MINES PARISTECH

RAPPORT DE STAGE

Vers une architecture unifiée d'intégration et extraction des données journalistiques

Auteur :
Hugo CISNEROS

Encadrants :
Xavier TANNIER
Ioana MANOLESCU

24 septembre 2018

MINES PARISTECH

Résumé

Option Management des Systèmes d'information

Rapport de stage

Vers une architecture unifiée d'intégration et extraction des données journalistiques

par Hugo CISNEROS

Une partie du travail de journaliste consiste à enquêter, vérifier les faits, les expliquer tout en respectant certains principes éthiques d'exactitude, d'impartialité et aussi de responsabilité. L'accessibilité d'une grande quantité de données à travers plusieurs nouveaux médias ainsi que la prolifération des informations douteuses ou fausses rendent cette tâche d'autant plus difficile. En parallèle, de récents progrès en bases de données, analyses de données et apprentissage automatique ont rendu possible la construction semi automatisée de grandes bases de connaissances à partir de sources de données diverses (articles, réseaux sociaux, page web, etc.). Ces techniques permettraient aux journaliste de disposer de réservoirs de connaissances dans lesquels puiser pour comprendre certains évènements de manière détaillée.

Le travail présenté ici consiste en une proposition d'architecture pour permettre aux journalistes d'accéder plus simplement à ces données hétérogènes par nature et d'en extraire des informations utiles. Ceci peut être accompli en liant des éléments les uns avec les autres ou en enrichissant certains éléments avec des informations supplémentaires. Dans le cadre de ce projet, une implémentation d'un système de compilation depuis un fichier de spécification vers une implémentation réelle de l'architecture est proposée.

Remerciements

Je tiens à remercier toutes les personnes qui ont contribué au succès de mon stage et qui m'ont aidé lors de la rédaction de ce rapport.

Tout d'abord, j'adresse mes remerciements à mon encadrante de stage Mme Ioana Manolescu, chef de l'équipe CEDAR de l'INRIA pour son accueil, le partage de son expertise et son temps. Grâce à ses conseils, j'ai pu me sortir des moments délicats.

Je remercie aussi vivement mon autre encadrant M. Xavier Tannier, Professeur à Sorbonne Université et Chercheur au LIMICS. Il m'a accueilli dans les locaux de son laboratoire et a passé beaucoup de temps à me conseiller et m'aider à avancer grâce à ses connaissances précises dans le domaine de ce stage.

Table des matières

Table des figures

1 Contexte

Le travail présenté dans ce rapport a été mené à bien pendant un stage de recherche effectué dans le cadre de l'option Management des systèmes d'information de Mines ParisTech. Cette partie présentera d'abord le journalisme de données, puis le cadre scientifique du projet présenté ainsi que le cadre dans lequel il s'est déroulé. Enfin, de notions concernant le Web Sémantique sont présentées, notamment avec des définitions de concepts qui sont réutilisés dans le texte.

1.1 Le journalisme de données

Journalisme de données est un terme récent étant apparu pour décrire un processus journalistique qui consiste à amasser et traiter une grande quantité de données pour appuyer ou créer une enquête ou une analyse. La bonne utilisation de ces données permet de véhiculer plus aisément un message complexe ou difficile à déceler avec les méthodes classiques. La démarche du journalisme de données peut être divisé en quatre grandes étapes qui sont nécessaires à l'exploitation de données :

1. Extraction et collection des données
2. Filtrage, nettoyage
3. Mise en place de visualisations pertinentes
4. Analyse de ces résultats

En plus de ces éléments techniques, il se rattache aussi fortement à des mouvements de la communauté des informaticiens comme celui de des logiciels libres et à source ouverte. L'ouverture de l'accès aux articles et aux données nécessaires à sa création est aussi très importante, car les sources doivent rester transparentes et vérifiables. Ces données peuvent par exemple venir de sources officielles ou être le fruit de fuites de la part d'organisations tierces.

Sont présentés ci-après les possibles cas d'applications du journalisme de données, ainsi que le problème essentiel et fortement lié au travail présenté dans ce rapport : la vérification de faits en journalisme dans le contexte actuel.

1.1.1 Utilisation des données en journalisme

Les données peuvent être la source d'une enquête comme ont pu le montrer par exemple les nombreux travaux d'investigations d'organisations comme le Consortium International des journalistes d'investigation (*International Consortium of Investigative Journalists* 2018, ICIJ). Celui-ci a notamment eu une portée importante grâce à des travaux comme leurs analyses des Panama Papers ou des Paradise Papers. Dans ces deux cas, les données étaient constituées d'une très grande quantité de fichiers dans des formats divers, scannés ou non, et une partie importante du travail de ces journalistes a été d'extraire de l'information utilisable de cette masse non structurée. Après extraction, ces informations ont été intégrées à une base de données basée sur une structure de graphe et permet d'obtenir des visualisations comme celles présentées Figures 1.1 et 1.2.

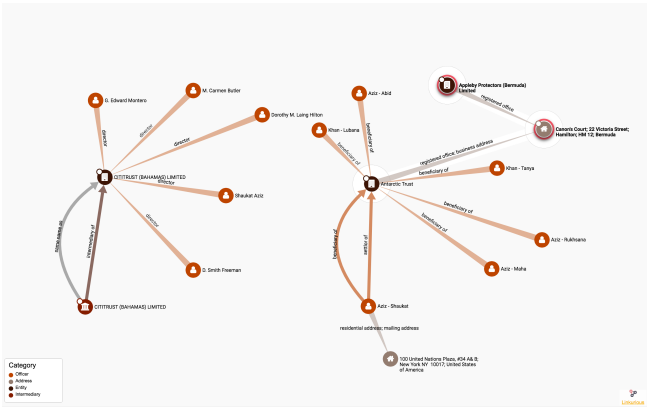

FIGURE 1.1 – Un exemple de graphe issu de l'investigation effectuée par l'ICIJ sur les Paradise Papers à propos des relations de Shaukat Aziz, ancien premier ministre du Pakistan.

Extrait le 07/06/2018 de https://offshoreleaks.icij.org/stories/shaukat-aziz. Copyright 2018 — The International Consortium of Investigative Journalists

Dans d'autres cas les données ne représentent qu'un appui, ou une illustration qui peut permettre soit d'enrichir le message initial, soit de le rendre plus facile à appréhender. Le site web Information is Beautiful est un exemple de publication qui se base sur la visualisation pour véhiculer de l'information. Sur l'exemple Figure 1.2 extrait de ce site, les auteurs invitent leurs lecteurs à reconsidérer les grands scandales médiatiques récurrents comme celui des jeux vidéos violents en représentant la distribution des fréquences de publications d'articles de presse à leur sujet.

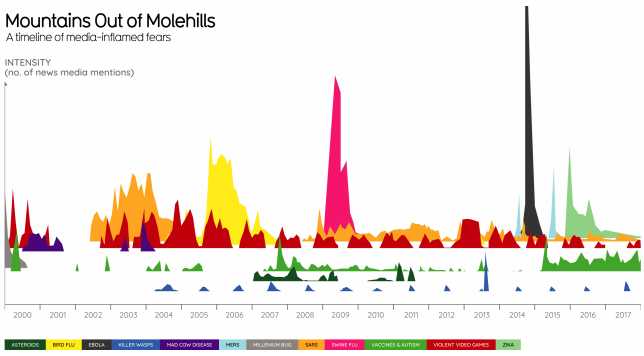

FIGURE 1.2 – Une visualisation du nombre de mentions de différents évènements et/ou sujets dans les médias par David McCandless (Information is Beautiful)

Extrait le 07/06/2018 de https://informationisbeautiful.net/visualizations/mountains-out-of-molehills/. Copyright 2018 — Information is beautiful

Le journalisme de données est une extension du journalisme, et s'est imposé grâce à l'abondance de l'information et des données qu'a apporté l'ère numérique. Ceci est montré par les nombreuses initiatives de journaux de créer et publier du contenu basé sur ou appuyé par des données et qui jouissent d'une visibilité importante comme Les Décodeurs (affiliés à *Le Monde*) ou Désintox (par *Libération*) en France. Ces nouvelles techniques impliquent de nouvelles compétences, comme la programmation pour automatiser l'extraction et la mise en forme des données, la familiarité avec les bases de données pour le stockage, ou encore la programmation web pour la présentation et la création d'outils de visualisation. Par conséquent, de nombreux outils furent développés pour offrir une interface simplifiée à l'interaction

avec les données aux personnes n'étant pas familières avec les systèmes classiques de traitement de données, ou n'ayant pas les compétences en informatique nécessaires à leur utilisation.

1.1.2 Le problème de la vérification de faits

La verification de faits ou *fact-checking* consiste à étudier l'exactitude de faits déclarés, souvent par des personnages publiques (politiciens, dirigeants d'entreprise, etc.). Cette activité est et a toujours été la tâche principale du journaliste, que ce soit pendant l'écriture d'un article ou pendant la validation préalable à une publication. La vérification est nécessaire pour assurer la réputation d'un média supposé respecter la déontologie du journaliste d'abord, et aussi pour se prévenir de potentielles actions légales à son encontre.

Bien que cette notion soit intimement liée à l'activité de journaliste, elle a connu de profondes modifications, ceci notamment avec le développement d'internet ou d'autres technologies récentes. Parmi celles-ci, certaines ont eu un impact particulièrement important :

— L'échelle à laquelle la vérification des faits doit désormais se faire a explosé. C'est à cause des nombreux points de dissémination de l'information qu'internet a permis de créer, ainsi qu'au journalisme collaboratif parfois appelé *citizen journalism* (GOODE, 2009) que la quantité de faits à vérifier. ainsi que la complexité des choses à vérifier est devenu particulièrement importante.

— La transparence et la visibilité du processus de vérification a mis cette activité au premier plan du travail de journaliste alors qu'elle restait un travail fait en amont de la publication et l'écriture au sein du média lui-même. Les conclusions ne suffisent pas à convaincre, puisque le public cherche souvent à connaître les étapes et les sources des informations utilisées.

— Enfin, la complexité de la vérification des faits a beaucoup augmenté, notamment avec la multiplication des sources possibles d'informations, aux fiabilités variables. Certaines sources institutionnelles sont particulièrement fiables (INSEE ou Eurostat pour l'UE), mais certaines autres ne sont valables que sur un laps de temps limité, existent sous des formes difficilement exploitables, etc. De plus, il est crucial de prendre en compte la complexité d'un fait lui-même, qui est rarement uniquement vrai ou faux, mais bien plus souvent partiellement vrai, représentant de manière erronée une information vraie, ou effectuant un mauvaise interprétation d'un chiffre, etc.

Le dernier point a de nombreuses implications pour la vérification des faits pour un journaliste. La vérification peut varier dans :

— Ses objectifs : détecter la fraude, la manipulation, vérifier des promesses de campagne, ou lutter contre la propagande.

— Son système de notation : « vrai », « globalement faux », « douteux », les codes couleurs, « c'est plus compliqué », etc.

— Les sources : bases de données commerciales, internes, sources publiques et ouvertes.

— Les types de faits qui sont vérifiés : la précision ou validité de chiffres, la rigueur logique d'un argument, l'existence d'une citation, mensonges, inconsistences dans un groupe d'information, erreurs ou omissions. Parfois, l'information a juste besoin d'être mise en contexte.

1.2 Le projet ANR ContentCheck

Le projet sur lequel j'ai travaillé s'inscrit dans le projet ANR ContentCheck ayant le nom complet *Content Management Techniques for Fact-Checking : Models, Algorithms, and Tools*. Il a démarré en janvier 2016 et doit durer jusqu'en 2020. Le but du projet est d'établir une description scientifique globale de la vérification de faits comme un problème de management de contenu, de mettre au point de modèles innovants basés sur des fondations formelles solides, de mettre au point, vérifier et valider de nouveaux algorithmes pour la vérification automatique de faits et construire un corpus de référence pour la vérification de faits.

Son originalité est qu'il est une des premières tentatives de grouper ensemble sur un projet de long terme des journalistes et chercheurs en informatiques, et plus particulièrement en manipulation de données et traitement automatique du langage, afin de tenter de comprendre et étudier le problème de la vérification de faits. Les partenaires du projet sont :

— L'INRIA avec l'équipe-projet OAK, puis CEDAR
— L'Université Rennes 1
— Le Monde
— Le LIMSI
— L'INSA Lyon (laboratoire LIRIS)

La présence d'une équipe de journalistes de Le Monde est un atout car cela permet d'ancrer les projets dans des problématiques concrètes et de valider les proposition de solutions. J'ai pu assister et participer à une réunion regroupant la plupart des membres du projet, notamment les journalistes du Monde.

1.3 Cadre

Le stage est supervisé par Ioana Manolescu, chef de l'équipe CEDAR : *Rich Data Analytics at Cloud Scale*, et Xavier Tannier, Professeur à Sorbonne Université et chercheur au LIMICS (Laboratoire d'Informatique Médicale et d'Ingénierie des Connaissances en e-Santé), anciennement chercheur au LIMSI. Les deux laboratoires où j'ai eu l'occasion de travailler se trouvent sur le campus des Cordeliers de Sorbonne Université et dans le bâtiment Alan Turing qui est partagé entre le Laboratoire d'Informatique de l'école Polytechnique (LIX) et l'INRIA.

