tabulaires sont toutefois difficiles à interpréter automatiquement, principalement à cause du manque de contexte disponible (par opposition aux textes en langage naturel) pour résoudre les ambiguïtés (Table 1). A ce problème s'ajoutent des hétérogénéités syntaxiques souvent mêlées à des artefacts typographiques (Table 2), mais aussi des dispositions et formats de tables parfois difficiles à appréhender.

Pour répondre à ce besoin, nous proposons un système d'annotation de données tabulaires automatique allant du pré-traitement jusqu'à l'identification d'entités et de relations. Afin de garantir sa généricité, ce système utilise uniquement les informations contenues dans les tables et ne prend donc pas en compte les titres et légendes ou d'éventuelles pages englobantes. Les principales contributions du système DAGOBAH présentées dans cet article sont :

— Une nouvelle chaîne de pré-traitement améliorant les résultats du framework DWTC (Eberius *et al.*, 2015).
— Un processus d'annotation automatique en trois étapes (annotation cellule-entité, annotation colonne-type et annotation colonnes-propriétés) utilisant les bases de connaissances Wikidata et DBPedia.
— Une approche alternative utilisant un partitionnement sur des plongements du graphe de connaissances Wikidata.

TABLE *1 – Extrait d'une table métier*

id	name	duration	genre[]	act[].familyName	act[].givenName	price.value
3540	Hector - HD	4980	drame	Solemani , Allen	Sarah , Keith	
296x	Les faussaires	5640	thriller , drame	Markovics , Diehl	Karl , August	5.99
8801	Les bien-aimés - HD	7920	comédie dramatique	Deneuve , Mastroianni	Catherine , Chiara	2.99
01j6	The village green	600	Série-Humour	Pagett , Walker	Daniel , Tim	

TABLE *2 – Extrait de la table* 54719588_0_8417197176086756912 *du challenge SemTab2019*

New ?	Title	Director	Year	Grade
66.66%		class=xl36 style='mso-ignore :colspan'>	1992.82	2.78
1	Big Momma's House	Gosnell	2000	
	Pi	Aronofsky	1998	A
	Fight Club	Fincher	1999	A-
2	Keeping The Faith	Norton ?	2000	
3	Royal Tenenbaums	Anderson	2001	B

La suite de cet article est structurée de la manière suivante. Un état de l'art des travaux dans le domaine de l'annotation de données tabulaires est proposé dans la section 2. Le système DAGOBAH est ensuite présenté dans la section 3. Une évaluation du système sur le challenge SemTab2019 est ensuite exposée dans la section 4. Enfin, une synthèse sur les capacités et les limites de l'approche ainsi que les travaux futurs viennent conclure le papier en section 5.

2 État de l'art

Le problème d'annotation sémantique peut être divisé en trois tâches (Jiménez-Ruiz *et al.*, 2020) : l'annotation cellule-entité (CEA), l'annotation colonne-type (CTA) et l'annotation colonnes-propriété (CPA).

Concernant les correspondances d'entités (CEA), deux approches sont utilisées dans la littérature. La première, largement répandue, consiste à trouver une correspondance pour chaque cellule d'une table donnée (Limaye *et al.*, 2010) (Bhagavatula *et al.*, 2015) (Kilias *et al.*, 2018) (Fernandez *et al.*, 2018). La seconde, quant à elle, tente de faire correspondre une ligne entière d'une table et une entité du graphe de connaissances cible (Efthymiou *et al.*, 2017) en partant du principe qu'une ligne représente une entité principale (colonne clé) et des attributs associés (autres colonnes). Ces correspondances peuvent être réalisées à l'aide de recherches syntaxiques (c'est à dire par comparaison de chaînes de caractères), d'alignement d'ontologies ou en exploitant des plongements lexicaux. La désambigüisation des entités candidates est ensuite traitée comme une tâche de classement et de décision classique, via l'utilisation d'heuristiques ou d'algorithmes de type PageRank (Efthymiou *et al.*, 2017), de mesure de similarité (Kilias *et al.*, 2018) (Fernandez *et al.*, 2018) ou de modèles à base de graphes (Ibrahim *et al.*, 2016).

Les principaux travaux sur le typage des colonnes (CTA) se basent sur l'inférence des classes à partir des entités issues de la tâche de CEA. Différents algorithmes existent, intégrant des heuristiques plus ou moins complexes orientées autour du vote majoritaire (Mulwad *et al.*, 2010). Enfin, l'extraction de relations (CPA) est généralement réalisée par correspondance de paires d'éléments de colonnes au regard des types et entités précédemment déterminés (Ran *et al.*, 2016).

Si les approches précédentes sont intrinsèquement séquentielles, on peut noter quelques travaux qui ambitionnent de réaliser conjointement les trois tâches via des modèles orientés graphes (Limaye *et al.*, 2010) (Zhang, 2014).

En parallèle, des efforts ont été réalisés par la communauté sur la mise à disposition de corpus d'évaluation pour ces différentes tâches. Si le premier corpus portait sur 428 tables Wikipédia (Limaye *et al.*, 2010), les suivants ont ensuite été étoffés soit en volumétrie (WTC (Lehmberg *et al.*, 2016) : 233 millions de tables réparties en trois catégories), soit en précision (T2D Gold Standard (Ritze *et al.*, 2015) : 1748 tables issues du WTC dont les lignes, les attributs et les tables ont été annotés manuellement avec des instances, propriétés et classes DBpedia respectivement). Certains corpus ont également été personnalisés pour répondre à

un besoin particulier (Wikipedia Gold Standard (Efthymiou *et al.*, 2017)) mais ne font pas à ce jour office de référence d'évaluation au sein de la communauté.

