Construction de patrons lexico-syntaxiques d'extraction pour l'acquisition de connaissances à partir du web

Chloé Monnin[1] Olivier Hamon[1,]
(1) Syllabs, 35-37 rue Chanzy, 75011 Paris, France
monnin@syllabs.com, hamon@syllabs.com

RESUME

Cet article présente une méthode permettant de collecter sur le web des informations complémentaires à une information prédéfinie, afin de remplir une base de connaissances. Notre méthode utilise des patrons lexico-syntaxiques, servant à la fois de requêtes de recherche et de patrons d'extraction permettant l'analyse de documents non structurés. Pour ce faire, il nous a fallu définir au préalable les critères pertinents issus des analyses dans l'objectif de faciliter la découverte de nouvelles valeurs.

ABSTRACT

Relation pattern extraction and information extraction from the web.
This article presents an information extraction method which collects additional information on the web so as to enrich already existing information and then fill in a knowledge base. Our method is based on lexical and syntactical patterns, both used as search queries and extraction patterns to allow the analysis of unstructured documents. To do so, we first defined relevant criteria coming from the analysis phase so as to ease the discovery of new values.

MOTS-CLES : Construction de patrons, extraction d'information, extraction d'entités nommées, syntaxe en dépendances, apprentissage de patrons d'extraction, web comme corpus.

KEYWORDS: Pattern construction, information extraction, named entities recognition, dependancy syntax, extraction pattern learning, web as a corpus.

1 Introduction

L'utilisation du web comme corpus d'analyse n'est pas une nouveauté. En effet, avec l'essor des moteurs de recherche dans la seconde moitié des années 90, l'exploration des contenus analysables linguistiquement était facilitée. Cependant, malgré le formatage requis pour l'interprétation des pages web par le navigateur ou l'apparition de standards tels que *schema.org*[1], les textes accessibles *via* un moteur de recherche restent du texte 'libre', contenant des informations, mais qui ne peuvent être interprétées directement par la machine autrement que comme des chaînes de caractères. L'analyse de ces textes devient par conséquent une nécessité pour en extraire de l'information en contexte, notamment pour la mettre en relation avec d'autres informations.

Nous proposons ici une méthode permettant de construire à partir du web des patrons lexico-syntaxiques d'extraction, ainsi que leur application pour extraire des informations similaires et/ou complémentaires à une information donnée.

[1] http://schema.org

Nos travaux se placent dans un contexte d'enrichissement de base de connaissances à Syllabs, dans le domaine de la génération automatique de textes où le besoin de nouvelles données est indispensable et se fait croissant. Si des bases de données existent pour de nombreuses thématiques, elles sont bien souvent incomplètes, voire erronées. De plus, la mise à disposition des bases prend du temps, impactant l'accès à ces données. Ainsi, il nous paraît nécessaire d'aller chercher de l'information là où elle se trouve, et en particulier sur le web. La masse de documents accessibles *via* les moteurs de recherche laisse penser que, dans leur grande majorité, les données les plus fréquentes seront correctes et, quand bien même le doute subsisterait, une validation semi-manuelle est toujours envisageable. Mais avant toute chose, les données doivent être extraites.

Le choix de travailler à l'aide de patrons d'extraction est motivé par l'aspect linguistique de nos travaux. En effet, les patrons résultant de notre implémentation ne sont pas seulement un ensemble de traits permettant l'extraction d'informations, mais aussi des représentations linguistiques de la formulation d'une relation entre deux valeurs (ou plus), apportant de la connaissance supplémentaire et qui pourrait permettre d'étudier la variabilité linguistique par exemple. Une autre application de ces patrons comme outil d'extraction d'entités nommées peut également être d'extraire de nouvelles entités qui ne sont pas connues actuellement.

Nous allons dans un premier temps revenir sur différentes méthodes d'extraction présentes dans la littérature, puis présenter notre méthode. Nous fournirons ensuite nos résultats sur une extraction à partir d'une paire de valeurs, ainsi que des résultats d'évaluation. Nous conclurons sur les optimisations et extensions envisagées pour l'amélioration de la méthode.

2 Etat de l'art

L'extraction d'informations a toujours été un domaine prédominant dans le traitement automatique du langage. Cette tâche consiste en la détection et l'extraction sous forme structurée, de données présentes dans des documents non structurés. Le développement des méthodes d'extraction a été stimulé par les conférences MUC (*Message Understanding conferences*) dès la fin des années 80, en organisant des compétitions financées par la DARPA (Grishman & Sundheim, 1996). L'objectif de ces conférences était d'extraire le plus d'informations possible sur des thèmes bien déterminés et d'évaluer les systèmes d'extraction d'informations selon une grille d'évaluation commune. En dehors de ces conférences, la communauté scientifique a développé d'autres méthodes, utilisant notamment l'extraction de relations à l'aide de patrons.

Brin (1998) avec son système DIPRE (*Dual Iterative Pattern Relation Extraction*) a proposé une méthode permettant d'exploiter le potentiel de la multiplicité de sources d'informations présentes sur le web. Cette méthode génère des patrons à partir d'exemples de valeurs fournies en entrée par l'utilisateur. Les patrons extraits sont ensuite utilisés pour rechercher de nouvelles instances de la relation que le patron représente. Agichtein & Gravano (2000) approfondissent le travail de Brin avec Snowball. Leurs contributions apportent entre autres une proposition d'évaluation du système ainsi qu'une méthode pour mesurer le poids des patrons extraits. Cependant, leur méthode est adaptée à des collections de documents hors web.