L'équipe CEDAR est un groupe-projet de l'INRIA et du LIX qui effectue des recherches dans deux domaines principaux :

— L'exploitation des infrastructures parallèles de traitement de données pour le développement de systèmes de stockage et traitement de Big Data.
— La création de nouveaux paradigmes d'interaction des utilisateurs avec les Big Data.

Xavier Tannier est spécialisé dans le traitement automatique du langage naturel et l'extraction d'information sur lesquels il travaillait au LIMSI, puis sur lesquels il travaille au LIMICS dans le domaine de l'application de ces techniques à la santé.

L'éloignement géographique des deux sites m'a conduit à passer plus de temps sur le campus des Cordeliers à Paris qu'au laboratoire à Saclay mais j'ai essayé de maintenir au moins un déplacement par semaine à ce dernier, notamment parce qu'un des mes encadrants est présent sur place, mais aussi pour profiter de l'environnement scientifique que cela procure. Le double encadrement a notamment causé quelques problèmes organisationnels par moments, surtout car il peut être difficile

de croiser trois emplois du temps différents mais cela n'a pas eu d'impact sur le bon déroulement du stage.

Un avantage important d'avoir deux encadrants, surtout quand ils ont des sujets d'expertise différents, est qu'ils peuvent avoir chacun leur vision sur le projet et les difficultés rencontrées et que celles-ci peuvent être complémentaires et rendre le travail plus dynamique.

Le travail que j'ai effectué ne correspondait pas à un cahier des charges, ni à un objectif définit précisément, ce qui a donné lieu au début du stage à une phase importante d'exploration de l'état de l'art et de la littérature existante portant sur des sujets similaires au mien (voir 2). Ce travail de familiarisation et de compréhension avec des travaux précédents m'a permis non seulement de gagner beaucoup de connaissances dans le domaine rapidement, mais aussi de prendre le recul nécessaire à la formulation d'un problème concernant mon sujet et la mise en place des moyens de le résoudre.

1.4 Web Sémantique

Le Web Sémantique désigne une extension du World Wide Web (WWW) établie par le Consortium du World Wide Web (W3C) principalement via des standards comme le Ressource Description Framework (RDF). Cette notion a été popularisée par Tim-Berners Lee dans son ouvrage Weaving the Web (BERNERS-LEE, 1999).

Le standard RDF fut à l'origine inventé pour modéliser les métadonnées de documents mais il est progressivement devenu une méthode de description et de modélisation bien plus général et utilisé dans un grand nombres de ressources web. Son but est de faciliter l'interconnection des données entre de multiples sources.

Deux concept qui furent beaucoup développé dans le cadre du Web Sémantique et qui sont fortement liés entre eux sont les bases de connaissances et les graphes de connaissances.

1.4.1 Bases de connaissances

De la manière la plus générale, une base de connaissances désigne une technologie qui permet de stocker un ensemble de données structurées et non structurées dans un système informatique. Cette notion peut s'appliquer à plusieurs systèmes différents, comme une base de données relationnelle ou une ontologie. Les ontologies sont un terme regroupant une convention de nommage, la définition de catégories, propriétés et relations d'une collection d'entités et de données dans un ou plusieurs domaines particuliers.

1.4.2 Graphes de connaissances

Le terme graphe de connaissances (GC) existe aussi et désigne selon (EHRLINGER et WÖSS, 2016) un *système qui intègre de l'information dans une ontologie et applique un raisonneur pour en faire dériver de nouvelles connaissances*. Un GC peut aussi être vu comme un graphe RDF (voir 1.4 pour la définition de RDF) , c'est à dire un ensemble de triplets RDF (s, p, o), eux-mêmes des ensembles ordonnés des termes RDF suivants :

1. $s \in U \cup B$ un sujet.
2. $p \in U$ un prédicat.
3. $o \in U \cup B \cup L$ un objet.

Les ensembles ci-dessus étant U les URI, B les noeuds vierges, L les littéraux. Un GC a comme son nom l'indique une structure de graphe, qui peut être directionnel ou non et peut avoir un nombre arbitraire de types de liens. Cette structure relativement flexible permet de modéliser des données complexes comprenant de nombreux types d'informations, d'entités et de relations. Certains systèmes utilisant des GC seront décrits dans 2.

2 Travaux Connexes

Le travail présenté dans ce rapport est en lien avec de nombreux autres domaines de recherche actifs. Les différents problèmes envisagés sont difficiles et peuvent donner naissance à des approches diverses. Seront présentés ici l'intégration d'information, qui consiste à regrouper sous un schéma commun des données issues de plusieurs sources qui peuvent être très différentes. Ensuite, une manière de grouper et de rendre exploitable les données très utilisée est la base de connaissances (voir 1.4). Beaucoup de méthodes ont été proposées pour les construire, allant d'approches manuelles à d'autres complètement automatique. Avec plusieurs bases de connaissances déjà existantes, une manière d'en constituer une plus importante consiste à les aligner pour pouvoir les fusionner.

2.1 Intégration d'information

Depuis le développement des bases de données sur des sujets variés, le problème de leur utilisation intégrée sous une unique interface, à laquelle pourraient s'adresser toute les requêtes des utilisateurs, a été étudié à partir de l'article fondateur WIEDERHOLD, 1992. Un système d'intégration de données peut accomplir cette tâche de deux façons :

— En chargeant et intégrant toutes les données dans une base unique, qui sera la seule utilisée ; c'est l'approche dite "entrepôt de données" JARKE, 2003 ;

— En gardant les données dans les différentes bases de données sous-jacentes et en traitant des requêtes sur l'ensemble des bases en distribuant le travail d'évaluation entre ces systèmes et une couche supplémentaire qui assure l'intégration, appelée médiateur. Chaque source est alors interfacée avec le médiateur par une composante logicielle adaptée aux capacités (p.ex. langage de requête) de chaque source, appelé wrapper (ou adaptateur).

Dans le contexte entrepôt, des questions étudiées portent surtout sur l'analyse des données par des requêtes d'aggrégation, (p.ex. analyse des ventes par type de produit et mois de l'année, etc.).

Dans un médiateur, les questions étudiées concernent la construction des adaptateurs, la description de leurs capacités de traitement de requêtes (afin que le médiateur sache quelles parties d'une requête il peut leur déléguer TOMASIC, RASCHID et VALDURIEZ, 1998) et aussi l'optimisation de requêtes, avec par exemple IVES et al., 1999. Cette dernière tâche est complexifiée par la nécessité de s'appuyer sur différentes sources (bases de données sous-jacentes) dont on ne contrôle pas la charge ni le temps de réponse.

Plus récemment, I. Manolescu, X. Tannier et des collègues ont développé Tatooine (BONAQUE et al., 2016), un système d'intégration de données hétérogènes conçu aussi dans le but de faciliter l'exploration de données pour les journalistes. Toutefois, Tatooine ne fournit que la capacité d'évaluer des requêtes en croisant et en raffinant les résultats extraits à partir de différentes sources ; s'en servir nécessite

de comprendre des concepts typiquement confinés à l'intérieur d'un serveur de gestion de bases de données telles que jointure, sélection etc. et donc il n'a pas atteint son but du point de vue de l'usabilité. Le travail décrit dans ce rapport vise à

— Combler ce manque en reposant le problème à partir du point de vue de l'utilisateur et en lui permettant de spécifier plus facilement des traitements complexes ;
— Mieux intégrer les tâches d'extraction d'entités et relations dans l'architecture, justifié par le rôle important qu'elles jouent dans les applications que nous avons étudiées. Cette importance est dûe à la préponderence des données textuelles (ou quasi textuelles, p.ex. JSON).

2.2 Construction de base de connaissances

La construction de base de connaissances peut être faite de multiples manières. Parmi celles-ci, les méthodes automatique sont récemment devenues un sujet de recherche important. Celui-ci s'est ensuite divisé en deux grandes disciplines : l'extraction avec schéma cible (ou *slot-filling*) et l'extraction sans schéma cible (aussi appelée *open information extraction*).

2.2.1 Différentes approches pour la construction de bases de connaissances

(NICKEL, MURPHY et al., 2016) classifie les approches pour la construction de bases de connaissances en quatre catégories :

— Systèmes basés sur la connaissance d'experts.
— Systèmes basés sur la collaboration d'un communauté.
— L'extraction automatique sur des données semi-structurées.
— L'extraction automatique sur des données non structurées, elle même divisée en deux types.
  — Avec schéma cible
  — Sans schéma cible

Les trois premières approches seront abordées dans cette section et la dernière fera l'objet d'une étude plus détaillée dans 2.2.2 et 2.2.3.

Dans le cas de l'extraction automatique à partir de données non structurées, la tâche peut être décrite de plusieurs manières différentes qui correspondent à des contextes légèrement différents.

— La création automatique de base de connaissances à partir de pur texte. Approche par exemple utilisée dans (CARLSON, BETTERIDGE et KISIEL, 2010).
— La prédiction de nouveaux liens ou entités dans une base de connaissance existante mais incomplète ou inexacte.

D'autres tâches sont aussi étudiées dans la littérature comme la résolution d'entités ou le groupement d'entités basé sur les liens de la base de connaissance, aussi appelée détection de communauté dans le domaine de l'analyse de réseaux sociaux.

Dans une base de connaissance représentée par un graphe, un triplet représente une information considérée comme vraie. Cependant, il existe plusieurs manières de considérer les triplets n'existant pas dans cette base.

***Closed World Assumption* (CWA)** les triplets n'étant pas dans la base sont considérés comme automatiquement faux. C'est à dire que si deux entités n'ont pas de lien entre elles, elles n'ont pas de relation entre elles.

***Open World Assumption* (OWA)** un triplet non existant est inconnu, et peut donc être soit vrai soit faux. Cette hypothèse correspond plus aux bases de connaissances à portée universelle comme Freebase ou Wikidata où la quantité de triplets possibles est si grande qu'on ne peut espérer les avoir tous dans la base.

Une variante de la première hypothèse est la *Local Closed World Assumption* (LCWA) qui postule que la base de connaissances est seulement localement complète, c'est à dire que pour une entité e et une relation r et au moins un triplet $(e, r, \cdot)$ observé, alors tous les autres possibles triplets $(e, r, \cdot)$ sont faux. Cette hypothèse, plus réaliste que la CWA [1] , est par exemple utilisée dans (GALÁRRAGA et al., 2013 ; X. DONG et al., 2014) car elle permet d'obtenir aisément des exemples négatifs dans le cas d'une classification binaire supervisée.

Systèmes basés sur la connaissance d'experts

Ce type de système est basé sur les connaissances et l'ajout manuel d'éléments d'informations par des experts du domaine ciblé (p. ex. l'ajout de triplets dans un graphe).

WordNet, une base de données lexicales de la langue anglaise, où les mots sont groupés par synonymes exprimant différents concepts (appelés *synsets*) est un exemple de ce genre de système. Les mots y sont séparés entre formes littérales apparaissant dans du texte et le sens de ceux-ci. Plusieurs sens forment donc plusieurs synsets, et des relations d'hyperonymie et d'hyponymie sont répertoriées entre les synsets.

UMLS (*Unified Medical Language System*) est une base de concepts médicaux avec leurs relations et leur hiérarchie qui permet de d'avoir rapidement accès à une très grande quantité d'informations médicales. Elle est aussi basée sur les contributions d'experts qui remplissent manuellement la base.

Ce type de base a l'avantage d'être composé de contenu de haute qualité car contrôlé finement par les experts qui y contribuent. En revanche, elles souffrent pour la même raison de gros problèmes pour passer à l'échelles et nécessitent des ressources importante pour atteindre des tailles importantes.

Systèmes basés sur la collaboration d'un communauté

D'autres systèmes, de manière similaire au fonctionnement d'outils collaboratifs comme Wikipedia, se servent des contributions d'une communauté pour constituer la base de connaissances.

Le désormais abandonné projet Freebase (BOLLACKER et al., 2008) était basé sur ce fonctionnement, tout comme son successeur Wikidata (IC et KRÖTZSCH, 2014) par la suite. Ce dernier a de plus pu profiter du contenu accumulé sur Freebase grâce à des mappings qui ont permis d'exporter des données d'une base à l'autre.

Comme le montrent Wikipedia et les bases de connaissances mentionnées ci-dessus, les contributions d'une communauté active peuvent permettre de grandement mitiger les faiblesses des systèmes utilisant des experts d'un point de vue de la capacité à passer à l'échelle, au détriment de la qualité du contenu qui nécessite d'être correctement sourcé et vérifié pour être exploitable.

1. Selon (MIN et al., p.d.), en Octobre 2013 la base Freebase n'avait pas de lieu de naissance pour 93.8% des personnes et pas de nationalité pour 78.5%.

Extraction automatique sur données semi-structurées

Pour permettre d'allier une qualité de contenu suffisante avec un passage à l'échelle non envisageable pour des humains (bases de connaissances particulièrement grandes, dans un domaine très spécifique ou ayant une structure très complexe), des méthodes automatiques de création de bases de connaissances ont été mises au point.