Plus récemment, le challenge SemTab2019 proposait à des équipes de comparer les performances de leurs systèmes d'annotation de données tabulaires pour les trois tâches mentionnées précédemment. Le challenge s'est déroulé en quatre manches (Jiménez-Ruiz *et al.*, 2020) proposant des corpus différents par leur taille (respectivement 63, 11925, 2162 et 818 tables) et leur nature avec des tables de plus en plus complexes tant au niveau des formats que des informations à traiter (Hassanzadeh *et al.*, 2019). Dix sept équipes ont participé au challenge avec sept d'entre elles ayant participé à au moins deux des quatre manches du challenge, dont le système DAGOBAH présenté dans cet article. MTab (Nguyen *et al.*, 2019) propose l'utilisation d'une approche probabiliste couplée à des requêtes sur plusieurs services de recherche sur les bases DBpedia, Wikidata et Wikipedia en utilisant des stratégies multilingues. Une majorité des approches (Cremaschi *et al.*, 2019), (Morikawa, 2019), (Oliveira & D'Aquin, 2019), (Thawani *et al.*, 2019), (Steenwinckel *et al.*, 2019) prennent la forme de système à base de recherche de candidats dans les services précédents, de calcul de similarité syntaxique et de votes majoritaires, telle que notre approche de base décrite en section 3.2. IDLab propose une approche itérative où l'annotation de cellules ayant peu d'ambiguïté vient renforcer au fur et à mesure des itérations la désambiguisation des cellules plus complexes (Steenwinckel *et al.*, 2019). Tabularisi utilise les alias des entités définies dans Wikidata et s'assure que les colonnes ont une cohérence sémantique à l'issu du processus d'annotation (Thawani *et al.*, 2019). ADOG créé un index de DBpedia dans la base de données NoSQL ArrangoDB et utilise essentiellement la distance de Levenstein pour mesurer la différence entre les valeurs des cellules et les étiquettes des entités (Oliveira & D'Aquin, 2019). MantisTable se démarque grâce à une interface graphique poussée permettant de configurer le processus d'annotation automatique et et visualiser les résultats (Cremaschi *et al.*, 2019). LOD4ALL utilise essentiellement des requêtes SPARQL ASK pour obtenir des candidats et déduire des contraintes de type sur les colonnes (Morikawa, 2019). Les évaluations montrent que les approches ayant recours à des techniques de recherche élaborées et à des tâches de nettoyage et de pré-traitement optimisées pour les tables considérées obtiennent de meilleures résultats.

L'approche DAGOBAH propose une approche originale, basée sur des plongements de graphes, se démarquant des autres concurrents. Si cette technique a obtenu de très bon résultats sur la première manche, elle n'a pas encore permis d'atteindre les résultats escomptés dans les autres manches. Cela est en partie due à une préparation des données moins poussée que pour les autres approches, l'idée étant de proposer une solution générique pouvant s'appliquer à une grande variété de tables. Cependant, les approches à base de plongements restent une piste intéressante à explorer. Elles permettent de résoudre plus efficacement les ambiguïtés lorsque le contenu de la table est peu explicite. De plus, elles présentent une meilleure robustesse aux changements pouvant survenir dans les jeux de données ou dans les graphes de connaissances.

3 DAGOBAH : Un système d'annotation de données tabulaires de bout en bout

DAGOBAH est implémenté comme un ensemble d'outils utilisés de manière séquentielle pour atteindre les objectifs suivants :

1. L'identification de correspondances entre une table et un graphe de connaissances (i.e. processus d'annotation sémantique). Cet article se concentre uniquement sur les techniques employées et les résultats obtenus pour cette fonctionnalité.
2. L'enrichissement des graphes de connaissances en transformant en triplets les connaissances contenues dans les tables. Le graphe de connaissances cible de DAGOBAH est Wikidata pour plusieurs raisons incluant la fraîcheur des informations qu'il contient, sa large couverture et la qualité des données (Färber *et al.*, 2015). Par conséquent, des adaptations ont dû être réalisées pour le challenge SemTab2019 afin de supporter DBPedia.
3. La production de métadonnées pouvant être utilisées pour le référencement et l'indexation de jeux de données et des processus de recherche et de recommandation sub-

séquents (Chabot *et al.*, 2019a).

Pour fournir ses fonctionnalités d'annotation sémantique, DAGOBAH est structuré de la manière suivante.

Le module de pré-traitement (Section 3.1) réalise le nettoyage des tables et une première caractérisation de leur structure et de leur contenu. Les modules d'annotations accomplissent ensuite les tâches du challenge à proprement parler. Deux méthodes ont été étudiées pour mener à bien ces tâches : une méthode de base exploitant des services de recherche (i.e. "lookup") et des mécanismes de votes (Section 3.2) et une approche géométrique basée sur un partitionnement appliqué à des plongements du graphe Wikidata (Section 3.3).

3.1 Pré-traitement des données tabulaires

Plusieurs informations liées à la disposition de la table et aux types de données en présence sont utiles afin de traiter correctement les informations contenues dans les tables. La chaîne de pré-traitement intégrée dans DAGOBAH génère plusieurs informations dont l'orientation de la table, la présence ou non d'une en-tête, la présence d'une colonne clé et son index ainsi que les types primitifs des colonnes. De plus, elle supporte plusieurs formats de tables en entrée : CSV, TXT, JSON, XLS et XLSX. Dans un contexte d'exploitation réel dans lequel les utilisateurs possèdent peu ou pas de connaissances sur les tables à traiter, les informations produites sont décisives pour la qualité des annotations. Cette chaîne s'est inspirée du DWTC-Extractor [1] en améliorant plusieurs des algorithmes proposés. La précision de la chaîne DAGOBAH a été évaluée sur le corpus de la première manche du challenge (Table 3) et comparée à une version modifiée de l'outil du DWTC (qui n'utilise pas les balises HTML et supporte ainsi davantage de formats de données).

Détection d'orientation de tables. L'algorithme proposé par le DWTC-Extractor est basé sur la taille des chaînes de caractères et l'hypothèse que les valeurs des cellules d'une même colonne ont une longueur similaire. Cependant, la robustesse de cet algorithme peut être améliorée. En effet, deux chaînes de caractères représentant des éléments très différents peuvent avoir la même longueur (e.g. "Paris" et "$10cm^2$').

Pour pallier cette limite, DAGOBAH introduit un nouvel algorithme basé sur un système de typage primitif des colonnes en utilisant 11 types Tp_i (chaîne de caractères, nombres flottants, date, etc.).

