UNIVERSITÉ DE MONTRÉAL

IFT6285
TRAITEMENT AUTOMATIQUE DES LANGUES NATURELLES

Projet 2: Extraction d'information

par:

Ilyas Amaniss
(20148123)
Henri-Cedric Mputu Boleilanga
(20174905)
Eugénie Yockell
(20071932)

Date de remise: 23 décembre 2020

1 Introduction

Dans ce projet, on s'intéresse à l'extraction d'information. C'est une tâche d'intérêt en 2020, dans un monde où la quantité de données grandi de manière exponentielle. L'extraction d'information permet de gérer cette quantité de données pour retirer automatiquement les informations utiles. On se restreint ici aux articles du journal Le Devoir. Nous avons accès à 10 184 articles de différents sujets. L'objectif est d'extraire différents types d'informations. On s'intéresse à l'extraction d'abréviations, à l'identification d'entité nommées et à l'extraction d'information ouverte (hyponymie).

Les 10 184 articles ont été tirés du journal Le Devoir et ont été publié entre septembre 2018 et août 2010. En plus du contenu de l'article lui-même, quelques métadonnées tel que la date de publication, quelques mots clés décrivant le contenu de l'article et le lien vers l'article originale. L'un des avantages d'utiliser un tel corpus est majoritairement la clarté et propreté du texte contenu. En effet, lors du projet précédent, énormément de pré traitement devait être fait sur les données puisqu'elles venaient de messages de blogueurs et contenaient donc des erreurs d'orthographes, des *emojis* et énormément de ponctuations. Malgré que les corpus contenant de tels particularités sont quelques fois intéressant pour certaines tâches (tel la classification faite au projet 1) avoir accès à du contenu écrit majoritairement par des journalistes professionnels et ne contenant (presque) pas de fautes, aucuns bruits tel que *emojis*, facilite grandement la tâche à accomplir ici: l'extraction d'information.

2 Expériences

2.1 Acronymes

La recherche d'acronymes consiste à associer les acronymes avec leurs définitions. La définition est souvent contenue dans le voisinage de l'acronyme et est couramment entre parenthèse. Soit l'acronyme est entre parenthèse, ou bien la signification est entre parenthèse.

2.1.1 Méthodologie

Pour cette expérience, nous avons donc décidé de comparer et d'explorer les extractions d'abréviations faites par deux algorithmes différents: la librairie `SciSpacy` et l'algorithme Schwartz-Hearst (1). Il sera intéressant de comparer quels acronymes seront trouvé par l'un et non par l'autre et de déterminer si la combinaison potentielle de ces deux techniques serait avantageuse à ce projet.

La librairie `SciSpacy` est une extension de la bibliothèque `Spacy` mais utilisée particulièrement lors de traitement de langue naturel pour le domaine biomédicale, scientifique et clinique. Plus spécifiquement, le module AbreviationDetector de cette librairie sera utilisé afin d'extraire les abréviations de notre corpus en employant le modèle frCoreNewsLg qui est un réseau neuronal de convolution (CNN) entraîné sur des corpus de la langue française, intéressant donc pour notre corpus de nouvelles étant lui aussi français.

L'algorithme Schwartz-Hearst, originellement lui aussi développé afin d'être utilisé dans un contexte médical où leurs corpus sont souvent composés de plusieurs acronymes, est constitué de deux tâches principales. La première étape est l'extraction de paires <*forme courte*, *forme longue*> provenant du texte. Une de ces paires est ainsi extraite lorsqu'il existe une correspondance entre les caractères de la *forme courte* et celles de la *forme longue*. La seconde étape consiste ensuite à trouver la bonne *forme longue* parmi les candidats potentiels se trouvant autour de la *forme courte*. La méthodologie spécifique de cet algorithme est décrite dans le document (1).

Afin d'avoir une idée générale du nombre possible d'acronymes présent dans ce corpus et ainsi estimer la qualité et le nombre d'acronymes trouvé par chaque algorithme, la librairie de Regex sera utilisée. En effet, en cherchant certaines caractéristiques dans la phrase tel qu'un mot écrit de minimum deux majuscules, la plupart des acronymes pourront être retrouvés. Cependant, cette méthodologie afind d'évaluer le nombre d'abréviations trouvées comporte certaines limitations qui seront bien sur analysées ensuite.