Les méthodes présentées ci-dessus sont supervisées, dans le sens où les outils reçoivent en entrée des valeurs définies préalablement. Que ce soit simplement un couple de valeurs pour DIPRE ou une représentation formelle d'une relation, cette entrée est le point de départ de la méthode.

Les méthodes suivantes sont semi-supervisées, puisque leur point d'entrée n'est pas une valeur mais un corpus. En effet, contrairement aux méthodes ci-dessus, celles-ci ont pour but d'extraire toutes les relations au sein d'un corpus défini, et non une relation particulière servant d'amorce.

Lin & Pantel (2001) font l'hypothèse distributionnelle pour développer DIRT, dont le but est d'extraire des relations d'inférence (dans ce cas précis des paraphrases). Leur méthode permet d'extraire ce qu'ils nomment des *slot fillers*, soit des entités qui apparaissent dans les trous laissés par leur patrons à partir d'arbres syntaxiques en dépendances, de règles de comparaison et d'un calcul de similarité afin d'identifier les paraphrases. Etzioni et al. (2004) publient les premiers résultats de leur système KnowItAll, un outil permettant d'extraire ce qu'ils nomment des faits, soit des informations mises en relations. Leurs travaux se poursuivront les années suivantes avec KnowItNow (Cafarella et al., 2005). Leur méthode s'appuie sur des clauses, soient des règles prédéfinies pour extraire des entités qui répondent à ces règles. Les améliorations de KnowItNow concernent principalement la rapidité d'exécution de l'outil ainsi que l'apprentissage de règles d'extraction. Etzioni et al. (2011) approfondissent leur système et développent Reverb, un système d'extraction de relations basées sur les verbes. Ce système applique un filtre syntaxique reposant sur des patrons morphosyntaxiques (des expressions régulières de parties du discours) et un filtre lexical. Plus récemment, Akbik et al. (2012) ont étendu l'extraction de relation non supervisée en extrayant toutes les relations d'un corpus puis en les classant par clustering.

Les travaux cités ci-dessus sont implémentés pour des contenus en anglais. Pour un contexte français, il est possible de trouver des méthodes similaires appliquées à des documents techniques, comme par exemple le système Prométhée (Morin, 1999) qui extrait des schémas lexico-syntaxiques représentatifs d'une relation sémantique.

Les méthodes d'extraction de relations et d'informations à partir de patrons sont très développées au sein des méthodes d'extraction en général. Dans la conception de notre outil, nous nous sommes beaucoup inspirés des travaux que nous évoquons ci-dessus. Notre méthode utilise aussi bien des analyses syntaxiques en dépendances (Akbik) que des patrons d'extraction (Etzioni, Agichtein, Morin). De plus notre corpus d'extraction est également le web, comme (Brin).

3 Présentation de la méthode

Nous décrivons dans un premier temps la méthode de construction de patrons d'extractions ainsi que les outils utilisés, en prenant l'exemple des valeurs d'entrée suivantes : "*Victor Hugo*" et "*26 février 1802*", soit un nom et une date de naissance. Victor Hugo étant un personnage dont la vie a bien été documentée, nous avons estimé qu'il ferait un bon point de départ pour extraire différentes façons de renseigner une date de naissance. Puis, nous décrivons la méthode d'extraction de nouvelles valeurs à partir de ces patrons. La Figure 1 ci-dessous résume l'ensemble de la méthode.

3.1 Construction des patrons

La méthode de construction des patrons d'extraction se déroule selon plusieurs étapes. Tout d'abord, nous utilisons les valeurs d'entrée comme requête de moteur de recherche afin de recueillir des contenus textuels. Ces textes sont ensuite segmentés en phrases, qui sont filtrées selon la pertinence de l'information présente. Les phrases sélectionnées sont ensuite analysées syntaxiquement et les patrons sont construits à partir de ces analyses. Nous détaillons chacune de ces étapes ci-dessous.