Sur la base de ces types, un score d'homogénéité $Hom(x)$ est calculé sur chaque ligne et chaque colonne x (équation 1). La moyenne de l'ensemble des lignes et des colonnes est ensuite comparée et, selon le ratio, la table est dite "HORIZONTAL" ou "VERTICAL" (dans une table horizontale, les entités sont représentées en ligne et les attributs en colonne).

$$Hom(x) = [\frac{1}{|x|} \sum_{Tp_i \in x} (1 - (1 - 2 \times \frac{|Tp_i|}{|x|})^2)]^2 \quad (1)$$

où :
x est une ligne ou une colonne de la table,
$|x|$ est le nombre d'éléments de la ligne ou de la colonne considérée,
Tp_i est le type primitif $i, i \in [1; 11]$,
$|Tp_i|$ est le nombre d'occurrences du type i dans x.

Extraction d'en-tête. L'algorithme utilisé dans DAGOBAH est basé sur les types primitifs évoqués précédemment. L'extracteur d'en-tête utilise les hypothèses suivantes : en règle général, l'en-tête d'une colonne est une chaîne de caractères et possède un type différent du reste de la colonne. L'utilisation de ces deux heuristiques permet d'identifier assez précisément si une table contient ou non des en-têtes (Table 3). Il est à noter que le framework DWTC offre également un outil de détection d'en-tête mais ce dernier contient plusieurs bugs rendant son évaluation impossible.

1. https ://github.com/JulianEberius/dwtc-extractor

Détection de colonne clé. La colonne clé d'une table contient les identifiants des entités décrites par les autres colonnes. Il est à noter que dans la première version de la chaîne de pré-traitement, nos algorithmes font l'hypothèse que la clé d'une table est une colonne seule ce qui n'est pas toujours le cas en réalité (e.g. une table représentant des personnes et ayant pour colonne clé une agrégation des colonnes "nom" et "prénom"). L'algorithme proposé tire partie des informations de typage définies lors de la détection d'orientation et, à l'instar de l'extraction d'en-tête, d'heuristiques. La colonne clé est caractérisée comme une colonne, se situant à gauche dans la table, contenant des chaînes de caractères dont la majorité sont des valeurs uniques (identifiant des entités).

Cette chaîne de pré-traitement a été particulièrement utile durant la première manche du challenge pour identifier automatiquement les informations contenues dans les en-têtes des tables ainsi que la colonne à annoter (la colonne à annoter dans les tables lors du challenge était la colonne clé). L'information de colonne clé peut également être utile pour déterminer le type d'entités décrites par une table. Enfin, lorsque l'objectif est d'enrichir un graphe de connaissances avec les informations des tables, la détection de la colonne clé peut se révéler utile pour déterminer le sujet des triplets RDF générés. La mise à disposition des colonnes à annoter lors des manches suivantes a rendu de fait une partie de la chaîne de pré-traitement moins indispensable. Toutefois, cela ne remet pas en question l'utilité de tels outils pour des applications réelles.

TABLE *3 – Précision des tâches de pré-traitement sur le corpus de la manche 1*

Tâche/Outil	DWTC	DAGOBAH
Détection de l'orientation	0.9	**0.957**
Extraction d'en-tête	Non évalué	**1.0**
Détection de colonne clé	0.857	**0.986**

3.2 Méthode à base de vote majoritaire

Recherche d'entités. Afin d'optimiser les opérations de recherche (i.e. lookups), un premier processus de nettoyage est appliqué aux données. L'objectif de ce dernier n'est pas de corriger tous les problèmes des chaînes de caractères, mais d'avoir une transformation générique couvrant les artefacts les plus courants. Cela inclut l'homogénéisation de l'encodage et la suppression des caractères spéciaux (parenthèses, crochets et caractères non alphanumériques).

Cinq services de recherche sont ensuite utilisés simultanément pour trouver des entités candidates à partir du contenu des cellules : l'API Wikidata, le moteur Cirrus Search de Wikidata, l'API DBPedia, l'API Wikipédia ainsi qu'un index Elasticsearch interne dans lequel ont été stocké les labels, les alias et les types associés aux QIDs Wikidata (étape 1 de la Figure 1). Cette dernière source permet de garder le contrôle des stratégies de recherche et ainsi d'augmenter le nombre de candidats potentiels en élargissant le champ de recherche. A l'inverse, les autres APIs sont des services utiles mais fonctionnent comme des boîtes noires sur lesquelles nous n'avons pas de contrôle quant aux résultats retournés. Dans cette étape, un enrichissement des entités avec les informations issues de l'index Elasticsearch est également réalisé, et ce, afin de disposer, pour chaque entité, de son QID, c'est à dire de son identifiant unique dans Wikidata ce qui nous sera utile pour la suite du processus.

Un ensemble de traitements d'agrégation et d'enrichissement des candidats est ensuite effectué (étape 2 de la Figure 1) :

— Un comptage des occurrences, sur la base du QID, est réalisé à la sortie du processus de recherche afin de sélectionner les entités candidates les plus fréquemment retournées parmi les cinq services ainsi que leurs types.

— DBPedia étant la base de connaissances cible du challenge, les QIDs et les types Wikidata sont traduits en entités DBPedia. Pour cela, des requêtes SPARQL tirant parti des prédicats *owl:sameAs* et *owl:equivalentClass* sont utilisées.