YAGO (Yet Another Great Ontology, Fabian M SUCHANEK, KASNECI et WEIKUM, 2007) est un système dont la première version a été présentée en 2007 et qui se base sur WordNet et Wikipedia pour constituer une base de connaissances de manière automatique à partir de données semi-structurées. Dans la première version de YAGO, les individus sont extraits de pages Wikipedia et du système de catégorisation de Wikipedia, visible en bas des pages et dans les *infoboxes*, puis les synsets de WordNet sont utilisés pour créer les relations entre ces entités et peupler leur ontologie dont la structure est spécifié dans l'article. Les versions suivantes du système ont étendu et affiné l'extraction et la structure de l'ontologie, intégrant de nouveaux éléments.

Une approche similaire est utilisée par DBPedia (AUER et al., 2007), où la structure de certaines parties des pages Wikipedia est exploitée pour extraire des informations structurées. Ces informations sont :

— Les informations sur les pages déjà stockées sous forme relationnelle par le logiciel MediaWiki sur lequel Wikipedia repose.
— Les textes et liens présents dans les infoboxes des pages qui suivent un schéma relativement consistant pour les pages de la même catégorie.

2.2.2 Extraction automatique sur données non structurées

Il existe de nombreuses techniques pour extraire de manière automatique des données depuis une source non structurée vers un schéma fixé, pour beaucoup basées sur l'apprentissage relationnel statistique, c'est à dire la création de modèles statistique à partir de données relationnelles. Celles-ci, comme mentionnées dans 2.2.1 permettent d'accomplir un certain nombre de tâches distinctes.

Ces techniques d'apprentissages peuvent être classifiées en différents types :

— Les méthodes basées sur des variables latentes associées aux entités et relations du graph (TROUILLON et al., 2016 ; BORDES et al., 2013 ; LIN et al., 2017 ; NICKEL, TRESP et KRIEGEL, 2011).
— Le méthodes uniquement basées sur des métriques et calculs sur le graphe lui-même (p. ex. le *path-ranking algorithm*) (WANG et al., 2016).
— Les méthodes basées sur des modèles graphiques, souvent une combinaison de la logique du premier ordre et des modèles graphiques de Markov (JIANG, LOWD et DOU, 2012 ; PUJARA et al., 2013 ; ZHANG et al., 2017).

Ces méthodes se basent sur un ensemble initial et fixé de relations et classes pour les entités. Ce schéma préalable peut être défini par l'application visé de la base de connaissance ou être pris comme un sous-ensemble des classes et relations de bases de connaissance existantes (comme Freebase, Wikidata ou DBPedia). Ces mêmes bases de connaissances sont ensuite souvent utilisées comme référence pour évaluer la performances des outils proposés à travers la littérature. La qualité des méthodes d'extraction à proprement parler peut, par exemple dans le cas d'un graphe où les éléments sont des triplets, être évaluée à partir de scores classique de classification comme la précision, le rappel, etc.

Certains systèmes complets sont exposés plus bas afin d'illustrer comment les méthodes ci-dessus peuvent permettre d'effectivement extraire et s'assurer de la qualité de l'information extraite de sources non structurées de données.

DeepDive - un système basé sur les méthodes graphiques

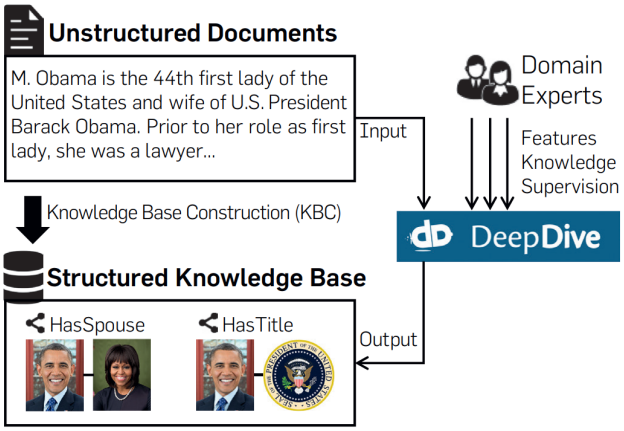

FIGURE 2.1 – Description superficielle du fonctionnement de DeepDive. Extrait de (ZHANG et al., 2017)

DeepDive (ZHANG et al., 2017) est un système permettant de peupler une base de données représentant une base de connaissances à partir de données non structurées. Il étend Elementary, présenté dans (NIU, ZHANG et al., 2012) Son but est de permettre à des experts d'un domaine de construire plus rapidement une base de connaissances grâce à des outils statistique et d'apprentissage automatique. Plus précisément, DeepDive se base sur les réseaux logiques de Markov (*Markov Logic Networks* ou MLN, NIU, RÉ et al., 2011 ; DOMINGOS et LOWD, 2009) pour permettre à des experts d'incorporer de la connaissance sous forme de règles logiques dans un modèle statistique qui peut ensuite extraire les triplets les plus probables. Un boucle retour et la possibilité pour un utilisateur d'analyser les erreurs du système permet d'itérer jusqu'à obtenir une qualité satisfaisante de résultats.

DeepDive extrait des tuples qui vont ensuite peupler une base de données. L'exemple le plus utilisé dans les articles explicatifs de DeepDive est celui du challenge TAC-KBP *Slot Filling* (*Knowledge Base Population - Text Analysis Conference*), où DeepDive a obtenu les meilleurs scores de précision, recall et F1 en 2014. Une partie de cette tâche consistait par exemple à obtenir les paires de personnes mariées à partir d'un large corpus de texte. Le schéma correspondant est présenté Figure 2.2 (2)

Le détail du fonctionnement du système dans le cadre de l'exemple ci-dessus est le suivant :

1. Des candidats sont extraits par des fonctions déterminées par l'utilisateur et associés à une variable aléatoire $X_{table,ID}$ dans $\{0,1\}$ (p.ex. toutes les mentions conjointes de deux personnes dans une même phrase constitueront un candidat pour la relation `marié à`) Ils sont ensuite associés à un score qui provient du contexte entourant les mentions de ces candidats, qui déterminera à quel point ce contexte rend probable leur relation.
2. Une phase de supervision permet d'incorporer au système des données annotées et ainsi fixer certains tuples (Donc certains $X_{table,ID}$). Une autre manière consiste en l'utilisation de la supervision distante (MINTZ et al., 2009), pour

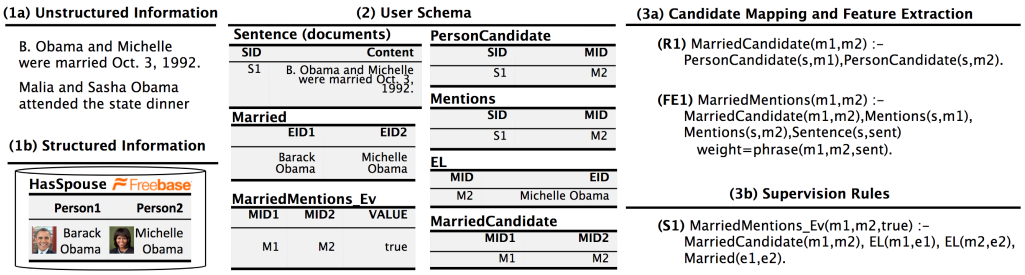

FIGURE 2.2 – Un exemple d'architecture du système DeepDive pour le challenge des personnes mariées. Extrait de (ZHANG et al., 2017)

s'aider de l'information d'une base de connaissance externe (dans laquelle on a par exemple des informations sur plusieurs couples) pour renforcer encore le modèle (Figure 2.2 (1b)).

3. Le modèle graphique basé sur les réseaux logiques de Markov (détaillés en Annexe A) est ensuite construit et la véracité de chaque tuple peut être inférée par échantillonnage de Gibbs sur le réseau.
4. Une phase d'analyse d'erreur permet à un utilisateur de consulter les éléments ayant été obtenus avec une haute confiance et d'écrire des règles supplémentaires pour corriger une faille du système. Des méthodes d'inférence incrémentale permettent de limiter la quantité de calculs à effectuer à nouveau pour actualiser les résultats.

Le système DeepDive a l'avantage de combiner la logique du premier ordre, qui permet à un utilisateur d'incorporer de manière relativement intuitive son savoir dans le modèle statistique qui est en revanche difficile à interpréter directement.

Snorkel : un outil d'annotation basé sur de la supervision faible

Snorkel (RATNER et al., 2017) a pour but d'être un outil qui permettrait à des scientifiques d'abord mais potentiellement à toute personnes ayant une connaissance suffisante d'un domaine de se constituer très rapidement un jeu de données annotées grâce à des fonctions d'annotation créées par l'utilisateur et des sources de supervision faibles comme la supervision distante (MINTZ et al., 2009), des motifs ou des heuristiques expertes.

Ce système est basé sur un modèle graphique génératif pour inférer une étiquette $\mathcal{Y} \in \{1, -1\}$ à partir d'un candidat x. Ce contexte très général permet d'imaginer de nombreux cas d'application, où par exemple x est une paire candidate pour une relation donnée (p. ex. `hasSpouse('Barack Obama', 'Michelle Obama')`) et il faut déterminer si ce candidat est valide ou non.

Les données sont représentées dans une matrice $\Lambda_{i,j} = \lambda_j(x_i)$ ou λ_j est une fonction d'annotation qui peut retourner -1, 1 ou 0 pour une abstention. Le modèle est ensuite défini par l'équation

$$P(\Lambda, Y) \propto \exp\left(\sum_i w^T \phi_i(\Lambda, y_i)\right)$$

Un exemple de modèle graphique le plus simple suivant cette spécification serait celui qui a toutes ses entrées indépendantes, c'est à dire $\phi_i(\Lambda, y_i) = (\mathbb{1}\{\Lambda_{i,j} = y_i\})$.

Ce modèle est équivalent à une régression logistique sur les fonctions d'annotation pour classifier les entrées.

Ce modèle de base est affiné dans (BACH et al., 2017) d'abord puis dans (RATNER et al., 2017) pour arriver à un modèle génératif prenant en compte la corrélation entre les fonctions d'annotation et la « propension à annoter » d'une fonction d'annotation. Ceci permet d'obtenir pour un jeu de donnée un ensemble de labels sous forme de probabilités.

Snorkel est théoriquement efficace dans le cas précis où la densité d'annotation produit par les fonctions en entrée n'est ni trop haute ni trop basse. Cependant, de manière similaire à DeepDive, les performances dépendent de manière très importante de la qualité et du nombre de fonctions d'annotation que l'utilisateur produit.

Google Knowledge Vault

Le Knowledge Vault mis au point par Google (X. DONG et al., 2014) a pour ambition de créer automatiquement une base de connaissance en parcourant le web. Il a été initialement construit à partir de la base de connaissances Freebase, construite par une communauté de volontaires, et se concentre donc sur des classes d'entités et relations issues de celle-ci. Essentiellement, ce système se base sur une classification supervisée de triplets extraits d'un corpus de pages web.

Les auteurs font l'hypothèse LCWA décrite dans 2.2.1 qui permet d'obtenir des exemples négatifs à partir de donnée annotées. Le système est organisé sous la forme d'une chaîne d'opérations dont la sortie d'une correspond à l'entrée d'une autre (aussi appelé *pipeline*). Les différentes étapes sont décrites ci-après :

1. Les extracteurs sont responsable d'extraire des triplets (s, p, o) (sujet, prédicat, objet) à partir des pages web en entrée du système. Il en existe quatre types conçus pour quatre types d'éléments d'information dans une page web :

   Texte Un modèle extrait des relations à partir d'un texte grâce à des augmentations classiques (reconnaissances d'entités nommées, étiquetage morpho-syntaxique ou *POS tagging*, analyse de dépendances, liaison d'entités) et l'entraînement d'un modèle de classification basé sur une régression logistique. Le je d'entraînement est obtenu par supervision distante (MINTZ et al., 2009)

   Arbres DOM Un extracteur qui utilise les données du *Deep Web* (par exemples stockées dans des bases de données non directement accessibles mais avec lesquelles des interactions sont possibles avec des formulaires HTML). Des outils spécifiques sont utilisée ici pour l'extraction de caractéristiques et le lien des entités. Ils furent développés dans le cadre du *Google Deep Web Crawl Project* (MADHAVAN et al., 2007)

   Tables HTML Cet extracteur part du principe que les tables HTML présentes dans les pages Web peuvent contenir des données relationnelles utiles. Des techniques d'extraction spécifiques à cet tâche ont été développés et sont décris dans (CAFARELLA, HALEVY et MADHAVAN, 2011)

   ANO Les pages annotées directement par les humains, notamment en suivant un des multiples standards (`schema.org`, `microformats.org`, *open graph protocol*, etc.) permettent d'accéder très facilement à des données de qualité sous réserve de l'existence d'un mapping entre le standard en question et le schéma Freebase ainsi que la capacité à effectuer du liens d'entités nommées entre les entités de la base de connaissance et celles de la page web annotée.