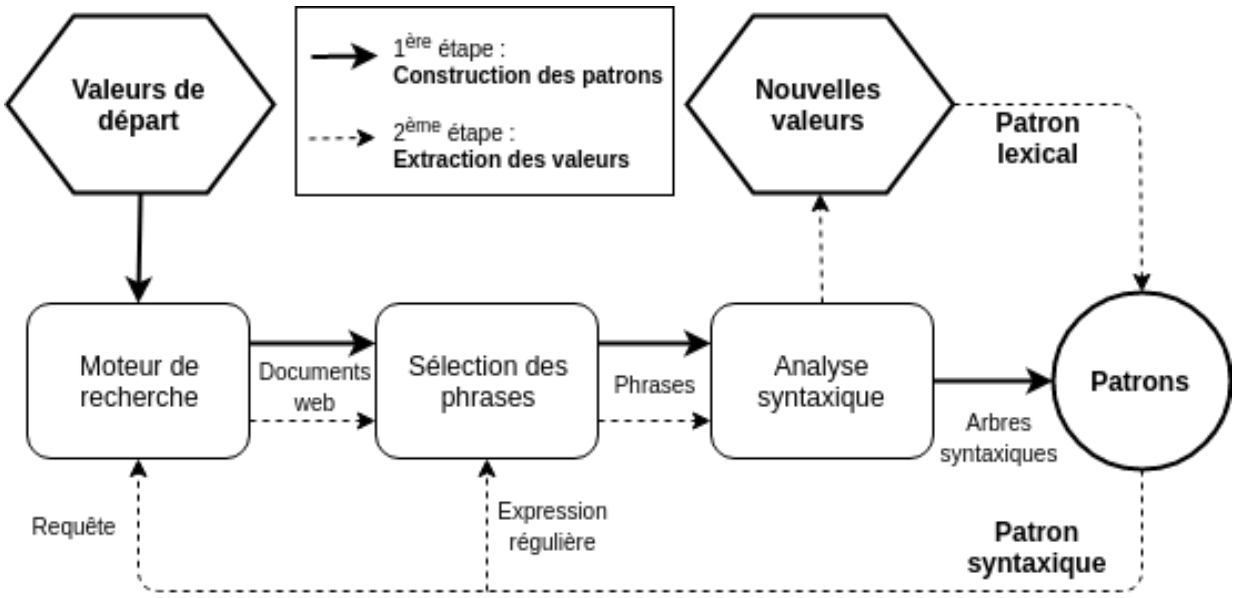

FIGURE 1: Schéma de notre méthode

3.1.1 Collecte de contenus sur le Web

La première étape consiste à rassembler des documents hétérogènes à partir desquels nous construirons les patrons. Les documents sont collectés sur le web : des valeurs passées en entrée ("Victor Hugo" et "2 février 1802") sont transformées en requêtes adaptées au moteur de recherche. La requête est ensuite envoyée à un moteur de recherche qui retourne des URLs (100 dans le cadre de nos tests). Pour extraire le texte utile des pages web, nous utilisons un extracteur appliquant l'algorithme BTE (Fin et al., 2001). Au terme de cette extraction, nous obtenons une série de contenus textuels et leurs URLs associées.

3.1.2 Sélection des phrases

Le texte que nous avons collecté lors de la précédente étape est segmenté en phrases, qui sont ensuite triées. Pour la segmentation nous utilisons le tokenizer PunktTokenizer (Kiss & Strunk, 2006) tel qu'il est implémenté dans le package NLTK. Nous sélectionnons ensuite les phrases d'après deux critères : la présence exacte de toutes les valeurs recherchées et la longueur de la phrase qui doit faire moins de 200 tokens pour être analysée.

Ce dernier critère est lié à la quantité d'informations que l'analyseur syntaxique peut traiter. Au-delà de 200 tokens, les relations à analyser requièrent trop de mémoire. Par ailleurs, après observation du corpus, ces séquences ne sont souvent pas pertinentes par rapport à ce que nous souhaitons extraire. La segmentation en phrases ne s'appliquant pas par exemple aux listes ou aux contenus de tableaux, ces séquences sont souvent de longues énumérations qui ne contiennent pas d'informations exprimant une relation explicite.

3.1.3 Analyse syntaxique des phrases sélectionnées

L'analyse syntaxique des phrases sélectionnées lors de la phase précédente est ensuite réalisée à l'aide de Talismane (Urieli, 2013), un analyseur syntaxique en dépendances. L'arbre syntaxique correspondant à une phrase est une liste ordonnée de nœuds possédant chacun des attributs tels que :

- l'identifiant du nœud correspondant à sa position dans la phrase,
- le token : la forme lexicale d'un mot dans la phrase,
- le lemme : la forme non fléchie du token,
- la partie du discours,
- les traits morphosyntaxiques du token : genre, nombre, personne, etc.,
- l'identifiant du token gouverneur,
- le lien de dépendance entre le token et son gouverneur.

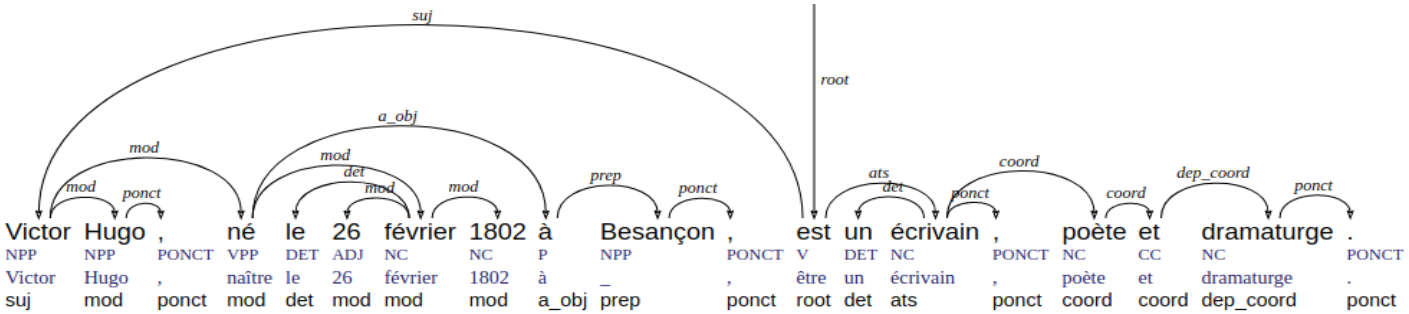

FIGURE 2 : Exemple d'arbre syntaxique[2]

Pour créer les patrons nous utilisons plusieurs de ces traits, à savoir les parties du discours, les liens de dépendances et la forme du token. Les liens de dépendances entre les valeurs présentes dans la phrase vont nous permettre d'établir une relation entre ces valeurs, tandis que les parties du discours et les formes lexicales nous serviront à la construction du patron en tant que tel.

3.1.4 Sélection des relations

Les relations entre les valeurs des arbres syntaxiques obtenus précédemment sont sélectionnées afin de créer les patrons. Nous considérons qu'une relation est présente dans une phrase s'il existe un chemin reliant les valeurs de départ. Chaque arbre est analysé comme un graphe non orienté[3], où chaque nœud correspond à un token de la phrase. Nous parcourons les nœuds et déterminons le chemin le plus court entre les valeurs de départ.