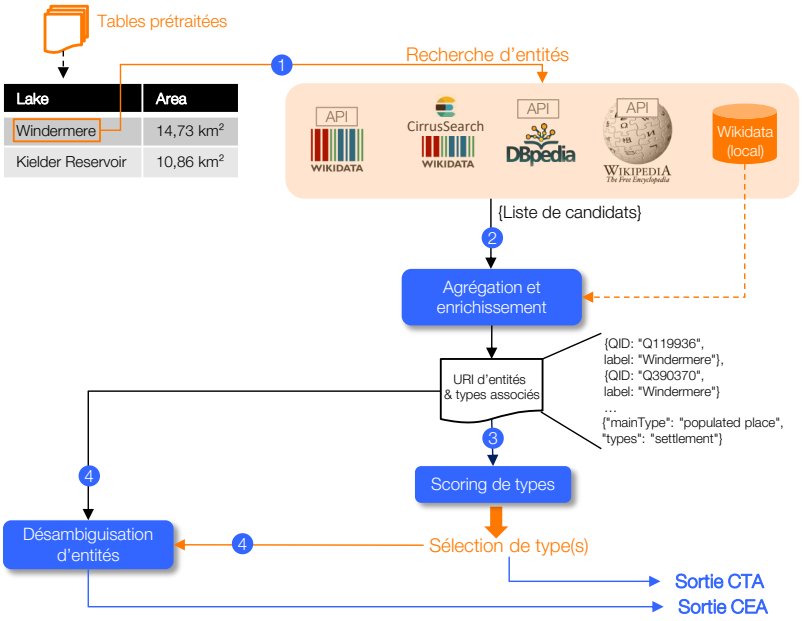

FIGURE *1 – Description du processus utilisé dans l'approche de base*

— En parallèle, le serveur Elasticsearch interne est interrogé afin de récupérer les types Wikidata (à partir des QID ou des labels et alias dans le cas de candidats DBpedia ou Wikipédia) qui sont utilisés comme les types pivots du système.
— Pour chaque entité candidate, la hiérarchie de classes parentes est enfin extraite à l'aide de requêtes SPARQL sur le endpoint DBPedia afin d'augmenter la liste de types candidats.

Typage des colonnes. L'étape suivante consiste à filtrer les types non pertinents, c'est à dire pas suffisamment représentatifs ou trop peu spécifiques. Un comptage basique des occurrences de type ne se révèle pas toujours pertinent étant donné que le type de la colonne considérée peut être très spécifique et pas suffisamment fréquent pour être considéré comme un type candidat. Pour corriger ce problème, un seuil sur les scores relatifs est utilisé. Pour chaque type $t_i \in \{t\}_y$ (liste de l'ensemble des types observés au sein de la colonne y), un score $S_y(t_i)$ est tout d'abord calculé, à partir duquel est déterminé un score relatif $R_y(t_i)$ (équation 2).

$$S_y(t_i) = \frac{|t_i|_y}{|t|_y} \quad , \quad R_y(t_i) = \frac{S_y(t_i)}{max(S_y(t_i))} \tag{2}$$

où :
t_i est le type $i \in \{t\}_y$,
$|t_i|_y$ est le nombre d'occurrences du type t_i dans la colonne y (i.e. le nombre d'entités possédant le type t_i),
$|t|_y$ est le nombre de types distincts dans la colonne y,
$max(S_y(t_i))$ est le score maximal obtenu par un des types i.

Seuls les types ayant un score $R_y(t_i) > 0.7$ (seuil configurable) sont considérés dans les étapes suivantes. Une méthode inspirée de TF-IDF est ensuite utilisée pour calculer la spécificité des types candidats et ainsi sélectionner les types les plus pertinents pour la colonne (étape 3 de la Figure 1). Cette méthode permet de déterminer l'importance d'un type $t_i \in \{t\}_y$

pour la colonne y (équation 3).

$$S_{spec}(t_i, y) = S_y(t_i) \times log(\frac{N_L(y)}{|t_i|_y}) \quad (3)$$

où :

$N_L(y)$ est le nombre d'entités candidates issues du processus de recherche pour la colonne y,

$|t_i|_y$ est le nombre d'occurrences du type t_i dans la colonne y (i.e. le nombre d'entités possédant le type t_i).

L'un des avantages inhérent à cette méthode est qu'elle possède une bonne indépendance vis à vis des bases de connaissances. Une alternative aurait été l'utilisation de la structure hiérarchique du graphe de connaissances pour mesurer la spécificité d'un type. Néanmoins, les performances en auraient été amoindries.

Désambiguisation d'entités. Afin d'améliorer les résultats de la tâche de CEA, les types issus de la sélection précédente sont utilisés afin de désambiguïser les entités candidates de chaque cellule (étape 4 de la Figure 1). Dans le cas où la première entité candidate possède l'un des types issus du CTA, cette entité est choisie comme résultat du CEA pour cette cellule. Dans le cas contraire, la liste des entités candidates est parcourue pour trouver une telle entité (si la liste d'entités candidates est initialement vide ou si aucune entité satisfaisante n'est trouvée, aucune annotation n'est produite pour la tâche de CEA).

Extraction des propriétés entre colonnes. La tâche de CPA a été implémentée de la même manière dans les deux approches (vote majoritaire et plongements). Elle consiste en une recherche syntaxique à partir des en-têtes de la table : les propriétés candidates dans les graphes de connaissances interrogés sont retenues. Une méthode plus élaborée a ensuite été testée sous la forme d'un algorithme récupérant l'ensemble des propriétés observées dans le graphe de connaissances entre les paires d'instances (résultats du CEA) des deux colonnes candidates et réalisant ensuite un vote majoritaire pour conserver uniquement la propriété la plus représentée entre les deux colonnes.

3.3 Méthode à base de plongements

Les méthodes à base de plongements permettent de représenter un ensemble de données textuelles (dans le cas de plongements lexicaux), ou multimodales en toute généralité, sous une forme vectorielle dans un espace de dimension fini. Bien que les dimensions de cet espace ne soient pas interprétables [2], une propriété intéressante est la proximité géométrique (vectorielle) des entités sémantiquement similaires. L'intuition sous-jacente à cette approche est que les entités d'une même colonne partagent a priori les mêmes caractéristiques sémantiques, et devraient donc se trouver géométriquement proches dans l'espace vectoriel des plongements par la nature même de ces derniers. Ainsi, les entités d'une même colonne devraient pouvoir être regroupées au sein de partitions cohérentes. Afin de comparer l'approche par plongements avec l'approche de base, une recherche syntaxique est appliquée à des plongements Wikidata en lieu et place des services de recherche externes précédemment utilisés.

Enrichissement des plongements et recherche d'entités. Les plongements Wikidata pré-entrainés (Han *et al.*, 2018) contiennent uniquement des QIDs Wikidata projetés dans un espace de dimension 200. Nous associons les labels, les alias ainsi que les types des entités aux entités en utilisant l'index Elasticsearch décrit précédemment (étape 1 du processus défini dans la Figure 2). Des opérations de recherche sont ensuite réalisées pour trouver les entités candidates (notées i dans la suite) correspondant au contenu de chaque cellule (étape 2 de la Figure 2). Le système implémente une première stratégie de recherche à base d'expressions régulières et une seconde stratégie à base de distance de Levenshtein.