2. Pour ajouter de la connaissance et augmenter les prédictions sur les triplets ayant un faible nombre d'exemples, des informations à priori sont ajoutées au système grâce à des quantités calculées à partir du graphe lui-même plutôt que le texte extrait. Il y a deux types de méthodes utilisées :

   PRA Le *Path ranking algorithm* a été introduit dans (LAO et COHEN, 2010) et est décrit avec plus de détails dans l'annexe B. Il consiste en essence d'un grande quantité de marches aléatoires sur un graphe de connaissances connu qui permet ensuite d'apprendre des chemins reliant deux entités qui ont une forte probabilité de correspondre à un relation entre celles-ci. Par exemple, pour la relation Freebase `marriedTo` entre deux éléments X et Y, l'algorithme peut apprendre qu'avec une forte probabilité, un chemin $X \xrightarrow{\texttt{parentOf}} Z \xleftarrow{\texttt{parentOf}} Y$ existera et réciproquement.
   L'algorithme permet ainsi de déduire un probabilité d'existence d'une relation entre deux entités en regardant uniquement comment elles sont liées dans le graphe.

   Réseau de neurones Une seconde manière d'avoir des probabilités à priori dans le modèle de Knowledge Vault consiste à créer une matrice G creuse en 3d où une première dimension est associée aux entités, une seconde aux relations et la troisième aux entités à nouveau. Ce cube contient des 1 là où le triplet est vérifié et des 0 ailleurs. Pour un triplet (s, p, o), trois vecteurs u_s, w_p et v_o leurs sont associés de manière à ce que la quantité suivante soit minimisée

   $$P(G(s,p,o) = 1) = \sigma(\beta^T f\left(A \cdot [u_s, w_p, v_o]\right))$$

   où $\sigma(x) = \frac{1}{1+\exp(x)}$ désigne la fonction sigmoïde, f un fonction non linéaire typiquement utilisée dans les réseaux de neurones comme tanh, A une matrice de taille $L \times 3K$, K la dimension de chaque vecteur u, w et v et β un vecteur de taille $L \times 1$. L est donc la dimension de la couche cachée du réseau de neurones et K celle de l'espace dans lequel les entités et relations sont représentées. Après avoir entraîné ce modèle sur une base de connaissances connue, le réseau de neurone permet d'obtenir une probabilité à priori pour un triplet (s, p, o) arbitraire grâce à l'équation ci-dessus.
   Plus de détails sur ces modèles seront donnés en annexe [2], notamment sur les variantes possibles de celui ci-dessus, leurs performances et complexités algorithmiques.

3. Pour les extracteurs, et les prédicteurs à priori, des modèles de classification (régressions logistiques dans l'article) sont appris sur le jeu d'entraînement pour donner des poids et importances différentes à chacun des composantes. Pour chaque triplet le nombre de fois qu'il a été extrait par extracteur ainsi que le score moyen à l'extraction sont utilisée pour régler le modèle. Il apparaît que l'extracteur le plus fructueux est celui basé sur les arbres DOM et que sa qualité se compare avec celle des tables HTML.
4. Finalement, une dernière classification binaire est effectuée entre le score du triplet extrait et son score à priori pour prendre en compte la relative importance de ces deux composantes. Le score résultant permet au système de décider si le triplet doit être ajouté à la base de connaissances ou non.

Le système global est représenté dans le schéma Figure 3.2.

2. Non présente dans la version préliminaire de ce rapport

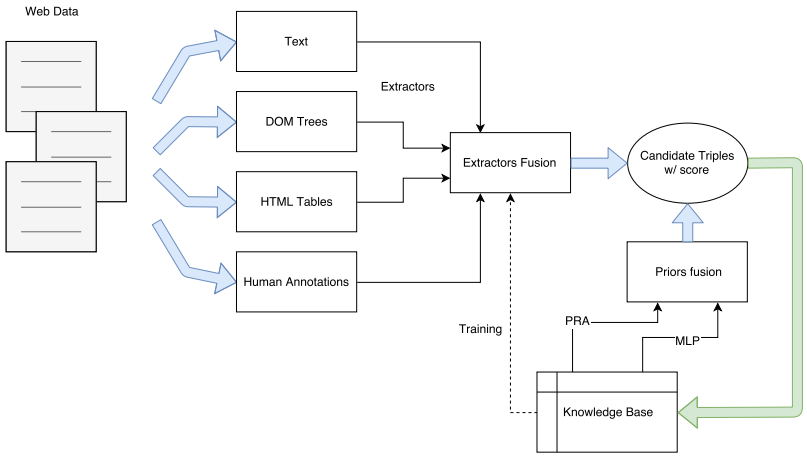

FIGURE 2.3 – Schéma descriptif du système Knowledge Vault de (X. DONG et al., 2014)

Le modèle Knowledge Vault exploite de manière parallèle différents paradigmes pour l'extraction de relations, utilisant des modèles de représentations latentes ainsi que la structure du graphe de connaissances lui-même avec le PRA par exemple. Knowledge Vault a la particularité d'avoir été développé par Google, qui a accès à une quantité très importante de ressources web, ce qui permet à la base de connaissances résultante d'être 38 fois plus grande qu'une autre construite à partir du système DeepDive (voir 2.2.2) sur des sources similaires [3].

En plus de cette architecure proposée, les auteurs présentent dans (X. L. DONG et al., 2015) une variante du paradigme classique d'extractions d'informations à partir de multiples sources appelé *Data fusion* et où l'information peut être représentée par une matrice avec une dimension correspondant aux éléments de données et une autre correspondant aux possibles sources. Cette variante est appelée *Knowledge fusion* et consiste formellement à ajouter une dimension à la matrice précédente pour ajouter les différents extracteurs de données possibles qui peuvent avoir leur qualité et capacité propre qui doit être prise en compte au moment de décider si un élément d'information extrait est vrai ou faux. La différence entre ces deux approches est illustrée dans le schéma Figure 2.4

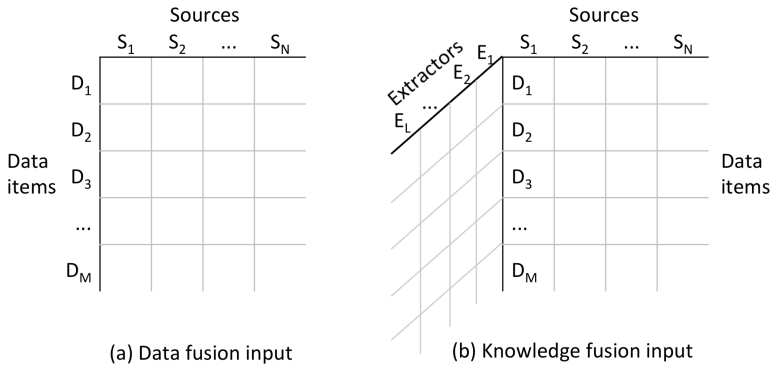

FIGURE 2.4 – Schéma descriptif des différences entre *Data fusion* et *Knowledge fusion*. Extrait de (X. L. DONG et al., 2015)

3. Les sources de données utilisées dans (X. DONG et al., 2014) n'étant pas standard, les deux expériences ne sont pas exactement comparables

2.2.3 Extraction automatique sans schéma cible

L'extraction d'information sans schéma cible (ou *Open information extraction*) a pour but d'extraire automatiquement à la fois les triplets d'information mais aussi les classes d'entités et types de relation. Ce type de système, s'ils étaient suffisamment performant, permettraient de passer l'extraction d'information à l'échelle du web entier et de la diversité importante d'information qui y est présente. Les relations extraites sont arbitraires et découvertes pendant l'extraction elle-même ; la plupart des méthodes d'extraction sans schéma cible essayent d'avoir les relations les plus génériques possibles.

Classiquement, il y a trois étapes à ce type d'extraction selon (ETZIONI et al., 2011) :

1. **Étiquetage** les phrases sont syntaxiquement analysées, puis les entités sont associées heuristiquement à des séquences de mots censés dénoter une relation. Cela permet de créer un ensemble annoté.
2. **Apprentissage** cet ensemble est ensuite passé dans un modèle de classification qui va déterminer si une séquence de mot représente une relation ou non.
3. **Extraction** de nouvelles phrases sont analysées et à nouveau classifiée par l'algorithme de l'étape 2.

Quelques exemples de systèmes implémentant et étendant les étapes présentées sont décrits plus bas. Bien que beaucoup de recherches aient lieu dans ce domaine, il reste difficile d'obtenir des résultats fiables et exploitables, comme le montrent les évaluations des systèmes présentés ci-après.

TEXTRUNNER

Le système TEXTRUNNER (YATES et al., 2007) implémente les étapes décrites ci-dessus pour faire de l'extraction d'information sans schéma. Les résultats sont très bruités et nécessitent beaucoup d'étapes de filtrage afin d'obtenir des relations concrète et bien formées ainsi que des entités bien formées. Parmi ces relation issues du filtrage, il reste une forte redondance, notamment dans les relations extraites qui se recoupent.

REVERB

Dans (FADER, SODERLAND et ETZIONI, 2011), les auteurs identifient les défauts du système TEXTRUNNER et mettent au point des manières d'en venir partiellement à bout. Ils utilisent par exemple des contraintes syntaxiques sur les phrases qui vont être utilisées pour extraire des relations. Cela leur permet notamment de réduire les relations incohérentes ou non informatives. De plus, des contraintes lexicales sur le nombre de mots permettent aussi de garder de la généralité dans les phrases qui décrivent les relations extraites. Les résultats ainsi obtenus sont significativement supérieurs à ceux du système TEXTRUNNER.

Une autre extension nommée R2A2 (ETZIONI et al., 2011) étend encore REVERB avec une série de motifs syntaxiques et lexicaux qui permettent de détecter les entités de manière plus fiable.

Modèles génératifs

Plus récemment, des approches utilisant des modèles génératifs ont été étudiées (CHAMBERS, 2011 ; CHEUNG, POON et VANDERWENDE, 2013) . Ceux-ci ont de multiples avantages, comme celui d'être plus simples conceptuellement que les modèles décrits plus haut tout en gardant des performances comparables avec moins de données d'entraînement. Ils permettent notamment dans des problèmes plus restreints (comme l'étude des évènements) d'obtenir de meilleurs résultats que les méthodes classiques (CHAMBERS, 2013 ; NGUYEN et al., 2015).

2.3 Alignement de bases de connaissances - Quelques exemples

Les exemples présentés tout au long de la section 2.2 correspondent, comme le titre l'indique, à une construction de base de connaissances, c'est à dire soit la construction de toute pièces d'une nouvelle base ou encore la complétion et ajout d'éléments à une base de connaissances préalablement existante. Ces approches ont été relativement bien étudiées, même si des progrès peuvent encore être espérés. Il est intéressant de noter qu'une autre approche existe, qui consiste à assumer que deux ou plus bases de connaissances sont disponibles et déjà construites mais présentent des recoupements. L'alignement de base de connaissances consiste à construire des méthodes pour automatiquement fusionner les entités et relations de ces bases de connaissances qui sont les mêmes, alors même que leur nom ou attributs ne sont pas nécessairement les mêmes. Ce domaine, qui semble avoir proportionnellement reçu moins d'attention que le premier, peut cependant permettre des applications intéressantes pour la construction automatique de bases de connaissances et le web sémantique.

2.3.1 PARIS

PARIS ou *Probabilistic Alignment of Relations, Instances, and Schema* (Fabian M. SUCHANEK, ABITEBOUL et SENELLART, 2011) est un algorithme pour l'alignement automatique d'ontologies au niveau des instances, relations et classes. Cette méthode permet de construire des estimations de la probabilité pour deux instances, relations ou classes d'être les mêmes. Le but de l'approche est de lier deux domaines déjà bien étudiés (l'alignement d'entités et l'alignement de schémas) dans un seul système.

Pour cela, PARIS distingue trois sous-tâches :

— L'équivalence d'instances
— Les sous-relations
— Les sous-classes

Le modèle est probabiliste et calcule donc des probabilités pour par exemple deux instances d'être les mêmes. Cette première tâche est présentée rapidement ci-dessous pour illustrer l'approche de formalisation effectuée par les auteurs de l'article. L'observation principale est que deux instances de graphes de connaissances x et x' distincts ont beaucoup de chances d'être équivalentes si il existe un grand nombre de y et y' équivalents tels que x est en relation avec y et x' en relation avec y'. Par ailleurs, cette observation tient dans le cas où la relation est inversement *fonctionnelle* (ou encore qu'elle est injective), c'est à dire qu'elle a tendance à n'avoir qu'un seul antécédent par élément [4].

4. Cette notation introduit des concepts associés à des fonction pour les relations d'un graphe, un triplet (s, p, o) n'est donc plus seulement un tuple mais un application $p(s) = o$ ou encore $p(s, o) = 1/0$, suivant les notations. La notion d'injectivité d'une relation devient ainsi plus évidente.

Les auteurs définissent la fonctionnalité d'une relation dans un graphe par la formule suivante

$$fun(r) = \frac{\#x : \exists y : r(x,y)}{\#x, y : r(x,y)}$$

Où $\#a$: **P** désigne le compte du nombre d'éléments a vérifiant la proposition P. Cette quantité est maximale lorsque elle est égale à 1 et que pour chaque y dans le graphe il y a un unique x vérifiant $r(x,y)$.