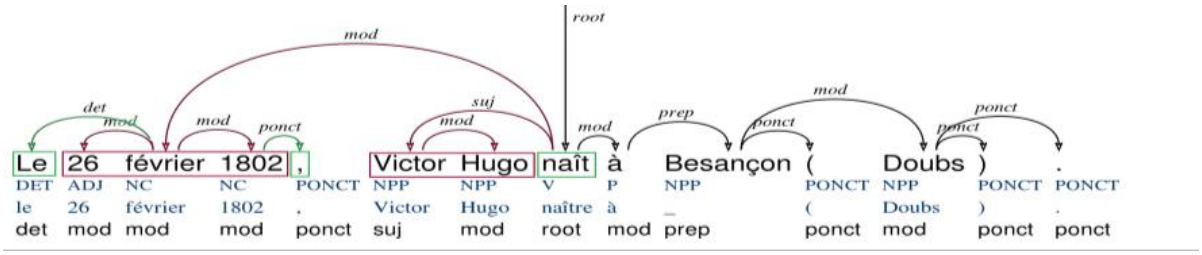

FIGURE 3 : Extraction du chemin
En rouge le chemin le plus court, en vert les tokens récupérés sur le chemin.

Une fois ce chemin déterminé, nous conservons les nœuds correspondant aux valeurs ainsi que ceux présents sur le chemin. Les chemins extraits correspondent aux sous arbres explicitant la relation entre les valeurs de départ, soit un ensemble de nœuds avec leurs traits associés. Une sélection des arbres s'opère ici puisque tous ne sont pas pertinents. En effet, dans certains cas il n'y a pas de chemin entre les valeurs parce qu'il n'y a pas de relation explicite dans la phrase (par ex. : "*26 février 1802 : Victor Hugo…*").

---

2 La visualisation des arbres syntaxiques a été réalisée *via* Arborator https://arborator.ilpga.fr

3 L'orientation dans les arbres syntaxiques sert à formaliser la relation hiérarchique entre les différents mots de la phrase. Seuls les liens sont utilisés ici, en faisant abstraction de la hiérarchie.

3.1.5 Définition des patrons lexico-syntaxique

Les patrons d'extraction sont enfin créés à partir des sous arbres sélectionnés et reliant les valeurs. Nous définissons un patron comme un objet composé de trois éléments :

1. Une représentation lexicale (sous phrase anonymisée),
2. Un ensemble de traits syntaxiques (patron syntaxique),
3. Une liste de catégories d'entités nommées attendues par le patron.

La création du patron se fait en deux étapes. La première, l'anonymisation, a pour objectif d'identifier les entités nommées dans la sous-phrase et de construire la partie lexicale du patron. La seconde est la sélection des attributs du sous arbre qui permettront de composer le patron syntaxique.

3.1.6 Construction du patron lexical

Nous analysons un tronçon à anonymiser à l'aide de l'extracteur d'entités nommées construit à base de règles afin de remplacer les entités trouvées par les caractères ".*", ce qui permet la réutilisation du patron pour l'extraction de nouvelles valeurs. Cette chaîne anonymisée correspond à la représentation lexicale du patron tandis que la liste des types d'entités nommées détectées est également conservée comme attribut du patron lexico-syntaxique. Pour l'extraction des entités nommées nous utilisons un système développé en interne à Syllabs (Ma et al., 2011). Les différents types d'entités concernent les noms de personne (*Person*), les noms de lieu (*Geo*), les dates (*Date*), les noms de profession ou de fonction (*Function*) et les noms d'organisation (*Organisation*).

Les sous-phrases sont construites à partir de tous les tokens sur le chemin reliant les valeurs. Parmi ces tokens, d'éventuelles entités nommées qui n'étaient pas présentes dans les valeurs de départ sont extraites. Par exemple dans la sous phrase "*Victor Hugo est né à Besançon le 26 février 1802*", nous identifions avec l'extraction d'entités nommées une nouvelle valeur, à savoir *Besançon*, le lieu de naissance de *Victor Hugo*. Le patron construit permettra d'extraire 3 valeurs : un nom, une date de naissance et un lieu de naissance.

3.1.7 Construction du patron syntaxique

La construction du patron syntaxique se fait à partir de certains attributs obtenus lors de l'analyse syntaxique. Ces attributs sont obtenus à partir des entités anonymisées lors de l'étape précédente. Pour chaque entité, les traits retenus dans le patron syntaxique sont la partie du discours, la forme lexicale du token du nœud gouverneur, et les éventuels enfants : si la partie du discours d'un nœud donné est un nom commun, un nom propre, une préposition simple ou composée, on ne pourra pas déterminer, lors de l'extraction, ni le nombre ni la nature des nœuds gouvernés par le nœud donné, ce trait est donc catégorisé en optionnel ('*optional*'). Pour toute autre partie du discours, le lien de dépendance et la position des enfants dans la phrase sont retenus pour ce trait.

```
{0: {u'children': 'optional',
     'gov': {u'né': u'suj'},
     'tag': u'NPP'},
1: {u'children': 'optional',
    'gov': {u'né': u'mod'},
    'tag': u'NC'}}
```

FIGURE 4 : Exemple de patron syntaxique correspondant au patron lexical ".*, *né le* .*"
Chaque entrée du patron correspond à une entité à extraire et ses attributs attendus