1. Le label d'une entité candidate ou l'un de ses alias doit inclure tous les termes contenus dans la cellule (sans prise en compte de l'ordre).

2. Des travaux de recherche s'intéressent à la question (Senel *et al.*, 2018), (Templeton, 2020)

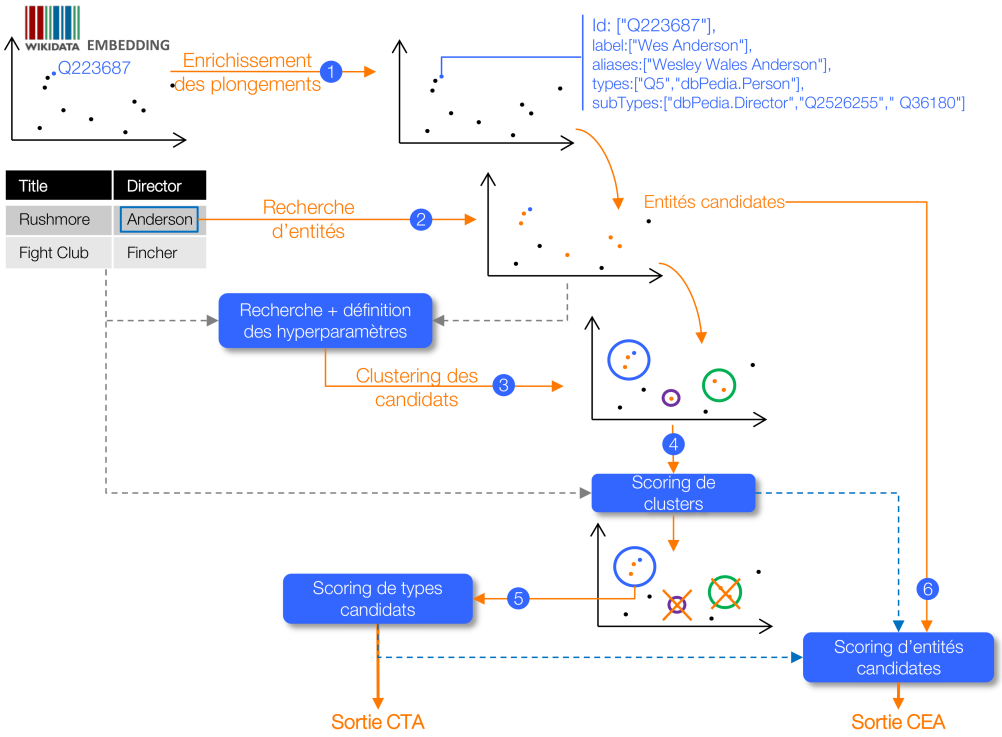

FIGURE 2 – *Description du processus utilisé dans l'approche par plongements*

2. Le ratio de Levenshtein (Sarkar *et al.*, 2016) (qui est fonction de la distance du même nom) entre le label de l'entité ou l'un de ses alias (obtenus suite au lookup) et le contenu de la cellule ($item_{sim}(i)$, où i est l'entité candidate) doit être supérieur ou égal à 0.75.

Si une entité (label ou alias) satisfait l'une des deux conditions précédentes, elle est alors ajoutée à la liste des candidates.

Partitionnement des entités candidates et choix des partitions. Après avoir choisi les candidats, le challenge est ensuite d'obtenir une partition contenant une large portion des candidats attendus pour une colonne donnée. Une stratégie de recherche Grid-search mesurant la capacité de plusieurs algorithmes et leurs hyper-paramètres à regrouper les bons candidats dans une même partition a été implémentée pour déterminer la meilleure configuration possible. Quatre algorithmes de partitionnement ont été évalués : le partitionnement en K-moyennes, le partitionnement spectral, DBSCAN et BIRCH. Cette recherche a permis d'établir que l'algorithme de partitionnement en K-moyennes (utilisant la distance euclidienne par défaut) obtient les meilleurs résultats (c'est-à-dire regroupe le plus de bons candidats d'une colonne tout en excluant les mauvais). Par ailleurs, l'analyse des résultats a permis de mettre en évidence qu'une valeur de K égale au nombre de candidats collectés par les opérations de recherche divisé par le nombre de lignes du tableau constituait une heuristique efficace permettant une automatisation du paramétrage de l'algorithme (étape 3 de la Figure 2). Une fois les partitions identifiées, l'étape suivante vise à choisir la partition représentant le mieux la colonne étudiée (étape 4 de la Figure 2). Les partitions possédant la plus importante couverture de lignes (i.e. le nombre de lignes ayant au moins un candidat dans la partition) sont retenus. Un score de confiance est ensuite associé à chaque candidat i à l'intérieur des partitions sélectionnées (équation 4).

$$S_c(i) = item_{conf}(i) \times item_{sim}(i)^w \tag{4}$$

où :

$item_{conf}(i)$ est le score de co-occurence donné par l'équation 5,

$item_{sim}(i)$ est le ratio de Levenshtein entre l'entité candidate i et la valeur de la cellule considérée,
$w \in \mathbb{N}^+$ permet de donner plus d'importance à la similarité syntaxique.

$$item_{conf}(i) = Fe(i) + 0.5 \times Fh(i) \tag{5}$$

où Fe et Fh sont définis par l'équation 6.

$$Fe(i) = \frac{e(i)+1}{N_C} \quad , \quad Fh = \frac{h(i)+1}{N_C} \tag{6}$$

où :
$e(i)$ est égal au nombre d'entités des autres colonnes de la table similaires aux valeurs de propriétés de l'entité candidate i extraites à partir du graphe de connaissance,
$h(i)$ est égal au nombre d'en-têtes des autres colonnes similaires à un label de propriété Wikidata associé à l'entité candidate i,
N_C est égal au nombre de colonnes de la table.

Le score de confiance normalisé pour une partition n donnée est ensuite calculé par l'équation 7.

$$S_k(n) = \frac{\sum_{i \in n} S_c(i)}{|n|} \tag{7}$$

où $|n|$ est le nombre d'éléments de la partition n.