2.1.2 Résultat

Utilisant la librairie de Regex, 3286 acronymes uniques ont supposément été trouvé. Parmi ceux-ci, certains acronymes semblait être correct, tel que: OMT (Organisation mondiale du tourisme), ANL (Armée nationale libyenne) et même l'EDP (Energias de Portugal). Cependant, parmi ces 3286 acronymes se trouvait aussi plusieurs éléments n'étant pas nécessairement des acronymes. En effet, on y retrouvait certains mots simplement écrit en majuscule tel que *REFUS*, *AUDACE*, *DISCULPATION*, des mots en anglais tel que *FANTASTIC*, *COCAINE*, *KEEP*, mais aussi certains termes très intéressant tel que des onomatopées comme *WOOOOOOO*, des films comme *C.R.A.Z.Y* et des noms d'artistes comme *KAYTRANADA*. Il est donc clair que parmi les 3286 termes considérés comme étant des acronymes par la Regex, une majorité n'est en fait que du bruit. Ce chiffre de 3286 acronymes ne sera donc utilisé ici que comme un possible plafond maximum du nombre d'acronymes présents dans ce corpus.

L'algorithme de Schwartz-Hearst a été en mesure d'extraire 1399 abréviations. D'abord, ce nombre correspond à un peu plus de 42.5% du plafond suggéré à l'aide de la Regex. Cependant, après un parcours manuel des résultats obtenus par cet algorithme, la qualité des résultats semble très prometteuse. En effet, la qualité de ces résultats est similaire à celle décrite dans (1) où une précision de plus de 95% était obtenus sur deux corpus différents. Voici quelques exemples des abréviations trouvés: DAM (Diversité artistique Montréal), 'AI' (Amnesty International), OPCQ (Ordre professionnel des criminologues du Québec), CIAM (Corporation internationale d'avitaillement de Montréal) et même certaines abréviations anglaises tel que WCS (Western Canadian Select) et CSMI

(Capital Sports Management Inc). Cependant, cet algorithme semble aussi contenir quelques lacunes qui sont présentées au tableau 1. On remarque que les acronymes problématiques sont hors normes. Effectivement, un acronyme classique est habituellement composé d'au moins deux lettres majuscules. Ici, l'algorithme est confus par ces formes contenant des chiffres mais aussi des lettres minuscules.

Acronymes	**Signification**
10e	0 à 20 ans seulement lundi
anti-EI	ayant visé des ressortissants des pays de la coalition
TET	territoire
re	rapportés comme tel. À

Table 1: Erreurs d'acronymes extraits.

Le CNN de `SciSpacy` à été en mesure d'extraire 1555 abréviations du corpus de ce projet, ce qui correspond à 47% des 3286 abréviations suggérés par la Regex et à 156 abréviations de plus que l'algorithme Schwartz-Hearst. Il reste par contre encore à déterminer si ces abréviations supplémentaires sont en effet corrects et légitimes, mais aussi à déterminer quelles sont les abréviations similaires et différentes entre l'algorithme Schwartz-Hearst et le modèle de `Spacy`. Cependant, à première vue, de façon similaire au premier algorithme, la très grande majorité des extractions faites avec `SciSpacy` semblent être de très bonne qualité. En effet, on y retrouve les acronymes IFDD (Institut de la Francophonie pour le développement durable), SFPQ (Syndicat de la fonction publique et parapublique), Cecobois (Centre d'expertise sur la construction en bois). On y retrouve aussi des acronymes anglais tel que NES (Nintendo Entertainment System), CRU (Commodities Research Unit) et GGB (Green Growth Brands). Cependant, ce modèle n'est pas parfait. Il contient aussi des extractions fautives, mais aussi du bruit. En effet, il a considéré le mot "troisième" comme étant un acronyme de "troisième" et il semble aussi quelques fois extraire du bruit tel qu'associer le mot "pays" à l'acronyme "pays observateur". Le détail des extractions est présent dans le code donné. Il est aussi très important de noter que cet algorithme a pris près de 333 fois plus de temps pour extraire ces acronymes que son compétiteur.