3.2 Utilisation des patrons lexico-syntaxiques pour extraire de nouvelles valeurs

Dans la section précédente nous avons présenté notre méthode de construction de patrons lexico-syntaxiques du web à partir de valeurs prédéfinies. Ces patrons vont nous permettre d'extraire de nouvelles valeurs comparables à celles de départ. Les deux méthodes sont, au final, assez proches dans leur fonctionnement. Nous utilisons les patrons lexicaux comme requêtes pour le moteur de recherche afin d'extraire des documents web et comme expressions régulières pour sélectionner les phrases pertinentes (1). Ces phrases sont analysées syntaxiquement et le patron syntaxique est utilisé pour en extraire de nouvelles valeurs (2). Ces valeurs sont enfin validées au cours de la phase d'identification, à l'aide d'un extracteur d'entités nommées (3).

3.2.1 Utilisation des patrons lexicaux pour la sélection de phrases

Afin de rassembler des documents pertinents pour l'extraction de nouvelles valeurs, nous utilisons dans un premier temps les représentations lexicales des patrons comme requêtes passées au moteur de recherche. En effet, la présence de caractères ".*" à la place des valeurs est considérée comme un joker par les moteurs de recherche[4]. Ainsi, un patron représenté par *".* est né le .* à .*"* permettra de retrouver des documents contenant des phrases correspondant au patron. Par ailleurs les caractères ".*" sont également des jokers dans la syntaxe des expressions régulières, que nous utilisons dans l'étape suivante. Comme dans la section précédente, chaque requête renvoie jusqu'à 100 URLs. Nous utilisons également le même algorithme d'extraction du texte utile des documents web retournés. Le patron lexical est appliqué aux contenus collectés en tant qu'expression régulière. Les segments détectés par le patron sont conservés puis analysés.

3.2.2 Extraction de nouvelles valeurs à l'aide des patrons syntaxiques

Les segments sélectionnés au cours de l'étape précédente sont analysés syntaxiquement avec Talismane afin d'obtenir les mêmes traits syntaxiques que ceux de la section précédente. Les patrons syntaxiques sont appliqués aux arbres issus de l'analyse. Nœud après nœud, les traits du patron sont comparés de manière itérative à ceux de l'arbre en commençant par la partie du discours, suivie du nœud gouverneur. Si un nœud de l'arbre possède les mêmes traits qu'un nœud du patron, le nœud et ses éventuels enfants sont conservés. Par exemple le patron suivant *".* est né à .* le .*"*, attend pour la première valeur à extraire les traits `{pos : NPP, gov : (né, suj)}`. Ainsi, la phrase "*Thomas est né à Rouen (France), le 27 février 1978.*" (dont l'analyse syntaxique est présentée en Figure 5) possède bien un nom propre (NPP) sujet du verbe *naître* (suj). La première valeur est ainsi extraite.

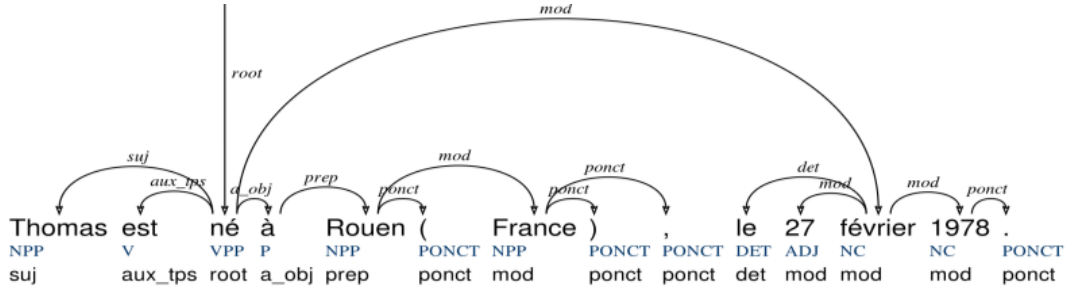

FIGURE 5 : Exemple de phrase sélectionnée après analyse syntaxique

[4] Exemple de syntaxe Google : https://support.google.com/websearch/answer/2466433

Nous procédons ainsi jusqu'à ce que toutes les valeurs du patron soient trouvées. Si un segment ne contient pas le nombre de valeurs attendu par le patron, elle n'est pas sélectionnée. Au terme de cette extraction nous conservons l'ensemble des segments sélectionnés et leurs valeurs extraites.

```
    "values": { "<Date>": "27 février 1978",
                "<Geo>": "Rouen France",
                "<Person>": "Thomas" },
    "sentence": "Thomas est né à Rouen ( France ) , le 27 février 1978 ."
```

FIGURE 6 : Exemple de valeurs extraites

3.2.3 Identification des entités nommées

Les entités nommées sont ensuite analysées à partir des segments sélectionnés lors de l'extraction des valeurs afin de valider que les valeurs extraites du segment correspondent bien au type d'entités attendues par le patron. L'extracteur d'entités nommées de Syllabs permet le typage des entités.

```
"values": { "<Date>": "27 février 1978",
            "<Geo>": "Rouen France",
            "<Person>": "Thomas" },
"sentence": "Thomas est né à Rouen ( France ) , le 27 février 1978 .",
"entities": [ {'name': 'Geo', 'text': u'France'},
              {'name': 'Person', 'text': u'Thomas'},
              {'name': 'Geo', 'text': u'Rouen'},
              {'name': 'Date', 'text': u'27 février 1978'}]
```

FIGURE 7 : Exemple de résultats de l'identification des entités nommées

Comme nous pouvons le voir dans la Figure 7, le type des valeurs extraites par le patron correspond ici à celui attendu. Dans le cas où nous aurions plusieurs valeurs possibles parmi les résultats (par ex. un nom et deux dates de naissances possibles), nous prenons celle qui a le plus d'occurrences.