A partir des candidats se trouvant dans les partitions retenues, un score est associé à chaque type rencontré parmi cette population. Le score d'un type donné est calculé en additionnant les scores de confiance de chaque individu possédant ce type (étape 5 de la Figure 2). Tous les types possédant un score supérieur à un seuil ($Max(score) * 0.75$) sont sélectionnés pour la suite du processus. Le type finalement retenu est le type le plus spécifique dans la hiérarchie DBPedia (la spécificité d'un type est calculée à partir du nombre de ses sous-classes ainsi que la distance le reliant au concept *owl:Thing*). Enfin, les entités candidates correspondant à chaque cellule sont examinées au regard du type retenu (étape 6 de la Figure 2). Le score, défini par l'équation 8, est calculé pour chaque entité candidate i appartenant à la partition n.

$$S_e(i) = w_T \times (0.2 \times S_k(n) + 0.5 \times S_c(i)) \tag{8}$$

où :
$w_T = 1.5$ si l'entité possède le type T résultant de l'étape de CTA,
$w_T = 1.2$ si l'entité a pour type un concept parent de T,
$w_T = 1$ sinon.

Pour une cellule de la table donnée, l'entité ayant le plus grand score est retenue comme résultat du CEA.

4 Résultats

Les différentes approches proposées ont été évaluées à l'aide de deux métriques sur les tâches de CEA et CPA : un score de précision et un score de F1 (équation 9).

$$Precision = \frac{|Labels\ corrects|}{|Labels\ produits|} \quad , \quad Rappel = \frac{|Labels\ corrects|}{|Labels\ cibles|} \quad , \quad F1 = \frac{2 \times P \times R}{P + R} \tag{9}$$

où $labels\ cibles$ fait référence aux cibles fournies par les organisateurs du challenge. Pour la tâche de CTA, un score primaire (AH) et un score secondaire (AP) ont été proposés par les organisateurs (équation 10)

$$AH = \frac{|P| + 0.5 \times |O| - |W|}{|T|} \quad , \quad AP = \frac{|P|}{|P| + |O| + |W|} \tag{10}$$

où P est le nombre d'annotations "parfaites" (le type correct le plus spécifique dans la hiérarchie de concepts), O est le nombre d'annotations "OK" (un concept parent de l'annotation parfaite en excluant la classe *owl:Thing*), W est le nombre d'annotations erronées et T le nombre de colonnes d'une table à annoter. Ces métriques permettent de prendre en compte la hiérarchie de classes du graphe de connaissances cible et ainsi d'évaluer la capacité des approches à produire les types les plus représentatifs mais également les plus spécifiques (et donc, porteur d'informations).

Pour des raisons de temps et de performance, l'approche par plongements a été uniquement testée sur le corpus de la première manche du challenge. Afin de permettre le calcul des scores primaires et secondaires, les données de CTA de cette manche ont été enrichies avec les types parents (excepté *owl:Thing*) du type parfait. Cette manche a permis d'évaluer les performances de l'approche par plongements par rapport à notre approche de base. Toutefois, les autres concurrents n'ont pas pu être comparés à nos approches sur cette manche car les méthodes d'évaluation du challenge ont évoluées entre la première manche et les suivantes. La Table 4 présente les scores respectifs de notre approche de base et de l'approche par plongements et montre la capacité de cette dernière à produire des types plus spécifiques. Seul notre approche de base et les résultats du meilleur concurrent (MTab (Nguyen *et al.*, 2019)) pour chaque tâche sur les trois dernières manches sont donc présentés dans la Table 5.

TABLE *4 – Comparatif de l'approche par plongements et de l'approche de base sur la première manche du challenge SemTab2019 réalisé le 30 novembre 2019 par un algorithme d'évaluation fourni par les organisateurs*

Tâche	CTA		CEA		CPA	
Critère	Prim. Score	Sec. Score	F1	Précision	F1	Précision
Baseline	0.479	0.242	**0.883**	**0.892**	0.415	0.347
Plongement	**1.212**	**0.336**	0.841	0.853	-	-

TABLE *5 – Résultats de l'approche de base et de MTab pour les trois dernières manches du challenge SemTab2019 évalués par la plateforme AICrowd le 30 novembre 2019*

	Tâche	CTA		CEA		CPA	
	Critère	Prim. Score	Sec. Score	F1	Précision	F1	Précision
Manche 2	Baseline	0.641	0.247	0.713	0.816	0.533	0.919
	MTab	1.414	0.276	0.911	0.911	0.881	0.929
Manche 3	Baseline	0.745	0.161	0.725	0.745	0.519	0.826
	MTab	1.956	0.261	0.970	0.970	0.844	0.845
Manche 4	Baseline	0.684	0.206	0.578	0.599	0.398	0.874
	MTab	2.012	0.300	0.983	0.983	0.832	0.832

Les résultats sur la tâche de CEA sont satisfaisants mais la baseline a montré des faiblesses lorsqu'il s'agissait de déduire les types attendus par l'évaluateur de la tâche CTA. Dans la première manche, le type produit par la baseline était soit trop générique, soit trop spécifique.

De plus, la baseline a montré deux limites importantes : une grande dépendance vis à vis des services de recherche (sur lesquels DAGOBAH a peu de contrôle) et des difficultés à paramétrer convenablement les algorithmes (en particulier, lorsqu'il s'agissait de trouver un bon compromis entre la spécificité du type d'une colonne et sa représentativité).

Concernant la tâche de CPA pour la première manche, l'utilisation de la méthode naïve de recherche syntaxique explique les mauvais résultats. Dans les manches suivantes, l'algorithme par vote majoritaire sur les propriétés candidates a été utilisé, ce qui a permis d'améliorer significativement la précision du CPA.

Concernant l'approche par plongement, les performances pour la tâche de CEA sont légèrement inférieures à la baseline. Cela s'explique notamment par l'utilisation de stratégies de recherche volontairement plus rudimentaires (comparativement aux méthodes de recherche très optimisées utilisées par la baseline), l'absence des bonnes entités parmi les partitions retenues ainsi que des problèmes liés aux opérations de correspondances entre Wikidata et DBPedia.