De plus, en comparant les résultats du CNN et de l'algorithme de Schwartz-Hearst, d'intéressantes trouvailles ont été faite. En effet, ces deux algorithmes avaient en commun 1303 abréviations. Aussi, `SciSpacy` a trouvé 252 abréviations qui n'ont pas été trouvé par l'algorithme de Schwartz-Hearst tel qu'un acronyme lui-même composé d'un acronyme: l'ISU (Institut de statistique de l'UNESCO) et aussi de plus simples acronymes étrangement non découvert par Schwartz-Hearst tel que ESSIMU (Enquête indépendante sexualité, sécurité et interactions en milieu universitaire). Pour sa part, Schwartz-Hearst a été en mesure de trouver 96 abréviations qui n'ont pas été trouvé par le CNN tel que We-Fi (Women Entrepreneurs Finance Initiative) et CISSS-AT (Centre intégré de santé et de services sociaux de l'Abitibi-Témiscamingue). Cependant, une grande majorité des différences entre les algorithmes semblent être fautives. Donc, nous pouvons considérer les 1303 abréviations communes entre ces deux techniques comme ayant une plus grand probabilité d'être de bonne qualité et les autres comme ayant une probabilité moindre d'être correct puisque l'acronyme n'aura été trouvé que par l'un des deux algorithmes.

2.1.3 Analyse

Pour ce qui est des résultats obtenus à l'aide de la Regex, il est important d'aborder plusieurs points. Puisque la tâche de ce devoir est non-supervisée, il était nécessaire de déterminer un moyen d'évaluer la qualité des algorithmes utilisés pour ce projet. Dans le meilleur des cas, une liste de toutes les abréviations présentes, mais aussi définis dans ce corpus aurait été donné en plus du corpus afin d'estimer la qualité des extracteurs. Cependant, n'ayant pas cette liste, une Regex a été utilisé afin d'avoir un estimé du nombre d'acronymes dans le corpus. L'un des inconvénients, par contre, de cette technique est d'abord le bruit extrait, comme présenté dans la section précédente, mais aussi le fait qu'aucune garantie n'est donné quant à la présence de la définition de l'acronyme dans l'article. En effet, l'auteur d'un article pourrait très bien employer un acronyme sans nécessairement le définir ou bien l'acronyme pourrait n'être défini que dans le titre par exemple. Un tel acronyme sera extrait par la Regex, mais il serait difficile (mais peut-être tout de même possible) de s'attendre à ce que l'un des algorithmes puisse extraire la signification de cet acronyme sans que sa définition ne soit présente dans le corpus, grâce par exemple à son apprentissage du corpus d'entraînement dans le cas du CNN. Donc, considérant cet aspect et le bruit présent provenant des extractions de la Regex, le nombre de 3286 acronymes devant être extraites de ce corpus par les algorithmes devrait être grandement revu à la baisse.

Ensuite, pour ce qui est des extractions faite par ces deux algorithmes, il était très intéressant de voir que la longueur d'un acronyme ne semble pas avoir d'impact sur la qualité de l'extraction. Effectivement, comme présenté précédemment, certaines abréviations ayant quelques fois plus de dix mots ont parfaitement été extraite par ces deux algorithmes tel que RNCREQ (Regroupement national des Conseils régionaux de l'environnement du Québec). Cette facilité semble provenir du fait qu'une correspondance presque parfaite (même en présence de certains adjectifs tel que "des", "de", "l'" et "du") entre les mots de l'abréviation et la première lettre de chacun des mots suivants est présente.

Un autre point intéressant est l'énorme différence de temps d'analyse entre les deux techniques utilisés. En effet, le CNN a pris près de 25 minutes afin d'analyser le corpus d'entraînement complet, comparé aux 4.5 secondes de son compétiteur. Cette différence vient probablement du fait que `Spacy` emploi un réseau neuronal, qui sont malheureusement majoritairement connu pour leur long temps d'entraînement mais aussi d'inférence. C'est ce qui justifie, entre autre, la recherche de meilleurs méthodologies d'entraînement de ces modèles (2), (3), (4) et aussi la limitation de l'application d'énormes réseau neuronaux en industrie.