La probabilité pour deux instances x et x' s'écrit donc

$$P_1(x \equiv x') = 1 - \prod_{r(x,y);r(x',y')} (1 - fun^{-1}(r) \times P(y \equiv y'))$$

Dans l'autre sens, si tous les éléments en relation avec x' ne le sont pas avec y et que la relation est bien fonctionnelle, alors les deux éléments x et x' ne doivent pas être équivalents, ce qui s'écrit

$$P_2(x \equiv x') = \prod_{r(x,y)} \left(1 - fun(r) \prod_{r(x',y')} (1 - P(y \equiv y')) \right)$$

Et finalement la probabilité que x soit équivalent à x' est

$$P(x \equiv x') = P_1(x \equiv x') \times P_2(x \equiv x')$$

Les auteurs définissent aussi d'autres quantités pour les sous-classes et sous-relations et initialisent toutes les probabilités à un nombre arbitraire. Ensuite le calcul des probabilités est fait itérativement sur tous les éléments. Les auteurs remarquent que la convergence, non théoriquement garantie, est empiriquement atteinte lors de chacune de leur expériences. Cet algorithme donne des résultats concluants sur les plus grosses bases de connaissances disponibles en 2011. Cependant, le calcul de toutes les probabilités demande un puissance et un temps de calcul important qui serait probablement un facteur limitant sur les bases actuelles.

2.3.2 SIGMA

SiGMA (*Simple Greedy Matching for Aligning Large Knowledge Bases*, LACOSTE-JULIEN et al., 2013) est une méthode qui se base sur la structure locale du graphe entourant chaque nœud pour aligner deux graphes de connaissances. Il se base sur un algorithme glouton qui assigne progressivement les nœuds d'un graphe à ceux d'un autre de manière à réduire à chaque étape le score SCORE = STATIC SIMILARITY + NEIGHBORHOOD SIMILARITY, c'est à dire minimiser pour un nœud, lui assigner le candidat dont la similarité brut (ou statique) est la plus grande avec lui et aussi dont le voisinage a la plus grande similarité. Le point crucial est donc de définir de bonnes métriques de similarité pour chacune de ces mesures. Il est assez naturel d'observer que cet algorithme souffre du problème du « démarrage à froid », c'est à dire qu'une assignation initiale est nécessaire pour que l'algorithme puisse commencer à assigner de manière relativement fiable les nœuds les uns avec les autres.

Dans l'article, les similarités sont définies de la manière suivante :

Similarité statique $s_{ij} = (1-\beta)\text{STRING}(i,j) + \beta\text{PROP}(i,j)$ où β est un coefficient qui détermine l'importance relative des deux éléments, STRING mesure le nombre de mots en commun des deux chaînes de caractères associées à i et j et PROP mesure le nombre de propriétés communes des deux nœuds.

Similarité dynamique Cette similarité est basé sur un comptage pondéré des voisins ayant déjà été assignés. Cette mesure de similarité permet non seulement de calculer le score à minimiser avec l'algorithme mais aussi à déterminer l'ordre de traitement des nœuds dans le graphe [5].

Il est important de noter que dans cet algorithme, malgré que les bases soient différentes, il est fait l'hypothèse que les relations se correspondent et qu'il existe un mapping des unes aux autres. Cet algorithme fonctionne dans le cas de bases de connaissances où les entités ont des propriétés et des relations.

2.3.3 Représentations pour l'alignement

L'alignement de graphes de connaissances peut aussi profiter des approches visant à représenter les entités et relations par des variables latentes dans un espace dense de faible dimension. C'est l'approche envisagée dans (CHEN et al., 2017), pour l'alignement de graphes de connaissances dans des langues distinctes.

5. Les nœuds les plus « facile » à assigner sont ceux qui ont le plus de voisins assignés et sont donc traités en priorité par l'algorithme.

3 Une architecture d'intégration et d'extraction pour le journalisme de données

Cette partie présente le travail effectué au cours du stage. Plusieurs éléments sont présentés : les objectifs du projet et la manière dont ils ont été définis, l'architecture proposée à proprement parler et ses spécificités, puis enfin les cas d'études qui ont permis à la fois de tester mais aussi d'améliorer la mise au point du système complet.

3.1 Objectifs

Le but de l'architecture proposée ici est de permettre de construire à partir de plusieurs sources de données (JSON, XML, fichier texte, base de données, etc.) un système capable d'extraire des données afin de peupler un schéma cible, trouver des liens entre les entités extraites, d'exécuter des opérations d'algèbre relationnelle arbitraires et d'enrichir les entités extraites avec des informations venant d'autres sources déjà structurées comme des bases de connaissances externes par exemple.

Ces extractions permettraient de construire un entrepôt de données dont le contenu pourrait être exploitable dans le cadre d'une ou plusieurs applications.

3.2 Architecture

Sont décrits ici les différents modules ou composantes logicielles qui constituent l'architecture du système. Plusieurs éléments sont considérés comme disponibles au préalable :

— Un ensemble de sources de données accessibles (pas de contrainte spécifique sur la forme de ces sources).
— Un schéma cible qui représentera les données (ou encore un modèles E-R).
— Un ensemble d'imports et de règles de mappings comme expliqué plus bas.
— Un ensemble de spécifications, c'est à dire pour un flux de données, un certain. attribut ou résultat d'une requête effectuée sur ces données est considéré comme clé primaire de cet ensemble.
Exemple : le système peut inférer que si R.a était une clé primaire de la table R, c'en est aussi une pour le flux obtenu à partir d'une requête SELECT * FROM R.
Exemple : l'utilisateur peut spécifier que l'attribut @depID d'un flux XML venant de l'url `nosdeputes.fr` est une clé primaire pour ces éléments.

3.2.1 Composants

Les différents types de modules sont les suivants :

1. **Importation de données** à partir de base de données structurées ou non, locales ou distantes.
   — Un importeur de données fonctionne pour un modèle de données précis et souvent pour un type de source de données qui lui correspond (fichier, flux RSS, flux Twitter, DB, etc.). Il peut par exemple être construit pour des données relationnelles dans n'importe quelle base de données supportant le langage SQL.
   — À partir d'un requête faite à la source de données sous-jacente à l'importeur, ce module met le résultat à disposition des autres modules.

   Les différents composants pour l'importation de données implémentés dans la version du logiciel à la date d'écriture du rapport sont :
   — Un composant pour extraire des données à partir d'un fichier CSV.
   — Un composant pour extraire des données à partir d'un fichier JSON ayant une structure particulière (une liste d'éléments ayant comme attributs des chaînes de caractères).
   — Un composant pour extraire des données à partir d'une base de données relationelle et une requête SQL donnée.
   — Un composant pour extraire des données à partir d'une base de données MongoDB, soit en spécifiant des attributs particuliers, soit en exécutant une requête dans le langage de requête MongoDB.
2. **Moteur de Mapping** Étant donnée une spécification de la manière selon laquelle les données doivent être ajoutées au schéma cible, ce module peuple effectivement les instances.

   Pour un mapping et un flux de données, le moteur de mapping applique ce mapping sur chaque élément du flux et créé les objets instanciés selon la spécification exacte du schéma cible.

   Ce module a un caractère fictif, puisqu'il fait en fait partie des modules d'importation de données. En effet, lorsque des données sont extraites selon un certaine méthodes à partir d'une source, le schéma de données sous-jacent est implicitement donné dans la méthode d'extraction. Ainsi, le flux de données résultant, représenté par le moteur Flink par des tuples, est une collection de d'instances correspondant à ce schéma implicite.
3. **Module de lien** Étant donnés deux flux de données ainsi qu'une fonction qui pour chaque paire d'instances retourne une instance unifiée produite à partir des deux premières, le module effectue une jointure des deux flux.
   — La sémantique de la jointure peut être une jointure normale, une jointure externe gauche, droite ou une jointure externe complète.
   — Il peut y avoir différentes hypothèses concernant la possible présence de doublons dans le flux de données. En cas de doublons dans l'une des sources, le module décide de choisir parmi les doublons ou de les unifier. Ces hypothèses peuvent notamment être informées par l'existence d'une clé primaire dans les données.

   Ces modules de jointure sont implémentés et disponibles. Deux manière d'envisager le résultat des jointures sont possibles dans le cadre du moteur Flink :
   — Sous la forme de tuples imbriqués (une jointure de deux flux de la forme `(Int, String)` et `(Int, Double)` donnerait `((Int, String), (Int, Double))`)
   — Sous la forme d'un tuple *plat* (pour l'exemple précédent le résultat serait `(Int, String, Double)` ou `(Int, Double, String)`)

   Les deux sont possibles à obtenir avec les modules de jointure, car la première est une représentation fidèle d'une jointure à proprement parler. La seconde

est en revanche plus utile dans le cadre de l'utilisation prévue du logiciel, car en intégrant un projection elle permet de ne manipuler que des tuples et évite de cette manière d'effectuer des opérations imbriquées sur des tuples imbriqués, ce qui pourrait rapidement rendre une série d'opérations difficile à suivre.

4. **Extraction d'entités** Étant donnés un flux de données entrant et une fonction qui identifie 0, 1 ou plus d'entités dans chaque élément du flux, l'extraction va créer une paire pour chaque élément de l'entrée et l'ensemble résultant de l'extraction.
5. **Extraction de relations** Similaire à ci-dessus, mais au lieu de retourner des ensembles d'instances, le module retourne un ensemble (possiblement vide) de paires d'instances représentant une relation extraite par une fonction d'extraction.

   Un seul module d'extraction d'entités est disponible, il permet d'appliquer un recherche basée sur des expressions régulières pour extraire dans un flux de données les occurrences d'éléments d'un autre flux.

   Ce module peut être utile dans le cas où l'occurence de noms de personnes ou entreprises doivent être recherchés dans un groupe de documents.
6. **Opérations relationnelles** Un module effectuant une opération classique d'algèbre relationnelle comme un filtrage, sélection, projection, union, intersection, co-group, etc.

   Des modules permettent non seulement d'effectuer des opérations relationnelles mais aussi d'effectuer des opération fréquentes de traitement de données textuelles ou autre. Par exemple sont implémentés un module pour appliquer une fonction arbitraire à un flux de données, pour compter les éléments distincts d'un flux de données, pour séparer un flux de données sous forme de texte selon un délimiteur donné.

Ces différents modules peuvent être chaînés pour obtenir un traitement plus ou moins complexe d'une ou plusieurs entrées.

3.2.2 Documentation

Un documentation du logiciel est disponible à l'URL suivante :
`https://data-journalism-extractor.readthedocs.io/en/latest/`
Celle-ci donne des instructions d'installations, quelques exemples et cas d'application du logiciel ainsi qu'une description détaillée du fonctionnement des modules implémentés.

Le code source est disponible sur l'hébergeur Github sous le nom de
`data-journalism-extractor`

3.2.3 Deux paradigmes de manipulation de données et de planification de tâches

L'implémentation de l'architecture décrite ci-dessous présente des particularités qui implique un choix de paradigme pour la représentation, la modélisation et la manipulation des données dans le cadre de l'application visée ici. Une distinction entre deux paradigmes est souvent faite : *workflow* et *dataflow*. Comme ces deux notions ne disposent pas de définition standard ni de documentation extensive à leur sujet, une définition spécifique au contexte étudié ici en est donnée.

Workflow

Le terme workflow (flux de travail) désigne une manière de représenter une suite de tâches. Le terme n'est pas spécifique à l'informatique ou au traitement de données et est par exemple utilisé dans le management.

Il désigne dans le domaine ici étudié, une spécification de tâches successives. Il y d'une étape à l'autre un passage de contrôle, c'est à dire que chaque élément de la suite de tâches peut connaître le statut des autres tâches et donc s'exécuter uniquement lorsque les conditions pour son exécution sont remplies [1], mais ce sont les seules données qui peuvent être transférées. C'est pourquoi ce type de système est différent des pipelines, qui utilisent la sortie des étapes précédentes comme entrées.

Il existe de nombreux systèmes pour la spécification et l'exécution de workflows. Parmi ceux-ci, Apache Airflow [2], Luigi [3] ou Toil [4] utilisent le langage Python comme pseudo-langage de spécification de workflow. Celui-ci permet de spécifier la nature des tâches, leur contenu ainsi que l'ordre de déroulement. Les workflows y sont représentés par des graphes acycliques dirigés (DAGs en anglais) et un moteur d'exécution se charge de dispatcher les tâches à travers différents *workers* et permet à des utilisateurs ou d'autres applications de visualiser, surveiller et modifier le déroulement de la suite de tâches.