4 Exemple d'extraction simple

À l'aide des patrons lexico-syntaxiques construits lors de la première phase de notre méthode, nous avons réussi à extraire de nouvelles valeurs similaires et complémentaires à celles de départ. Avec les deux valeurs présentées en exemple en entrée, nous avons extrait 529 ensembles de valeurs (uniques) dont 322 qui contenaient de nouvelles entités (c.-à-d. lieu de naissance et/ou profession). Nous détaillons dans cette section les résultats des deux phases de constructions des patrons et d'extraction de nouvelles valeurs, à partir de l'exemple *"Victor Hugo+26 février 1802"*.

4.1 Construction de patrons

Au terme de la première phase de collecte de documents sur le web, nous avons obtenu 91 documents (certaines des 100 réponses étant des liens morts). Après la phase de sélection des phrases contenant les valeurs, 47 d'entre elles ont été conservées, puis 47 arbres syntaxiques suite à l'analyse syntaxique, desquels les relations entre les valeurs de départ sont extraites. Une fois la sélection des relations terminée, nous avons obtenu 34 sous arbres syntaxiques soit, au terme de la phase de création des patrons, 34 patrons lexico syntaxiques permettant l'extraction de noms de personnes et leur date de naissance et, dans la majorité des cas (26 patrons), de nouvelles informations, à savoir le lieu de naissance et la profession.

4.2 Extraction de nouvelles valeurs

Les 34 patrons extraits sont appelés comme requêtes dans un moteur de recherche, nous donnant 3 400 documents, soit près de 25 000 segments à analyser. Nous détaillons les résultats pour un patron particulier *".* est né à .* le .*"* qui nous a permis d'extraire 19 ensembles de valeurs après analyse syntaxique et identification des entités nommées. La Figure 8 présente quelques valeurs extraites par ce patron. Au total, avec les 34 patrons nous avons extrait 1 380 valeurs. Parmi les 1 380 valeurs extraites, 653 sont validées par l'identification des entités nommées, soit 47 %.

```
"values": { "Person": "Jacques De Decker",
            "Geo": "Bruxelles",
            "Date": "le 19 août 1945" },
"sentence": "Jacques De Decker est né à Bruxelles le 19 août 1945 ."

"values": { "Person": "Yann Arthus-Bertrand",
            "Geo": "Paris",
            "Date": "le 13 mars 1946" },
"sentence": "Yann Arthus-Bertrand est né à Paris le 13 mars 1946 ."

"values": { "Person": "Thomas",
            "Geo": "Rouen France",
            "Date": "le 27 février 1978" },
"sentence": "Thomas est né à Rouen ( France ) , le 27 février 1978 ."
```

FIGURE 8 : exemples de valeurs trouvées par le patron "*.* est né à .* le .**"

D'autres exemples de patrons trouvés sont présentés dans le Tableau 1.

né le <Date> à <Geo> dans le <Geo> , <Person>
<Person> est né le <Date>
<Person> est né à <Geo> le <Date> ,
<Person> naît le <Date> à
la naissance de <Person> : <Person> est né le <Date>
Troisième et dernier enfant de <Person> et <Person> , <Person> naît à <Geo> le <Date> .
Éphéméride <Date> naissance de <Person> <Date>_11:05 Classé

TABLE 1 : Exemples de patrons construits à partir des valeurs "Victor Hugo" et "*26 février 1802*"

5 Evaluation de la méthode

Dans cette section, nous mesurons la capacité de notre méthode à extraire des informations pertinentes à partir de la requête "*Victor Hugo+26 février 1802*", à savoir des dates de naissance de personnalités sélectionnées dans un jeu d'évaluation issu de la base de connaissances DBpedia[5]. Pour ce faire, nous avons sélectionné 100 noms de personnalités françaises au hasard ainsi que leurs dates de naissance servant, elles, de référentiel.

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

5.1 Protocole

Les noms des personnalités sont utilisées comme amorce et sont susceptibles, en les associant aux patrons d'extraction, de permettre l'identification des dates de naissance à partir du web. Pour chaque patron, la valeur correspondant à l'entité de type nom est donc remplacée par le nom d'une personnalité. L'objectif est ici l'enrichissement d'une base de connaissances.

Les patrons d'extractions augmentés des noms de personnalités sont passés comme requête au moteur de recherche. Les documents retournés sont collectés et les segments pertinents sont sélectionnés en appliquant les patrons lexicaux comme expression régulière (cf. section 2.1). Une fois les segments sélectionnés, ils sont analysés syntaxiquement et les patrons syntaxiques sont appliqués (cf. section 1.3). Enfin, les valeurs de type *Date* sont extraites, normalisées (c.-à-d. deux chiffres pour le jour, le mois en toutes lettres et les quatre chiffres de l'année), puis la date la plus fréquemment trouvée pour chaque nom est comparée à celle de DBpedia.