Finalement, à la vue des résultats obtenus, il y aurait deux opportunités présentes pour l'obtention d'un extracteur d'acronymes plus complet pour ce projet. La première option serait de combiner les résultats de ces deux algorithmes afin d'obtenir un grand total de 1651 acronymes uniques pour ce corpus. Cependant, en augmentant le nombre d'acronymes total extraits, une augmentation du bruit et des extractions fautives sera aussi faites puisque le bruit émis par les résultats de chacun de ces deux modèles sera combiné. Donc, l'autre solution afin d'obtenir moins, mais par contre une meilleure qualité dans les extractions faites, sera de ne prendre que les acronymes communément trouvé par les deux modèles. Ces deux modèles étant très différent de nature (l'un étant un réseau neuronal et l'autre un algorithme sans apprentissage) la combinaison semble être la solution donnant la meilleure qualité de résultat. C'est donc cette méthodologie qui sera appliqué pour l'extraction d'abréviations du corpus test de ce projet.

2.2 Entités nommées

La recherche d'entités nommées est une branche importante dans le traitement automatique du langage naturel (TALN). Une entité nommée est une expression linguistique faisant référence au nom d'une personne physique, au nom d'une organisation, au nom d'un lieu, etc.

2.2.1 Méthodologie

Les entités nommées étant un élément devenu inéluctable dans la plupart de processus de TALN, il existe de nos jours plusieurs bibliothèques logicielles qui fournissent des fonctions qui permettent de faire la recherche d'entités nommées dans un texte.

Pour cette expérience, nous avons opté d'utiliser la librairie `Spacy`. Pour que celle-ci puisse fonctionner, il faut lui fournir un modèle statistique pré-entraîné. Nous lui avons donné le modèle suivant: $fr\_core\_news\_sm$. Ce modèle est fait pour le traitement automatique des textes en français.

Nous avons utilisé la fonction $ents$ de `Spacy` afin de trouver les entités nommées qui sont mentionnées dans chacun des articles de notre corpus. Cette fonction a déjà une liste de types d'entité particuliers qu'elle recherche. Les personnes PER, les noms d'organisations ORG, les locations LOC et les entités diverses MISC sont les types d'entités qui peuvent être produits en utilisant `Spacy`. Une fois que nous avons reçu toutes les entités nommées produites, il fallait trouver un moyen d'évaluer cela. Il est spécifié que `Spacy` avec les modèles que nous avons utilisés à un F-score autour de 83.42% en moyenne pour le plus grand modèle pré-entraîné. (5)

Mais ce score peut fortement changer en fonction du corpus dont on veut extraire les entités nommées. Ne connaissant aucun $benchmark$ pour notre corpus composé d'articles, nous avons choisi de sélectionner 100 articles du corpus et d'en faire une analyse manuelle. Nous avons trouvé les entités nommées et nous leur avons donné un type spécifique (PER, LOC, MISC ou ORG). Cette analyse est bien évidemment non objective car nous ne sommes pas des linguistes professionnels et le nombre d'articles considéré est bien faible par rapport au nombre total d'articles.

Il y a une grande quantité d'erreurs potentielles dans la recherche d'entités nommées. (6) Notre $benchmark$ nous a donc permis non seulement d'évaluer notre corpus mais aussi de détecter des erreurs dans les productions de `Spacy` que nous avons identifié puis que nous avons tenté de corriger. Nos recherches afin de corriger les productions de `Spacy` nous ont conduit vers les articles suivants (7) (8). Nous avons retenu de ces articles les approches suivantes pour améliorer nos résultats: 1. Utiliser un meilleur modèle statistique ; 2. Corriger les productions a l'aide du contexte de la phrase d'où elles ont été extraite.

Pour la première approche, nous avons simplement utilisé un modèle plus grand. Nous sommes passé de $fr\_core\_news\_sm$ à $fr\_core\_news\_md$ puis $fr\_core\_news\_lg$. Pour la deuxième approche, partant du modèle le plus grand, nous avons utilisé le $tagger$ de `Spacy` afin de rajouter du contexte. Nous avons utilisé la fonction 'pos_' de `Spacy` sur les mots de l'entité nommée et également sur les mots autour de ce qui a été identifié comme étant une entité nommée par `Spacy`. Nous avons regardé jusqu'à deux mots avant et après l'entité nommée. En fonction du type de l'entité, et du tag des mots du contexte, nous avons corrigé les productions de `Spacy`. La correction consiste tout d'abord à retirer les doublons dans les productions et enlever les productions qui répondent à certains critères: on a retiré toutes les productions de ce style ['Si', 'MISC'], c'est-à-dire les entités du type MISC dont l'entité à le tag "conjonction - sconj. Ensuite, on ajoute des mots dans les entités nommées : par exemple, la production ['Domi', 'PER'], deviendra ['Max Domi', 'PER'], car on identifie le mot d'avant Max comme étant un 'PROPN'. Pour une entité nommée du type PER, si le mot avant a le tag PROPN, on rajoute ce nom a l'entité nommée. Nous avons ainsi créé plusieurs règles de correction.