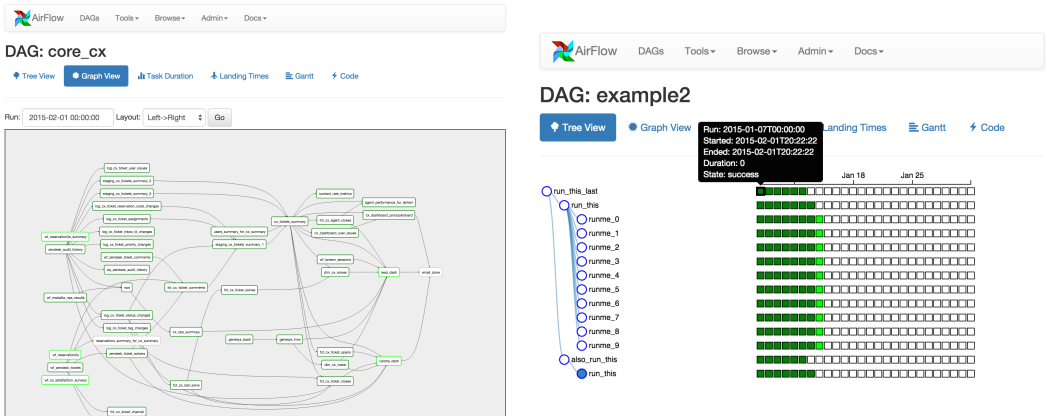

FIGURE 3.1 – Exemple de visualisation offerte par les outils de création de workflows, la première Figure est le DAG représentant une suite d'opération, et la seconde une table d'avancement des différentes opérations.

Extrait le 19/06/2018 de https://github.com/apache/incubator-airflow

Malgré les nombreux avantages que procurent ces systèmes et qu'ils permettent l'implémentation de fonction arbitraires, toutes les fonctionnalités permettant de manipuler les données et d'y effectuer des opérations ne sont pas facilement accessibles car ces sytèmes sont principalement destinés au management de workflow. Leur utilisation pour l'implémentation du système décrit dans 3.2.1 impliquerait faire reposer toutes les opérations, et traitements de données sur d'autres systèmes comme un système de base de données relationnelle par exemple. Le système envisagé et ses différents modules suggèrent plutôt une modélisation centrée sur les

1. Ces condition peuvent être complexes et dépendre à la fois du statut des autre tâches du workflow mais aussi de variables externes, ce qui fait la flexibilité de cette manière de représenter une série de tâches.
2. https://github.com/apache/incubator-airflow
3. https://github.com/spotify/luigi
4. https://toil.readthedocs.io/en/latest/

données et leur passage d'un module à l'autre, ce qui laisse penser que l'utilisation de ce type de système est superflue.

C'est pourquoi le paradigme dataflow semble plus adapté à ce type d'application.

Dataflow

Le dataflow se définit dans le contexte étudié ici par la spécification d'une suite de tâches. Par opposition au workflow, les tâches sont ici définies par une opération sur les données et sur le passage de celles-ci d'une tâche à l'autre. Le contrôle est ensuite implicitement guidé par le passage effectif de ces données et mis en place par le système de dataflow.

Google Cloud DataFlow [5], Apache Spark [6] et Apache Flink [7] sont des exemples de systèmes qui permettent d'implémenter des dataflows. Ces trois systèmes disposent de SDK en Java et permettent la manipulation de données dans des objets analogues aux graphes acycliques décrits plus hauts pour les workflows mais avec des opérations et spécifications de la nature des données en entrée et sortie contre des tâches simples pour les workflows.

Le système sélectionné pour la mise en place d'un prototype est Apache Flink, qui est une plateforme construite pour le traitement de données en streaming ou en batch. Il a l'avantage de permettre un passage à l'échelle facile et d'avoir de hautes performances. Il permet aussi de construire des applications basées sur le streaming de données même si ce n'est pas un fonctionnalité qui est utilisée ici.

L'architecture d'un programme Flink est simple et se compose d'une *DataSource*, objet qui extrait des données et qui communique avec une série de *Transformations*, où les données sont modifiées, qui enfin transfère les données à un *DataSink* où les données sont soit stockées ou transférées à d'autres programmes.

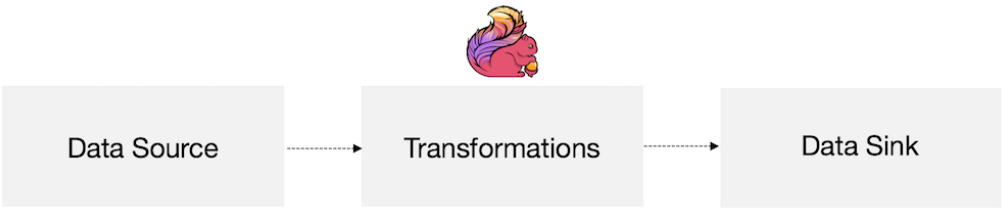

FIGURE 3.2 – Un programme simple avec Apache Flink

Extrait le 16/06/2018 de https://flink.apache.org/introduction.html

3.2.4 Implémentation

Pour être au maximum en accord avec les objectifs fixés dans 3.1, notamment être capable de construire une architecture modulaire à partir d'une série de spécifications des sources, types, opérations de traitement il a fallu choisir la manière dont cette spécification allait être faite et la manière dont celle-ci pourrait être convertie automatiquement dans un programme Flink.

5. https://cloud.google.com/dataflow/
6. https://spark.apache.org/
7. https://flink.apache.org/

Flink supporte Java et Scala [8], et c'est Scala qui a été choisi pour l'implémentation du système car il offre les possibilités d'un langage avec une forte composante fonctionnelle et qu'il est moins verbeux que Java, ce qui sera utile pour simplifier la manipulation des composantes pour la compilation comme expliqué plus bas.

La manière retenue pour parvenir à cet objectif fut de créer un langage de spécification basé sur JSON. Celui-ci permet de définir les modules stockés dans une liste à la racine `modules` du fichier de spécifications. Chacun de ces modules nécessite un certain nombre de paramètres pour pouvoir spécifier la manière d'extraire les informations. Par exemple, pour définir un module d'importation de données à partir d'un fichier CSV, il faut connaître le chemin du fichier, les éventuels délimiteurs de colonnes et lignes ainsi que les types des données extraites. Il pourrait être défini comme ceci :

```
{
    "name": "extractor1",
    "type": "csvImporter",
    "path": "/path/to/file.csv",
    "dataType": ["String", "String"],
    "fieldDelimiter": ";"
}
```

La compilation depuis ce langage de spécification vers du code Scala se fait à partir d'un programme écrit en Python qui utilise un moteur de *template* nommé Jinja2 [9] et qui permet, grâce à un langage de description de template de remplir à partir de variables lues dans le fichier un code Scala vierge et l'insérer par exemple dans un autre programme.

Le compilateur pourrait par exemple générer le code Scala suivant pour le module décrit plus haut. Pour une liste de modules, une liste de blocs de texte comme celui-ci peuvent être générés.

```
// ===== CSV Importer module extractor1 =====

val filePath_extractor1 = "/path/to/file.csv"
val lineDelimiter_extractor1 = "\n"
val fieldDelimiter_extractor1 = ";"

val extractor1 = env.readCsvFile[(String,String)](
    filePath_extractor1,
    lineDelimiter_extractor1,
    fieldDelimiter_extractor1)
```

Ce code peut ensuite être inséré dans le programme de base et constituera un nouveau module DataSource pour Flink après sa compilation. Dans le langage de template, toutes les variables ci-dessus sont des espaces réservés qui sont remplis pendant la compilation du fichier de spécifications par le programme écrit en Python.

Comme tous les modules ne sont pas aussi simples que celui-ci et que certains nécessitent de communiquer avec d'autres dans le graphe des tâches, certaines vérifications et ajustements sont nécessaires au delà du simple remplacement de chaque

8. Flink supporte aussi Python mais beaucoup de fonctionnalités ne sont pas supportées et la documentation est très peu fournie pour ce langage.

9. http://jinja.pocoo.org/

module par un bloc de texte. En effet, une des forces de Flink est qu'il permet facilement l'intégration de fonctions externes ("user functions") dans des tâches du graphe de calcul. Ainsi, les modules d'extraction d'entités et de relation peuvent être connectés avec des outils existants comme StanfordCoreNLP pour la reconnaissance d'entités nommées ou encore l'extraction de caractéristiques linguistiques.

Une trentaine de classes Scala ont été écrites, totalisant avec les compilateur et les templates à peu près 1200 lignes de code.

3.3 Cas d'étude

Les spécifications et l'implémentation de cette architecture de traitement de données pour le journalisme s'est faite par l'étude de cas réels d'utilisation de ce type d'outils. Cette manière de fonctionner a permis d'identifier rapidement les éléments nécessaire à son bon fonctionnement, les modules indispensables au traitement des données, le degré de flexibilité ainsi que la facilité d'utilisation de l'outil.

3.3.1 Députés et Lobbies en France

Comme en témoignent de nombreux articles de presse sortant assez régulièrement, le sujet des lobbies en France et de leur influence sur les parlementaires est particulièrement suivi. Il n'est pas pour l'instant sujet de rendre ces relations totalement transparentes, ce qui peut avoir pour effet de stimuler les imaginations.

Représentants d'intérêt en France

Le premier cas d'étude concerne l'étude des relations entre députés et représentants d'intérêts [10] en France. Selon la Haute autorité pour la transparence de la vie publique (HATVP), les représentants d'intérêts sont définis par trois points qui entraînent quelques obligations (NORMAND, 2018) :

1. Un représentant d'intérêt peut être une personne morale, c'est à dire une entreprise (publique ou privée), un cabinet d'avocats ou une société de conseils, un syndicat. « Un représentant d'intérêts peut également être une personne physique qui exerce à titre individuel, par exemple un consultant ou un avocat indépendant. »
2. Il doit également exercer une activité de représentation d'intérêts, « c'est- à-dire qu'il prend l'initiative de contacter un responsable public pour essayer d'influencer une décision publique. » Il s'agit souvent d'une loi ou d'une réglementation.
3. Il doit enfin exercer cette activité de façon principale ou régulière. « Il s'agit d'une activité principale s'il consacre plus de la moitié de son temps, sur une période de six mois, à préparer, organiser et réaliser des actions de représentation d'intérêts. Il s'agit d'une activité régulière s'il a réalisé à lui seul plus de dix actions d'influence au cours des 12 derniers mois ».

Les individus ou organisations qui remplissent les critères ci-dessus sont tenus depuis le 1er juillet 2017 de s'inscrire sur un registre public et disponible en données ouverte sur le site de la HATVP. Ces données sont publiées sous la licence ouverte Etalab sous un format JSON.

10. Communément appelés lobbyistes

Ce registre contient les informations concernant l'organisation représentante d'intérêts : ses dirigeants, représentants au Parlement, affiliations, et clients le cas échéant. Ces organisations sont par ailleurs tenues d'y déclarer toutes leurs activités relatives à la représentation d'intérêt (nature, proposition ou domaine visé et budget).

Les parlementaires

Les informations concernant les parlementaires français sont toutes disponibles et librement réutilisables sur les sites de l'Assemblée nationale et du Sénat.

Ces données sont aussi accessibles en données ouvertes sous la licence ouverte Etalab. Cependant, toutes les informations concernant les parlementaires, leurs affiliations et activités sont consignées dans des fichier monolithiques en format XML, JSON ou CSV difficilement exploitables en tant que tel.

C'est pour cela que le collectif Regards citoyens, dont le but est de mettre en place des « accès simplifiés au fonctionnement des institutions démocratiques à partir des informations publiques », formé en juillet 2009 a créé des services Web et une API basée sur REST [11] qui permet un accès simplifié aux informations des députés selon un schéma défini. Les APIs ainsi que des applications web permettant de parcourir les données concernant les sénateurs et députés sont disponibles aux adresses web `nosdeputes.fr` et `nossenateurs.fr` [12].

Le collectif a de plus mis à disposition des fichier contenant toutes ces informations avec un schéma plus simple que celui de l'Assemblée Nationale. Ce sont ces fichiers qui sont utilisés par la suite dans le système.

Objectifs

L'objectif de l'étude jointe de ces données est l'extraction de relations entre les députés et représentants d'intérêts. Celles-ci sont en effet très souvent cachées ou impossibles à détecter puisque les informations ne sont pas rendues publiques. L'idée qui a sous tendu le travail présenté ici est que même si ces relations ne peuvent être directement obtenues par des sources classiques, elles peuvent peut être l'être par des sources indirectes mais donc uniquement indicatives.

Le succès de cette approche permettrait de démontrer que ce type d'outils pourrait permettre à des journalistes de guider leurs investigations grâce l'exploitation conjointe de multiples sources de données hétérogènes.

Dans cette étude, trois axes ont été exploités pour mettre au jour des liens entre députés et représentants d'intérêts :

— **Biographie des députés** Presque tous les députés de la XVème législature disposent d'un profil Wikipedia, et donc peuvent être associés à une instance dans Wikipedia, Wikidata, DBPedia, etc. (voir 1.4.2 et 2.2.1).
Ce « profil » contient éventuellement des informations concernant les emplois précédents ou autres activités d'un député qui ne sont pas présentes dans leur fiche d'information standard. Une première approche très rudimentaire pour lier des députés avec avec des représentants d'intérêt par leur page Wikipedia serait par exemple d'extraire toute mention d'un nom de représentant et de

11. REST (Representational State Transfer) est un protocole très répandu pour l'échange entre une composante logicielle (typiquement service Web) disponible online et ses utilisateurs ; elle se base sur le protocole HTTP et permet ainsi extension de mettre à disposition des données par des URLs.