La précision (p), le rappel (r) et la f-mesure (f1) ont été utilisées et sont définies selon :

$$p = nb\ de\ valeurs\ correctes\ trouvées \div nb\ de\ valeurs\ trouvées$$
$$r = nb\ de\ valeurs\ correctes\ trouvées \div nb\ de\ valeurs\ à\ trouver$$
$$f1 = 2 * (p * r) / (p + r)$$

5.2 Résultats

5.2.1 Précision des patrons

Au terme de l'extraction, nous trouvons des valeurs pour 10 des 34 patrons d'extraction. Parmi ces 10 patrons, 4 d'entre eux n'ont trouvé aucune valeur correcte, soit une précision de 60 %. La précision pour l'ensemble des patrons est de 17 %. Le Tableau 2 présente le nombre des valeurs trouvées et correctes pour chacun des patrons présentés dans le Tableau 1.

Patron	Valeurs trouvées	Valeurs correctes
né le .* à .* dans le .* , .*	0	0
.* est né le .*	48	34
.* est né à .* le .* ,	1	0
.* naît le .* à	4	2
la naissance de .* : .* est né le .*	0	0
Troisième et dernier enfant de .* et .* , .* naît à .* le .* .	0	0
Éphéméride .* naissance de .* .*_11:05 Classé	0	0

TABLE 2 : Quelques exemples de patrons et le nombre de valeurs extraites par chacun d'entre eux

5.2.2 Evaluation de la méthode selon les valeurs identifiées

Concernant les scores d'évaluation par valeur, sur les 100 dates de naissance à trouver initialement, des valeurs ont été extraites pour 46 d'entre elles. Parmi ces 46 valeurs, et après normalisation, 34 correspondent aux dates indiquées dans DBpedia. La précision moyenne pour les 46 valeurs brutes extraites est de 74 % et le rappel sur l'ensemble des valeurs est de 34 %, ce qui donne une f-mesure de 47 %. Nous présentons ici deux résultats qui ne sont pas corrects et en expliquons les causes.

Le premier exemple est présenté en Figure 9. Dans cet exemple nous constatons que la date extraite n'est pas fausse, mais incomplète. En effet, nous recherchons des dates complètes (jour, mois et année), tandis qu'ici seule la date est spécifiée dans la phrase.

```
{ ".* est né le .*": [
    { "values": { "<Date>": "14 juillet",
                  "<Person>": "Marine ce joli bébé de 2,080 kg" },
      "sent": "Marine , ce joli bébé de 2,080 kg , est né le 14 juillet à 13 h
25 à la maternité de Chalon-sur-Saône ." } ] }
```

FIGURE 9: un exemple de résultat incomplet

Le second exemple, présenté en Figure 10, est un cas de mauvais découpage en phrases. En effet, l'expression régulière de ce patron extrayant tous les termes avant "Naissance", et la phrase étant mal découpée (probablement dû à l'absence de ponctuation entre les deux propositions "*Encyclopedia Universalis Repères biographiques*" et "*16 septembre 1867 Naissance de Jean-Baptiste Charcot à Neuilly-sur-Seine* ."), l'analyse syntaxique est faussée et tous les termes avant la date sont extraits.

```
{ ".* Naissance de .* à": [
    { "values": { "<Date>": "Encyclopedia Universalis Repères biographiques 16
septembre 1867",
                  "<Person>": "Jean-Baptiste Charcot" },
      "sent": "Source : Encyclopedia Universalis Repères biographiques 16
septembre 1867 Naissance de Jean-Baptiste Charcot à Neuilly-sur-Seine ." } ] }
```

FIGURE 10 : un exemple de résultat mal extrait

5.3 Conclusions de l'évaluation

Après observation des résultats, nous constatons que parmi les 34 patrons d'extraction initiaux, 24 d'entre eux ne trouvent aucune valeur, parce qu'ils contiennent trop de bruit (nous proposons un début de réflexion dans la section suivante concernant l'amélioration de la sélection de patrons.). Toutefois, jusqu'à présent, nous avons conservé l'ensemble des patrons afin d'être le plus exhaustif possible, obtenir un grand nombre de valeurs, et ainsi se fier à la valeur ayant le plus grand nombre d'occurrence. Notre méthode nous permet d'atteindre une précision de 74% pour l'extraction de dates seules, sachant que les erreurs viennent parfois de résultats incomplets. Cependant, nous devons revoir certaines phases de la méthode, notamment le découpage en phrases afin d'améliorer l'analyse syntaxique et par conséquent l'extraction de nouvelles valeurs.

6 Améliorations

Nous présentons dans cette section nos deux principales pistes d'amélioration de la méthode pour l'enrichissement de connaissances à partir du web.

6.1 Sélection des patrons avant la recherche de valeurs

Comme nous avons pu l'observer au cours de l'évaluation DBpedia, certains patrons ne permettent pas du tout d'extraire des valeurs. De plus, parmi les patrons construits, beaucoup sont très proches, avec parfois seulement un caractère de différence. Ainsi, nous avons cherché à sélectionner les patrons "utiles" avant l'extraction de nouvelles valeurs. La première question a été de déterminer quels seraient les critères de sélection d'un patron d'extraction. En effet, d'une extraction à l'autre, les patrons sont très différents, et notre méthode doit s'adapter à tous les cas de figure.

Nous avons envisagé d'utiliser la mesure de l'entropie de Shannon (2001) pour classer les patrons selon la quantité d'information. Le choix du calcul de l'entropie est motivé par la définition même de cette mesure, qui peut être calculée simplement à partir de chaînes de caractères, indépendamment de la langue ou du contexte.