Pour terminer, nous avons évalué notre approche qui corrige les productions en fonction du contexte. Les productions finales qui ont été corrigé et les $tags$ qui ont servie à la correction ont été faite avec l'utilisation du modèle statistique pré-entraîné le plus grand soit $fr\_core\_news\_lg$.

2.2.2 Résultat

Notre évaluation des entités nommées produites a été faite en fonction des 100 articles du corpus que nous avons analysés manuellement comme mentionne ci-haut.

Notre mesure d'évaluation est le F1-score. Cette métrique est la moyenne harmonique de la précision et du rappel où la précision est le nombre de résultats positifs correctement identifié divisé par le nombre de tous les résultats positifs, y compris ceux qui ne sont pas correctement identifiés, tandis que le rappel est le nombre de résultats positifs correctement identifié divisé par le nombre de tous les échantillons qui auraient dû être identifiés comme positifs. Nous avons commencé par calculer le F1-score en fonction de la taille du modèle pré-entraîné dont on peut voir les résultats au tableau 2.

	$fr\_core\_news\_sm$ (petit)	$fr\_core\_news\_md$ (moyen)	$fr\_core\_news\_lg$ (grand)
F1-score	81,3%	81,9%	82,7

Table 2: F1-score sur les productions de `Spacy` en fonction de la taille du modèle statistique. Aucune correction est appliquée.

Nous avons ensuite utilisé la même métrique pour les entités nommées reçues après correction. On peut voir le score obtenu avant et après les corrections dans le tableau 3. Le premier score a était obtenue avec l'utilisation du modèle le plus petit tandis que le score deuxième score (le score après correction) a été obtenue avec l'utilisation du modèle le plus grand.

F1 sans correction + petit modèle	F1 avec correction + grand modèle
81,3%	86,4%

Table 3: F1-score sur les productions de `Spacy` avant et après correction

2.2.3 Analyse

Comme on peut le voir dans la table 2, nos premières productions retourne un score légèrement en dessous de la moyenne qui est de 83.42% pour le modèle statistique que nous avons utilisé. L'utilisation d'un modèle de plus grande taille à un impact important sur l'amélioration des résultats. Cette augmentation reste tout de même inférieure à ce que nous espérions, soit un score très proche de 85% pour le plus grand modèle. Pour ces dernières productions, nous les avons analysé et nous avons identifié des types d'erreurs que nous avons essayé de corriger.

Original		**Correction**
1. ['Si', 'MISC']	→	None
2. ['Domie', 'PER']	→	['Max Domie', 'PER']
3. ['Montréal', 'LOC'] ['Nord', 'LOC']	→	['Montréal Nord', 'LOC']
4. ['**', 'MISC']	→	None
5. ['Centre', 'LOC']	→	['Centre - Begin', 'LOC']
6. ['Canadienne', 'LOC']	→	Non corrigé

Les deux premier types d'erreurs ont été discuté plus haut dans la méthodologie. Le type 3 est équivalent au type 2. Le type 4, consiste aux signes de ponctuations qui ont été classifié comme étant de la classe MISC. Pour ce type d'erreur, la production est simplement supprimée. Les mots séparés par un signe de ponctuation comme un trait d'union constituent le type 5. Ce type d'erreurs peut être corrigé avec l'utilisation du tag qui précise si les mots du contextes sont un noms composés. Quand c'est le cas, le mot unique est remplacer par ce groupe de mots. Le dernier type d'erreur que nous avons identifié n'a pas pu être résolus. Pour ce type, les productions produites ne sont pas bonnes. Dans le contexte de la phrase où 'Canadienne' a été obtenu, ce mot ne fait pas référence à une localisation. Nous n'avons pas corrigé ce genre d'erreur, car c'est une tâche complexe qui nécessite une plus grande analyse.