12. Toutes les informations nécessaires à l'utilisation des APIs, le code source et les différents services mis en place par Regards citoyens sont disponibles sur le profil Github du collectif https://github.com/regardscitoyens/

la marquer comme exemple positif. L'utilisation d'un procédé comme le TF-IDF permet d'assigner un score aux occurrences des nom de représentants afin d'éviter les noms trop fréquents ou qui peuvent représenter autre chose qu'une relation (p. ex. « Google »)

— **Champ Lexical** Dans le registre de la HATVP, les représentants d'intérêt déclarent leur secteur d'activité ou d'intérêt. Chaque secteur peut être associé à un champ lexical. Pour les députés et sénateurs, les deux sites `nosdeputes.fr` et `nossenateurs.fr` calculent et extraient une trentaine de mots représentatifs du registre de ce parlementaire. Ces mots sont extraits par TF-IDF sur toutes les questions écrites et orales, ainsi que les interventions au Sénat ou à l'Assemblée de ce parlementaire.
Ces deux ensembles de mots peuvent être utilisés pour obtenir un score de correspondance entre les intérêts d'un députés et ceux d'un représentant d'intérêt. Cette recherche part de l'hypothèse selon laquelle un député aura probablement des relations avec les représentants d'intérêts qui correspondent à son champ d'action.

— **Comptes Twitter** Pour les représentants d'intérêts et les parlementaires, les registres ouverts communiquent un compte Twitter associé si il existe. Ce compte permet d'obtenir de manière rapide et relativement aisée un assez large corpus de communications publiques, plus informelles que les communications habituelles dans d'autres médias et qui sont potentiellement plus représentatives des opinions de chacun des ces acteurs. Pendant ce travail, un corpus de 140000 Tweets ainsi que de Retweeters pour chacun d'eux, totalisant environ 145000 utilisateurs distincts et s'étalant sur une période de janvier 2018 à début mai 2018 [13].
Pour obtenir un score à partir du nombre des tweets et retweeters, sont envisagées :

1. Étudier les relations directes x FOLLOWS y qui peuvent indiquer une relation.
2. Utiliser des mesures de similarités comme la mesure de Jaccard sur les ensembles de retweeters entre un député et un représentant d'intérêts. $\left|\frac{RT_A \cap RT_B}{RT_A \cup RT_B}\right|$ où RT_X représente l'ensemble des retweeters de l'entité X.

D'autres méthodes de mesure d'influence dans des graphes peuvent être envisagées afin d'extraire les nœuds influents ainsi que leur sphère d'influence de manière plus fine.

Le schéma Figure 3.3 illustre l'architecture globale, présente le schéma dans lequel s'inscrivent les différentes entités et met en évidence les différents points de liaison envisagés.

Le schéma s'articule avec quatre sources de données :

— Les données de `nosdeputes.fr` où chaque parlementaire dispose d'informations à son sujet.
— Les données de `nossenateurs.fr` où chaque parlementaire dispose d'informations à son sujet.
— Le document de la HATVP qui comprend une liste des représentants d'intérêts (lobbies et cabinets de lobby) ainsi que des informations sur eux et leurs activités.

13. Ces données sont extraites par un script automatique qui récupère les Tweets par ordre chronologique. Ces chiffres sont valables à la date du 19/06/2018.

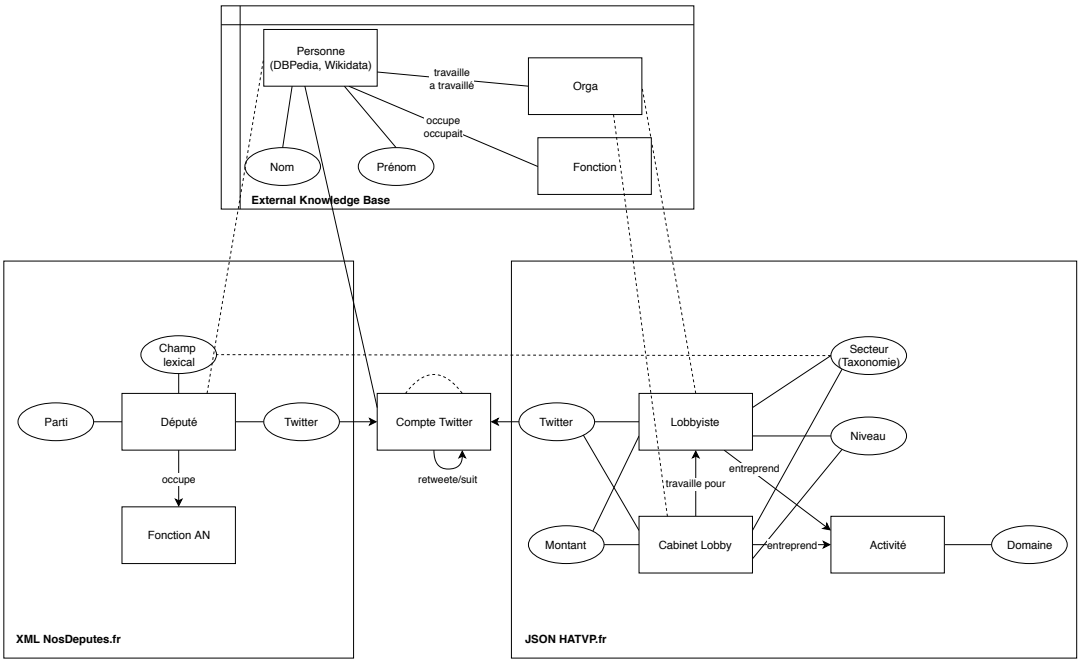

FIGURE 3.3 – Description des données impliquées le cas d'étude des relations entre parlementaires et représentants d'intérêts. Les lignes en pointillés tracent les liens potentiels à découvrir entre des objets de données de différentes bases.

— Les comptes Twitter des députés et obbyistes ainsi que des informations venant de bases de connaissances extérieures qui permettent d'enrichir les données initiales.

Après mise en place des composantes nécessaires à chaque étape dans un fichier de spécifications, la compilation donne le résultat qui est présenté dans l'illustration de la Figure 3.4 [14].

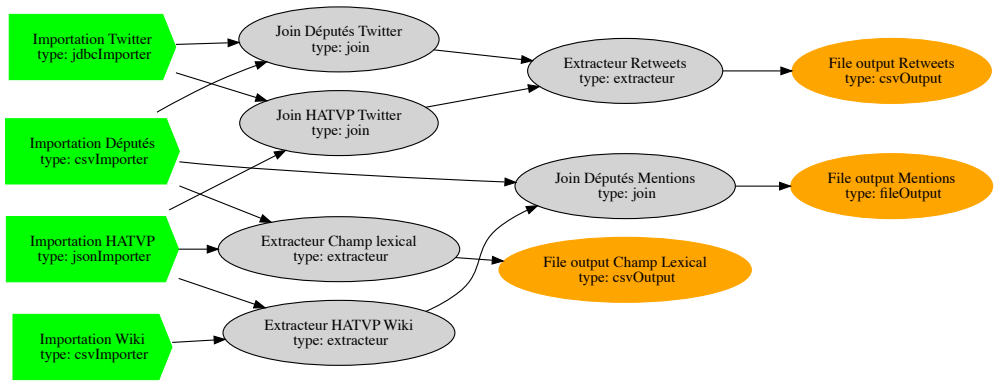

FIGURE 3.4 – Graphe des opérations nécessaires pour la mise en place de l'étude de cas sur les parlementaires et les représentants d'intérêts. Il a été produit pendant la compilation du fichier de spécification.

14. Le graphe Figure 3.4 a été produit pendant l'étape de compilation durant laquelle le programme construit un « graphe de calcul », la librairie de visualisation de graphes Graphviz (https://graphviz.gitlab.io/) a été utilisée.

3.3.2 Second cas d'étude envisagé : Débats sur le changement climatique

Le changement climatique est un sujet qui donne lieu à la diffusion de beaucoup d'informations fausses, partiellement fausses ou mal présentées, notamment au État-Unis où la question de la réalité du changement climatique reste un débat.

Climate Feedback est un institut de recherche qui regroupe un réseau de scientifiques travaillant sur le climat dans plusieurs pays. Ils travaillent à analyser et annoter avec des liens vers des articles scientifiques les articles de presse influents qui traitent de la question du changement climatiques ou de questions associées. Il en résulte un corpus de données annotées traitant du changement climatique avec des noms de scientifiques et les informations qu'ils ont contribuées.

Le Climate negotiations browser est une application web mise en place par le médialab de Sciences-Po Paris, le LSIR de l'EPFL et l'atelier Iceberg à Nantes qui permet d'explorer vingt ans de discussions internationales au cours de différents sommets et grandes conférences à propos du changement climatique publiées dans le Earth Negotiation Bulletin.

Ces deux sources de données indépendantes permettraient peut être de mettre en place un nouveau cas d'étude dont l'objectif serait de cartographier les relations entre les scientifiques, leurs propos et positions dans le domaine du changement climatique ainsi que les États et organisations internationales dans le domaine. Un second cas d'étude permettrait de confirmer l'intérêt de l'architecture proposée pour de multiples applications et sa capacité à s'adapter à différent types de données et différent types de tâches.

Le cas d'étude a été étudié et partiellement implémenté avec un langage de script afin d'explorer les possibilités qu'offaient les sources de données présentées ci-dessus. Après plusieurs essais, il a semblé que le projet ne pourrait être suffisamment avancé pendant le temps restant du stage pour obtenir des résultats satisfaisants. Ceci est principalement dû aux sources de données qui, bien qu'accessibles aisément par le web, n'exposent pas d'interface accessible de manière automatisée. Ceci implique donc que leur exploitation doit être effectuée via des scripts d'extraction à partir du contenu des pages web elles-mêmes. Ce travail nécessitant un temps supplémentaire, l'implémentation complète du cas d'étude n'a pas pu être effectuée.

3.4 Conclusion et possibles futurs développements

Ce travail fut achevé fin Août 2018 et pourra faire l'objet d'apports supplémentaires dans le futur. Dans le cas où le projet serait pris en charge par une autre personne, j'ai communiqué quelques avis concernant les axes prioritaires dans l'ajout et le développement de nouvelles fonctionnalités pour ce logiciel.

Celles-ci s'appuient sur trois directions importantes :

— La création d'une interface graphique pour l'utilisateur final. Comme ce projet tente de construire un outil utilisable par des journalistes permettant à ceux-ci d'utiliser des techniques élaborées d'analyse de données, il doit aussi être suffisamment ergonomique pour être utilisable. Une interface graphique permettant à un utilisateur d'interagir directement avec la représentation interne de la série d'opérations comme graphe acyclique dirigé serait une manière intéressante de bien rendre compte à la fois de l'orde et l'interdépendance des opérations que le dit utilisateur définirait.

— L'augmentation et l'ajout de modules. Les opérations pouvant être implémentées sont extrêmement nombreuses et peuvent donner lieu à l'utilisation de toutes sortes de techniques et d'outils supplémentaires. Le développement

de ces modules devra être fait en concordance avec les éventuels besoin que pourraient avoir les personnes utilisant le logiciel.

— L'ajout de tests et mécanismes facilitant la maintenance du code ainsi que l'installation du logiciel. Ce travail correspond plus à celui d'un développeur système, puisqu'il implique une connaissance fine de la manière de tester et de rendre l'utilisation du logiciel plus ergonomique. Pour l'instant, quelques test unitaires existent mais ils ne couvrent pas tous les modules disponibles.

A Réseaux logiques de Markov

Les réseaux logiques de Markov (RICHARDSON et DOMINGOS, 2006 ; DOMINGOS et LOWD, 2009) sont une tentative d'unification de la logique du premier ordre avec les modèles graphiques probabilistes dans un formalisme unique. Ceci a pour but de permettre un "assouplissement" des règles de la logique du premier ordre et leur utilisation dans un cadre de modèles probabilistes.

Dans un modèle de Markov représentant la distribution jointe de l'ensemble de variables aléatoires $X = (X_1, X_2, ..., X_N) \in \mathcal{X}$ où X vérifie :

— G est un graphe non dirigé où chacun des X_i est un nœud.

— Pour chaque clique du graphe, il y a un fonction de potentiel ϕ_k qui est une fonction à valeurs réelles non négative qui est une fonction de l'état de la clique.

La distribution jointe représentée par le réseau de Markov est alors

$$P(X = x) = \frac{1}{Z} \prod_k \left(\phi_k(x_{\{k\}})\right)$$

où $x_{\{k\}}$ est l'état des variables de la clique k et Z la fonction de partition est égale à $\sum_{x \in \mathcal{X}} \prod_k \left(\phi_k(x_{\{k\}})\right)$

Dans un réseau logique de Markov, le potentiel de clique est remplacé par l'exponentielle d'une somme pondérées de caractéristiques :

$$P(X = x) = \frac{1}{Z} \exp\left(\sum_j w_j f_j(x)\right)$$

où les fonction f_j peuvent être n'importe quelle fonction à valeur réelle de l'état, en particulier un fonction binaire : $f_j(x) \in \{0, 1\}$. L'inférence dans ce type de réseaux est #P-complet, c'est à dire que tout problème de comptage en temps non polynomial peut être réduit en temps polynomial à celui-ci et il est par exemple équivalent au problème qui consiste à compter le nombre de réponses positive à un problème de satisfaction booléenne (#-SAT).