On constate que plus le patron est petit, plus son score d'entropie est faible. Dans le cas des patrons de naissances que nous avons utilisé dans cet article, nous pouvons faire un lien entre le score de précision du patron et celui de l'entropie. Il est ainsi possible d'émettre l'hypothèse que plus le patron est simple, plus il est efficace. Cependant, si cette hypothèse se vérifie, nous devrons encore déterminer un seuil de sélection du patron, soit par rapport à un score défini, soit par rapport à une proportion de patrons à sélectionner. Cette question est encore à l'étude.

6.2 Extractions à distance

Nous avons défini par extraction à distance le fait d'essayer d'appliquer l'outil d'extraction non seulement au sein d'une phrase, mais aussi à de plus grandes distances, par exemple sur l'ensemble d'un document. Notre problème se rapprochant de la résolution de chaînes de coréférences, nous avons implémenté une méthode inspirée de (Hobbs, 1986) et (Lappin & Leass, 1994) pour retrouver l'antécédent d'un pronom sujet. L'idée principale est de retrouver le nom de l'entité à laquelle le pronom fait référence en se basant sur les informations morphosyntaxiques et l'arbre syntaxique des phrases précédentes.

Ainsi, pour une paire de phrases comme "Deborah Gibson naît le 31 août 1970 à Brooklyn. Elle est la troisième d'une famille de quatre enfants."[6], notre objectif est de lier les deux phrases et d'obtenir les informations comme quoi *Deborah Gibson est née le 31 août 1970 à Brooklyn*, et *Deborah Gibson est la troisième d'une famille de quatre enfants*.

Talismane possédant un niveau d'analyse morphologique, nous avons ajouté la méthode dans la chaîne de traitement. Ainsi, lorsque nous trouvons un pronom sujet, nous gardons les informations de genre et de nombre et tentons de retrouver la première entité dans les phrases précédentes qui partagent ces informations morphologiques. Cependant, Talismane n'identifie pas assez bien les noms, et notamment leur genre, et par conséquent ne les étiquette pas tous en genre et en nombre. Pour pouvoir utiliser cette solution de résolution de chaînes de coréférences, nous pensons réentrainer Talismane sur un corpus de noms annotés en genre et en nombre.

7 Conclusion et travaux futurs

Nous avons présenté une méthode de construction semi-supervisée de patrons de relation et d'extraction en français, à partir du web. Nous avons montré que notre méthode permettait d'extraire des valeurs similaires (nom, date) et complémentaires (lieu, fonction) à partir d'un unique jeu de valeurs en entrée. Cependant, notre méthode n'est pas complète, et comporte encore des faiblesses, notamment au niveau de la sélection des patrons pour l'extraction de nouvelles valeurs. Par ailleurs, l'utilisation d'arbres syntaxiques en dépendances et la disponibilité de ressources multilingues nous permettrait de porter la méthode dans d'autres langues que le français. Une de nos prochaines étapes de réflexion sera également de porter la méthode à des relations de plus de deux valeurs.

[6] Exemple issu de la page Wikipedia : https://fr.wikipedia.org/wiki/Deborah_Gibson

Références

AGICHTEIN E., GRAVANO L. (2000). Snowball: Extracting relations from large plain-text collections. In *Proceedings of the fifth ACM conference on Digital libraries* (pp. 85-94). ACM.

AKBIK A., VISENGERIYEVA L., HERGER P., HEMSEN H., LÖSER A. (2012). Unsupervised Discovery of Relations and Discriminative Extraction Patterns. In 24th *International Conference on Computational Linguistics*, COLING 2012 (pp. 17-32).

BRIN S. (1998). Extracting patterns and relations from the World Wide Web. In *International Workshop on the World Wide Web and Databases* (pp. 172-183). Springer, Berlin, Heidelberg.

CAFARELLA M., DOWNEY D., SODERLAND S., ETZIONI O. (2005). KnowItNow: Fast, scalable information extraction from the web. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing* (pp. 563-570). Association for Computational Linguistics.

ETZIONI O., CAFARELLA M., DOWNEY D., KOK S., POPESCU A., SHAKED T., SODERLAND S., WELD D., YATES A. (2004). Web-scale information extraction in knowitall. In *Proceedings of the 13th international conference on World Wide Web* (pp. 100-110). ACM.

ETZIONI O., FADER A., CHRISTENSEN J., SODERLAND S., MAUSAM M. (2011). Open Information Extraction: The Second Generation. In *IJCAI* (Vol. 11, pp. 3-10).

FINN A., KUSHMERICK N., SMYTH B. (2001). Fact or fiction: Content classification for digital libraries. *Joint DELOS-NSF Workshop on Personalisation and Recommender Systems in Digital Libraries*. Dublin.

GRISHMAN R., SUNDHEIM B. (1996). Message Understanding Conference - 6: A Brief History. In: *Proceedings of the 16th International Conference on Computational Linguistics* (COLING), I, Kopenhagen, 466–471.

KISS T., STRUNK J. (2006). Unsupervised Multilingual Sentence Boundary Detection. *Computational Linguistics 32*: 485-525

LIN D., PANTEL P. (2001). DIRT discovery of inference rules from text. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM.

MA J., MOUNIER M., BLANCAFORT H., COUTO J., DE LOUPY C. (2011) LOL: Langage objet dédié à la programmation linguistique. *In Proceedings of TALN*.

MORIN E. (1999). Extraction de liens sémantiques entre termes à partir de corpus de textes techniques. *Thèse de doctorat*. Nantes.

SHANNON C. E. (2001). A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1), 3-55.

URIELI A. (2013). Robust French syntax analysis: reconciling statistical methods and linguistic knowledge in the Talismane toolkit. *PhD thesis*. Université de Toulouse II-Le Mirail.