La correction qui consistait à rajouter du contexte nous a permis d'obtenir un score proche de ce que nous espérions (autour de 85%). On voit qu'une correction des entités produites est nécessaire si nous voulons améliorer notre score. De tels corrections nécessite des analyses de productions et des textes afin de détecter les types d'erreurs et de tenter de les corriger. Une bonne correction consiste notamment à avoir un modèle statistique plus grand et à prendre en compte le contexte de chaque entité.

2.3 Extraction d'information ouverte (hyponymie)

L'extraction d'information ouverte consiste à extraire de l'information structuré à partir de documents souvent non structuré. L'information structuré consiste généralement de deux arguments relié par une relation sémantique. On s'intéresse ici uniquement au extraction de relation taxonomiques, soit de deux arguments relié par le verbe tre en relation d'hyponymie. L'extraction d'information ouverte permet de créer une base de connaissance pour obtenir une meilleur recherche d'information, pour produire un outil de fouille ou pour simplement obtenir une meilleur compréhension d'un texte.

2.3.1 Méthodologie

La tâche d'extraction d'information ouverte est très difficile dans notre cas. Les articles de journaux sont composés de phrases complexes qui ne permettent pas l'utilisation de patrons précis. On se retrouve aussi avec de très longues phrases contenant plusieurs $nsubj$, parce qu'il y a plusieurs verbes. En français, pour une phrase, les *syntactic dependency parsing* (gauche de la flèche) et les *part of speech tagging* (droite de la flèche) sont,

Louis	nsubj → PROPN	un	det → DET
Celestin	flat:name → PROPN	musicien	ROOT → NOUN
est	cop → AUX	électronique	amod → ADJ

Le premier extracteur développer est un extracteur par patrons. L'extracteur par patron se base sur des patrons pour faire l'extraction en utilisant `Matcher` de `Spacy`. Ce `Matcher` va permettre d'extraire des séquences de texte qui correspond parfaitement au patron définie. Pour créer cet extracteur, nous avons commencé par un extracteur très simple, puis en testant sur différentes phrases d'articles on améliore l'extracteur tout en le conservant plutôt simple.

Initialement, nous utilisions qu'un seul patron à la fois pour identifier le gouvernant de la copule *est* ainsi que le groupe qui lui est associée. Ce choix avait pour effet de limiter le type d'extraction que nous pouvions faire. En effet, lorsqu'il y avait plusieurs

mots non important pour notre extraction entre le sujet nominal et la copule *est*, notre extracteur devenait obsolète et ne pouvait pas produire l'extraction souhaitée. Nous avons donc pris la décision de configurer le `Matcher` de manière indépendante pour chaque groupe. Un patron pour le sujet et un patron pour le groupe verbal. Un des patrons est le suivant:

Patron pour le groupe de $nsubj$: [det(?) **nsubj(+)** (NOUN (*) or PROPN (*)]

Patron pour le groupe verbal: [**ÊTRE** det(?) amod(*) ROOT(+) case(?) det(?) amod(*) NUM(*) PROPN(*) NOUN(*) NUM(*)]

Pour définir les patrons, il est nécessairement de prendre en compte que certain mots peuvent apparaître aucune ou plusieurs fois. On définit (*) un mot présent 0 ou plusieurs fois, (+) un mot présent une ou plusieurs fois, (?) un mot présent 0 ou 1 fois. De plus, nous avons un det (déterminant) avant le sujet nominal ($nsubj$), suivi d'un nom ou d'un nom propre. Tandis qu'à la suite du verbe tre, on peut avoir déterminant, un $case$ (préposition ou déterminant), un $amod$ (adjectif), et la racine ($ROOT$). On ajoute aussi une préposition à la suite de la racine qui peut être suivi d'une variété de noms ou nombre. Cet extracteur est nommé extracteur par patron.