Toutefois, l'approche par plongement a montré son efficacité lorsqu'il s'agissait de déterminer le type d'une colonne (qui est une base de départ pour obtenir des annotations de CEA plus juste, en utilisant des techniques de désambiguisation). De plus, cette approche se révèle être particulièrement intéressante lorsque les cellules de la table étudiée contiennent des mentions incomplètes. Par exemple, dans la table 54719588_0_8417197176086756912 du challenge, l'une des colonnes contient des réalisateurs de films référencés uniquement par leur nom de famille (dans ce type de cas, la baseline obtient de mauvais résultats). Un partitionnement en K-moyennes sur un sous-ensemble de cette table (4 lignes uniquement) donne de très bons résultats comme présenté dans la Figure 3. En effet, la partition verte contient une grande partie des candidats attendus en résultat du CEA (une étape de désambiguisation doit malgré tout être réalisée pour trouver le bon résultat de certaines cellules en utilisant S_e).

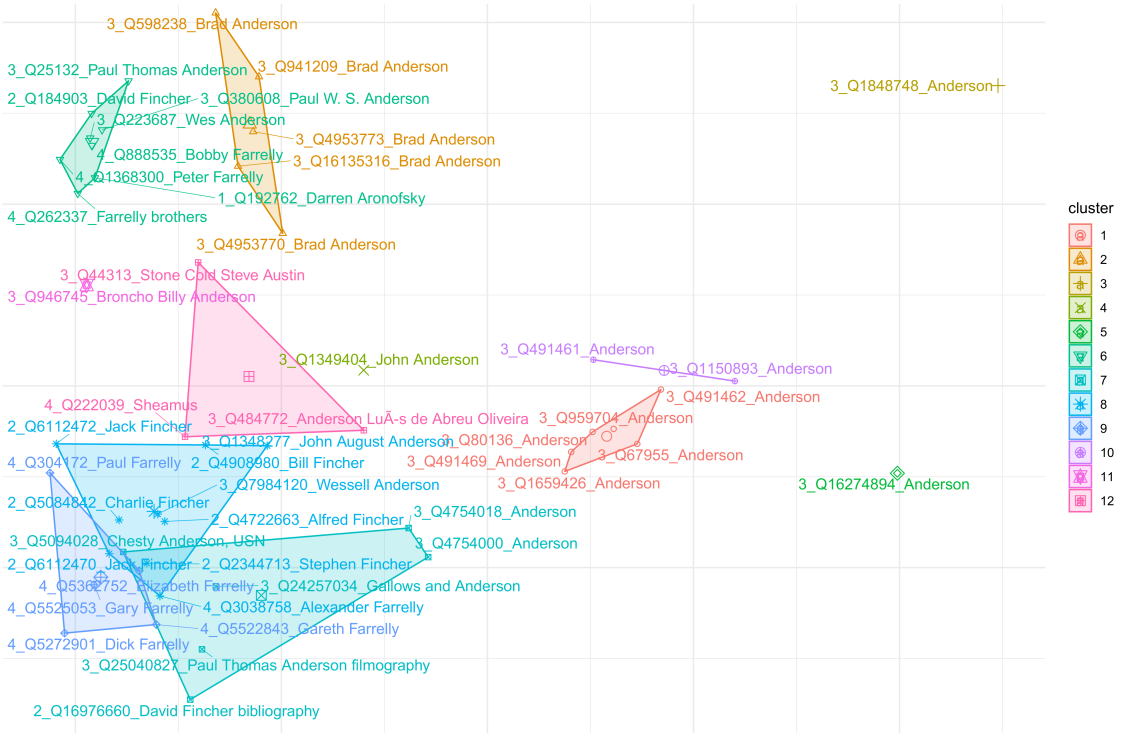

FIGURE *3 – Résultat du partitionnement en K-moyennes sur des plongements Wikidata*

5 Conclusion et travaux futurs

Nous avons présenté le système DAGOBAH, qui propose une approche de base et une approche à base de plongements de graphes pour générer des annotations sémantiques sur des données tabulaires. L'approche par plongement de graphes a montré des résultats encourageants, en particulier sur la tâche de CTA, et une réelle capacité à inférer des candidats à partir d'informations incomplètes, et ce, avec une étape de nettoyage et de préparation des données réduite au minimum. Cependant, l'optimisation des hyper-paramètres reste un problème complexe.

La méthode utilisée par DAGOBAH pour calculer le nombre de partition en sortie de l'algorithme des K-moyennes (basée sur les résultats de recherche et les propriétés de la table) donne de bons résultats mais pourrait ne pas se révéler suffisamment robuste pour l'ensemble des jeux de données (c'est la raison pour laquelle les partitions dont les scores sont les plus élevées sont conservées et pas uniquement la meilleure partition). D'autres algorithmes de partitionnement doivent être testés car ils pourraient être plus précis pour trouver le meilleur compromis entre le fait d'avoir tous les candidats dans une partition unique d'une part, et un nombre suffisant de partitions pour bien discriminer les candidats d'autre part.

La combinaison de Wikipédia et de Wikidata dans un espace de plongements commun conjointement à l'utilisation de plongements lexicaux de type fastText (Bojanowski *et al.*, 2016), ainsi que l'exploitation de la recherche sémantique en plus de la recherche syntaxique, pourraient améliorer considérablement le processus d'annotation sémantique. Enfin, une piste prometteuse est d'avoir recours à une approche complètement vectorielle (Chen *et al.*, 2018) consistant à apprendre des plongements à partir des lignes de la table (de type Poincaré (Nickel & Kiela, 2017) afin de capturer les hiérarchies latentes) combinée à des contraintes géométriques déduites de la structure de la table. Il s'agirait ensuite de trouver des transformations (globales ou locales) entre cet espace vectoriel et un espace de plongements Wikidata/Wikipédia afin d'améliorer la qualité des annotations.

Références

BHAGAVATULA C. S., NORASET T. & DOWNEY D. (2015). TabEL : Entity linking in web tables. In *14th International Semantic Web Conference (ISWC)*, p. 425–441.

BOJANOWSKI P., GRAVE E., JOULIN A. & MIKOLOV T. (2016). Enriching word vectors with subword information. *arXiv preprint arXiv :1607.04606*.

CHABOT Y., GROHAN P., LE CALVEZ G. & TARNEC C. (2019a). Dataforum : Faciliter l'échange, la découverte et la valorisation des données à l'aide de technologies sémantiques. In *Extraction et Gestion des Connaissances (EGC)*, Metz, France.