Il existe pour ce problème des méthodes d'inférence efficace comme l'échantillonnage de Gibbs qui ne sera pas détaillé ici.

Un réseau logique de Markov L est un ensemble de paires (F_i, w_i) où F_i est une formules de logique du premier ordre et w_i un nombre réel. Le réseau est ensuite définit de la manière suivante :

1. Chaque réalisation de chaque prédicat de L correspond à 1 nœud binaire qui vaut 1 si le prédicat est vrai et 0 sinon. (Une réalisation d'un prédicat est une formule atomique sans variable associée p. ex. `sontMarriés`('Barack', 'Michelle'))
2. Pour chaque possible réalisation de chaque formule F_i de L, une caractéristique est associée qui vaut 1 si la formule est vrai et 0 sinon. Le poids de cette caractéristique est w_i

3. Le réseau a une arête entre deux nœuds si et seulement si les prédicats correspondants apparaissent ensemble dans une formule de L

B Path Ranking Algorithm

Le *Path Ranking Algorithm* ou "Algortihme de classement de chemins" (LAO et COHEN, 2010) est un algorithme de recommendation développé par Ni Lao et William W. Cohen.

L'intuition derrière cet algorithme qui permet de classer les paires susceptibles de correspondre à une certaine raltion est la suivante : deux entités ayant une relation donnée auront probablement un certain nombre d'autres relations différentes (d'ordre potentiellement supérieur à 1) qui les lient.

Par exemple, un tel algorithme pourrait prédire que deux entités (X, Y) sont mariées car elles sont liées par une relation de parenté avec une troisième entité.

Pour apprendre ces différentes règles, l'algorithme exécute une marche aléatoire sur le graphe de relations. Cela permet de construire un modèle linéaire qui représente l'importance relative de chaque relation tierce entre deux entités pour chaque relation cible que l'on souhaite prédire.

Bibliographie

AUER, Sören et al. (2007). « DBpedia : A nucleus for a Web of open data ». In : *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. T. 4825 LNCS, p. 722–735. ISBN : 3-540-76297-3. DOI : 10.1007/978-3-540-76298-0_52.

BACH, Stephen H. et al. (2017). « Learning the Structure of Generative Models without Labeled Data ». In : *arXiv :1703.00854 [cs, stat]*. arXiv : 1703.00854. (Visité le 16/04/2018).

BERNERS-LEE, Tim (1999). *Weaving the Web : The Original Design and Ultimate Destiny of the World Wide Web by Its Inventor*. Harper San Francisco. ISBN : 978-1-4028-4293-1.

BOLLACKER, Kurt et al. (2008). « Freebase : a collaboratively created graph database for structuring human knowledge ». In : *SIGMOD 08 Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, p. 1247–1250. ISSN : 07308078. DOI : 10.1145/1376616.1376746.

BONAQUE, Raphaël et al. (2016). « Mixed-instance querying : a lightweight integration architecture for data journalism ». In : *PVLDB* 9.13, p. 1513–1516.

BORDES, Antoine et al. (2013). « Translating Embeddings for Modeling Multi-Relational Data ». In : *Advances in NIPS* 26, p. 2787–2795. ISSN : 10495258. DOI : 10.1007/s13398-014-0173-7.2.

CAFARELLA, Michael J., Alon HALEVY et Jayant MADHAVAN (2011). « Structured data on the web ». In : *Communications of the ACM* 54.2, p. 72. ISSN : 00010782. DOI : 10.1145/1897816.1897839.

CARLSON, Andrew, Justin BETTERIDGE et Bryan KISIEL (2010). « Toward an Architecture for Never-Ending Language Learning. » In : *In Proceedings of the Conference on Artificial Intelligence (AAAI) (2010)*, p. 1306–1313. ISSN : 1098-2345. DOI : 10.1002/ajp.20927.

CHAMBERS, Nathanael (2011). « Template-Based Information Extraction without the Templates ». en. In : p. 11.

— (2013). « Event Schema Induction with a Probabilistic Entity-Driven Model ». en. In : p. 11.

CHEN, Muhao et al. (2017). « Multilingual knowledge graph embeddings for cross-lingual knowledge alignment ». In : *IJCAI International Joint Conference on Artificial Intelligence*, p. 1511–1517. ISBN : 978-0-9992411-0-3. DOI : 10.24963/ijcai.2017/209.

CHEUNG, Jackie Chi Kit, Hoifung POON et Lucy VANDERWENDE (2013). « Probabilistic Frame Induction ». en. In : p. 10.

DOMINGOS, Pedro et Daniel LOWD (2009). *Markov Logic : An Interface Layer for Artificial Intelligence*. 1st. Morgan et Claypool Publishers. ISBN : 978-1-59829-692-1.

DONG, Xin Luna et al. (2015). « From Data Fusion to Knowledge Fusion ». In : ISSN : 21508097. DOI : 10.14778/2732951.2732962.

DONG, Xin et al. (2014). « Knowledge vault : a web-scale approach to probabilistic knowledge fusion ». In : *Proceedings of the 20th ACM SIGKDD international*

conference on Knowledge discovery and data mining - KDD '14, p. 601–610. ISSN : 0893-6080. DOI : 10.1145/2623330.2623623.

EHRLINGER, Lisa et Wolfram WÖSS (2016). « Towards a Definition of Knowledge Graphs ». en. In : p. 4.

ETZIONI, Oren et al. (2011). « Open Information Extraction : The Second Generation ». en. In : p. 8.

FADER, Anthony, Stephen SODERLAND et Oren ETZIONI (2011). « Identifying Relations for Open Information Extraction ». In : *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Edinburgh, Scotland, UK. : Association for Computational Linguistics, p. 1535–1545. (Visité le 12/04/2018).

GALÁRRAGA, Luis Antonio et al. (2013). « AMIE : association rule mining under incomplete evidence in ontological knowledge bases ». en. In : ACM Press, p. 413–422. ISBN : 978-1-4503-2035-1. DOI : 10.1145/2488388.2488425. (Visité le 12/06/2018).

GOODE, Luke (2009). « Social news, citizen journalism and democracy ». en. In : *New Media & Society* 11.8, p. 1287–1305. ISSN : 1461-4448, 1461-7315. DOI : 10.1177/1461444809341393. (Visité le 18/06/2018).

IC, Denny Vrandec et Markus KRÖTZSCH (2014). « Wikidata : A Free Collaborative Knowledge Base ». en. In : p. 7.

International Consortium of Investigative Journalists (2018). en-US. (Visité le 07/06/2018).

IVES, Zachary G. et al. (1999). « An Adaptive Query Execution System for Data Integration ». In : *SIGMOD 1999, Proceedings ACM SIGMOD International Conference on Management of Data, June 1-3, 1999, Philadelphia, Pennsylvania, USA*. P. 299–310. DOI : 10.1145/304182.304209.

JARKE, Matthias (2003). *Fundamentals of data warehouses, 2nd Edition*. Springer. ISBN : 3540420894. URL : http://www.worldcat.org/oclc/49824734.

JIANG, Shangpu, Daniel LOWD et Dejing DOU (2012). « Learning to Refine an Automatically Extracted Knowledge Base Using Markov Logic ». en. In : IEEE, p. 912–917. ISBN : 978-1-4673-4649-8 978-0-7695-4905-7. DOI : 10.1109/ICDM.2012.156. (Visité le 24/04/2018).

LACOSTE-JULIEN, Simon et al. (2013). « SiGMa : Simple Greedy Matching for Aligning Large Knowledge Bases ». In : *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '13*, p. 572. ISSN : 9781450321747. DOI : 10.1145/2487575.2487592.

LAO, Ni et William W. COHEN (2010). « Relational retrieval using a combination of path-constrained random walks ». en. In : *Machine Learning* 81.1, p. 53–67. ISSN : 0885-6125, 1573-0565. DOI : 10.1007/s10994-010-5205-8. (Visité le 13/06/2018).

LIN, Hailun et al. (2017). « Learning Entity and Relation Embeddings for Knowledge Resolution ». In : *Procedia Computer Science*. T. 108, p. 345–354. ISBN : 978-1-57735-701-8. DOI : 10.1016/j.procs.2017.05.045.

MADHAVAN, Jayant et al. (2007). « Web-scale Data Integration : You can only afford to Pay As You Go ». In : *Cidr 2007* 7, p. 342–350.

MIN, Bonan et al. « Distant Supervision for Relation Extraction with an Incomplete Knowledge Base ». en. In : p. 6.

MINTZ, Mike et al. (2009). « Distant supervision for relation extraction without labeled data ». In : *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP : Volume 2 - ACL-IJCNLP '09* 2.2005, p. 1003. ISSN : 1932432469. DOI : 10.3115/1690219.1690287.

NGUYEN, Kiem-Hieu et al. (2015). « Generative Event Schema Induction with Entity Disambiguation ». en. In : Association for Computational Linguistics, p. 188–197. DOI : 10.3115/v1/P15-1019. (Visité le 19/06/2018).

NICKEL, Maximilian, Kevin MURPHY et al. (2016). *A review of relational machine learning for knowledge graphs*. T. 104. 1. ISBN : 1-08-980130-0. DOI : 10.1109/JPROC.2015.2483592.

NICKEL, Maximilian, Volker TRESP et Hans-Peter KRIEGEL (2011). « A Three-Way Model for Collective Learning on Multi-Relational Data ». en. In : *Proceedings of the 28th International Conference on Machine Learning*, p. 8.

NIU, Feng, Christopher RÉ et al. (2011). « Tuffy : Scaling up Statistical Inference in Markov Logic Networks using an RDBMS ». In : *Proceedings of the VLDB Endowment* 4.6, p. 373–384. ISSN : 2150-8097. DOI : 10.14778/1978665.1978669.

NIU, Feng, Ce ZHANG et al. (2012). « Elementary : Large-scale Knowledge-base Construction via Machine Learning and Statistical Inference ». In : *International Journal on Semantic Web and Information Systems (IJSWIS)* 8.3, p. 42–73. ISSN : 15526283. DOI : 10.4018/jswis.2012070103.

NORMAND, Grégoire (2018). *Transparence : cinq choses à savoir sur les lobbies en France*. fr. (Visité le 18/06/2018).

PUJARA, Jay et al. (2013). « Knowledge Graph Identification ». en. In : *The Semantic Web – ISWC 2013*. T. 8218. Berlin, Heidelberg : Springer Berlin Heidelberg, p. 542–557. ISBN : 978-3-642-41334-6 978-3-642-41335-3. DOI : 10.1007/978-3-642-41335-3_34. (Visité le 24/04/2018).

RATNER, Alexander et al. (2017). « Snorkel : Rapid Training Data Creation with Weak Supervision ». In : *Proceedings of the VLDB Endowment* 11.3. arXiv : 1711.10160, p. 269–282. ISSN : 21508097. DOI : 10.14778/3157794.3157797. (Visité le 16/04/2018).

RICHARDSON, Matthew et Pedro DOMINGOS (2006). « Markov logic networks ». en. In : *Machine Learning* 62.1-2, p. 107–136. ISSN : 0885-6125, 1573-0565. DOI : 10.1007/s10994-006-5833-1. (Visité le 20/06/2018).

SUCHANEK, Fabian M., Serge ABITEBOUL et Pierre SENELLART (2011). « PARIS : Probabilistic Alignment of Relations, Instances, and Schema ». In : p. 157–168. ISSN : 21508097. DOI : 10.14778/2078331.2078332.

SUCHANEK, Fabian M, Gjergji KASNECI et Gerhard WEIKUM (2007). « YAGO : a core of semantic knowledge ». In : *Proceedings of the 16th international conference on World Wide Web*, p. 697–706. ISSN : 01695347. DOI : 10.1145/1242572.1242667.

TOMASIC, Anthony, Louiqa RASCHID et Patrick VALDURIEZ (1998). « Scaling Access to Heterogeneous Data Sources with DISCO ». In : *IEEE Trans. Knowl. Data Eng.* 10.5, p. 808–823. DOI : 10.1109/69.729736.

TROUILLON, Théo et al. (2016). « Complex Embeddings for Simple Link Prediction ». In : *arXiv :1606.06357 [cs, stat]*. arXiv : 1606.06357. (Visité le 16/04/2018).

WANG, Quan et al. (2016). « Knowledge Base Completion via Coupled Path Ranking ». en. In : Association for Computational Linguistics, p. 1308–1318. DOI : 10.18653/v1/P16-1124. (Visité le 17/04/2018).

WIEDERHOLD, Gio (1992). « Mediators in the Architecture of Future Information Systems ». In : *IEEE Computer* 25.3. DOI : 10.1109/2.121508.

YATES, Alexander et al. (2007). « TextRunner : Open Information Extraction on the Web ». In : *Proceedings of Human Language Technologies : The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*. Rochester, New York, USA : Association for Computational Linguistics, p. 25–26. (Visité le 12/04/2018).

ZHANG, Ce et al. (2017). « DeepDive : Declarative Knowledge Base Construction ». In : *Communications of the ACM* 60.5, p. 93–102. ISSN : 0163-5808. DOI : 10.1145/2949741.2949756.