Une deuxième méthode est utilisée. Elle consiste à regarder l'arbre des dépendances syntaxiques. On débute par la racine de l'arbre et on parcours les enfants. On recherche le groupe $nsubj$, soit le sujet du verbe, qui est un enfant de la racine. On parcourt ensuite les enfants du $nsubj$ afin d'ajouter quelconque adjectifs ou prépositions. On parcours aussi les enfants près de la racine pour ajouter les adjectifs ou autre. Afin d'obtenir des résultats pertinents, nous avons utiliser la condition que le sujet et la racine peuvent seulement être des noms. Ce choix peut paraître trop restrictif, mais ça peut être un choix judicieux lorsqu'on fait affaire à une grande quantité de pronoms. On retrouve souvent des phrases comme *Il est X* ou *C'est X*. Ce genre de phrase ne contiennent aucune information lorsqu'on ne connaît pas à quoi les pronoms font références. Il existe une manière de pallier à ce problème en observant les noms dans le voisinage. On peut ainsi remplacer le pronoms par le nom et obtenir une extraction contenant beaucoup plus d'information. (9) Une telle tâche est complexe à accomplir, car parfois même les informations locales ne sont pas assez pour s'assurer de l'origine du sujet. (10) Il est aussi intéressant de mentionner que nous avons ajouter une option pour prendre en compte le cas *est pas* en recherchant si les voisins du verbe être contiennent *pas*. Cet extracteur est nommé extracteur par dépendance.

2.3.2 Résultat

Les deux extracteurs retournent des résultats différents. Une comparaison est faite en appliquant l'extracteur sur 30 articles contenant environs 729 phrases. L'extracteur par patrons retourne 88 extractions tandis que le deuxième retourne 18 extractions. On mesure ici la qualité des extracteurs en mesurant que les extractions valides sont ceux qui contiennent de l'information intéressante sans aucune nécessité de contexte. Les deux extracteurs vont souvent retourné des extractions qui nécessite des connaissances de contexte comme,

L'annonce été saluée par des organismes	on est réjoui
C' est le Jeune Barreau de Montréal	L'incident est produit après 56 secondes

En comparant les deux extracteurs, on remarque l'extracteur par patrons retourne 9 extractions pertinentes tandis que le deuxième retourne 7 extractions pertinentes. Ils ont 5 extractions en commun. Ce qui est intéressant est que le extracteur par dépendance est beaucoup plus dense en information et contient beaucoup moins de bruits. Sur 30 articles, 38.8% était valide tandis que pour l'extracteur par patrons seulement 10.2% de ces extractions étaient valides.

En faisant une analyse manuelle, on remarque tout de même que le extracteur par dépendance ne prend pas en compte certaine extraction que l'extracteur par patron est capable d'obtenir. Comme on avait mentionné, on posait que la racine pouvait seulement être un nom. Cette condition permet de réduire le bruit, mais elle est trop restrictive. Par exemple, l'extracteur n'était pas en mesure de voir des extractions valides comme *Bernie Sanders est vieux*, car la racine est une adjectif. On ignorait aussi les extractions tel que *M. Vajna Evita est décédé*, car la racine est un verbe. Nous avons alors généralisé l'extracteur par dépendance pour qu'il puisse prendre en compte des racines de nom, adjectif ou verbe. Nous appelons cet extracteur, extracteur par dépendance modifié.

Sur les 30 mêmes articles testé plus haut, cet extracteur par dépendance modifié trouve 82 extractions. Parmi ces 82, il trouve 14 extractions pertinentes. Ce qui correspond à une densité de 17.1%. On juge que ça correspond à un bon compromis entre une grande densité d'information peu bruité de l'extracteur par dépendance de base et l'extracteur par patron. Afin de déterminer si cet extracteur par dépendance modifié est meilleur que l'extracteur par patron, nous allons pousser l'analyse à 60 articles, ce qui correspond à une totalité de 1465 phrases. Par 'meilleur' on sous entend un extracteur qui génère une bonne densité d'extractions indépendantes. Les résultats sont présentés au tableau 4.

	Extractions totales	Extractions pertinentes	Densité
Extracteur par patron	172	21	12.2 %
Extracteur par dépendance modifié	173	36	20.8 %

Table 4: Extractions effectués par les deux extracteurs sur 60 articles.