CHABOT Y., LABBÉ T., LIU J. & TRONCY R. (2019b). DAGOBAH : An End-to-End Context-Free Tabular Data Semantic Annotation System. *SemTab2019, ISWC Challenge*.

CHAPMAN A., SIMPERL E., KOESTEN L., KONSTANTINIDIS G., IBÁÑEZ L.-D., KACPRZAK E. & GROTH P. (2019). Dataset search : a survey. *The VLDB Journal*, p. 1–22.

CHEN J., JIMENEZ-RUIZ E., HORROCKS I. & SUTTON C. (2018). ColNet : Embedding the Semantics of Web Tables for Column Type Prediction. In *33rd AAAI International Conference on Artificial Intelligence*.

CREMASCHI M., AVOGADRO R. & CHIEREGATO D. (2019). Mantistable : an automatic approach for the semantic table interpretation. *SemTab2019, ISWC Challenge*.

EBERIUS J., BRAUNSCHWEIG K., HENTSCH M., THIELE M., AHMADOV A. & LEHNER W. (2015). Building the dresden web table corpus : A classification approach. In *2015 IEEE/ACM 2nd International Symposium on Big Data Computing (BDC)*, p. 41–50 : IEEE.

EFTHYMIOU V., HASSANZADEH O., RODRIGUEZ-MURO M. & CHRISTOPHIDES V. (2017). Matching web tables with knowledge base entities : From entity lookups to entity embeddings. In *16th International Semantic Web Conference (ISWC)*, p. 260–277.

FÄRBER M., ELL B., MENNE C. & RETTINGER A. (2015). A Comparative Survey of DBPedia, Freebase, OpenCyc, Wikidata, and YAGO. *Semantic Web Journal*, p. 1–5.

FERNANDEZ R. C., MANSOUR E., QAHTAN A. A., ELMAGARMID A., ILYAS I., MADDEN S., OUZZANI M., STONEBRAKER M. & TANG N. (2018). Seeping semantics : Linking datasets

using word embeddings for data discovery. In *34th International Conference on Data Engineering (ICDE)*, p. 989–1000.

HAN X., CAO S., LV X., LIN Y., LIU Z., SUN M. & LI J. (2018). Openke : An open toolkit for knowledge embedding. In *International Conference on Empirical Methods in Natural Language Processing (EMNLP)*, p. 139–144.

HASSANZADEH O., EFTHYMIOU V., CHEN J., JIMÉNEZ-RUIZ E. & SRINIVAS K. (2019). SemTab2019 : Semantic Web Challenge on Tabular Data to Knowledge Graph Matching - Data Sets. Zenodo.

IBRAHIM Y., RIEDEWALD M. & WEIKUM G. (2016). Making sense of entities and quantities in Web tables. In *International Conference on Information and Knowledge Management*, p. 1703–1712.

JIMÉNEZ-RUIZ E., HASSANZADEH O., EFTHYMIOU V., CHEN J. & SRINIVAS K. (2020). SemTab 2019 : Resources to Benchmark Tabular Data to Knowledge Graph Matching Systems. In *17th European Semantic Web Conference (ESWC)*.

KILIAS T., LÖSER A., GERS F. A., KOOPMANSCHAP R., ZHANG Y. & KERSTEN M. (2018). Idel : In-database entity linking with neural embeddings. *arXiv preprint arXiv :1803.04884*.

LEHMBERG O., RITZE D., MEUSEL R. & BIZER C. (2016). A Large Public Corpus of Web Tables containing Time and Context Metadata. In *25th International Conference Companion on World Wide Web (WWW Companion)*, p. 75–76.

LIMAYE G., SARAWAGI S. & CHAKRABARTI S. (2010). Annotating and searching web tables using entities, types and relationships. In *36th International Conference on Very Large Data Bases (VLDB)*, p. 1338–1347.

MORIKAWA H. (2019). Semantic table interpretation using lod4all. *SemTab2019, ISWC Challenge*.

MULWAD V., FININ T., SYED Z. & JOSHI A. (2010). Using linked data to interpret tables. In *1st International Workshop on Consuming Linked Data (COLD)*.

NGUYEN P., KERTKEIDKACHORN N., ICHISE R. & TAKEDA H. (2019). Mtab : Matching tabular data to knowledge graph using probability models. *arXiv preprint arXiv :1910.00246*.

NICKEL M. & KIELA D. (2017). Poincaré embeddings for learning hierarchical representations. *ArXiv*.

OLIVEIRA D. & D'AQUIN M. (2019). Adog - annotating data with ontologies and graphs. *SemTab2019, ISWC Challenge*.

RAN C., SHEN W., WANG J. & ZHU X. (2016). Domain-specific knowledge base enrichment using wikipedia tables. In *IEEE International Conference on Data Mining (ICDM)*, p. 349–358.

RITZE D., LEHMBERG O. & BIZER C. (2015). Matching HTML Tables to DBpedia. In *5th International Conference on Web Intelligence, Mining and Semantics (WIMS)*, p. 1–6.

SARKAR S., PAKRAY P., DAS D. & GELBUKH A. (2016). JUNITMZ at SemEval-2016 Task 1 : Identifying semantic similarity using levenshtein ratio. In *10th International Workshop on Semantic Evaluation (SemEval)*, p. 702–705.

SENEL L. K., UTLU I., CSAHINUCC F., OZAKTAS H. M. & KOCC A. (2018). Imparting interpretability to word embeddings while preserving semantic structure. *arXiv : Computation and Language*.

STEENWINCKEL B., VANDEWIELE G., DE TURCK F. & ONGENAE F. (2019). Csv2kg : Transforming tabular data into semantic knowledge. *SemTab2019, ISWC Challenge*.

TEMPLETON A. (2020). Inherently interpretable sparse word embeddings through sparse coding. *ArXiv*.

THAWANI A., HU M., HU E., ZAFAR H., DIVVALA N. T., SINGH A., QASEMI E., SZEKELY P. & PUJARA J. (2019). Entity linking to knowledge graphs to infer column types and properties. *SemTab2019, ISWC Challenge*.

ZHANG Z. (2014). Towards efficient and effective semantic table interpretation. In *13th International Semantic Web Conference (ISWC)*, p. 487–502.