Les résultats du tableau 4 montre que le extracteur par dépendance modifié semble être meilleur que le extracteur par patron et génère une plus grande densité d'extractions pertinentes. Par contre, certains résultats démontrent que les extractions de l'extracteur par patron sont parfois de meilleur qualité. Ces exemples sont montrés au tableau 5. On remarque pour de mêmes extractions, l'extracteur par patron est parfois en mesure de relever plus d'information, car les patrons ont été créer pour prendre en compte de tel cas. Par contre, comme on voit au dernier exemple, ce n'est pas toujours le cas. En effet, le extracteur par dépendance est parfois en mesure de ressortir des informations très intéressante sur la société comme *90 % résidus Amérique une grande quantité déchets sont marques asiatiques* ou *Les dépenses animaux domestiques sont croissance au Québec*.

Extracteur par patron	Extracteur par dépendance modifié
L'économie américaine est dopée par une politique	L' économie américaine est dopée
Michael Jackson est mort le 25 juin 2009	Michael Jackson est mort
Les musées des Amériques sont nés au XIXe siècle	Les musées des Amériques sont nés
les petits sacs sont interdits depuis le	La France les petits sacs plastique sont interdits

Table 5: Comparaisons d'extractions effectués par les deux extracteurs sur 60 articles.

2.3.3 Analyse

On remarque que la création d'un extracteur par dépendance modifié était judicieuse. Celui-ci permet d'extraire une bonne densité d'informations pertinentes. L'extracteur par dépendance de base retournait une très haute densité d'informations peu bruité mais ces conditions étaient trop sévères et on perdait beaucoup d'informations. Cet extracteur est jugé inutile pour un cas où on cherche à retiré le plus d'information possible, même si on ajoute conséquemment du bruit.

On note que les extracteurs ont beaucoup de difficulté à pallier les problématiques de certaines subtilités de la langue française. Par exemple pour la phrase *L'Algérie est le théâtre depuis le 22 février de manifestations monstres* l'extracteur retourne *L'Algérie est théâtre*. Les extracteurs ne sont pas en mesure de repérer de tel métaphore (l'Algérie n'est pas un théâtre). Une façon gérer de tel pièges seraient d'avoir un corpus assez grand permettant d'associer une score de confiance sur les extractions en comptant le nombre de fois que les extractions sont présentes (un plus grande confiance sur des extractions fréquentes).

Un des problème avec l'extracteur par patron est qu'il peut être trop précis dans sa demande, car il faut qu'une séquence suive parfaitement ce patron. Pour garder une généralité, on créer un patron avec beaucoup de cas possibles. Le fait de créer un patron des patrons indépendant pour le sujet et pour le groupe verbal permet d'élargir notre rayon d'acceptation. Il est difficile de trouver une bonne balance entre créer des patrons trop généraux ou trop précis. Si on généralise trop les patrons on accepteras trop d'extractions inutiles. Par contre, cet extracteur a l'avantage de simplement se baser sur l'ordre des mots. En effet, l'extracteur par dépendance dépend des liens de dépendances de `Spacy`. Un résultat obtenue permet de bien comprendre cette problématique. On présente l'arbre de dépendance syntaxique d'une phrase à la figure 1 de l'annexe. On remarque que les liens de dépendance défini par `Spacy` ne sont pas valides. En effet, ici *L'* et *été* sont relié à la racine $roman$, alors que ce n'est pas le cas. Ce genre de problème induit l'extracteur en erreur qui extrait: *des quatre rois EST premier roman*, car en suivant l'arbre, *L'été* n'est pas relié au $nsubj$ *rois*. Donc, une des faiblesse du extracteur par dépendance est qu'il dépend des relations définies par `Spacy` qui sont parfois erronés. Par contre, le fait de dépendre de ces dépendance syntaxe peut parfois être un atout. Lorsque les dépendances définie par `Spacy` sont valides, il permet à l'extracteur de faire des liens que l'extracteur par patrons ne voit pas, parce que le patron n'accepte pas de tel forme dans ces patrons. C'est la raison pour laquelle le extracteur par dépendance retourne tout de même plus d'extractions pertinente.

Étant donné la problématique des pronoms qui n'est pas gérer, on juge que le extracteur par dépendance est meilleur que l'extracteur par patrons. En effet, il permet de retirer des extractions plus indépendantes du contexte, une plus grande quantité d'information et ainsi des extractions plus dense en information. Les deux extracteur retournent en général le même nombre d'extraction totale. Comme mentionné, on ignore toutes les extractions qui dépendent du contexte. Ces extractions pourrait tout de même parfois être utile si on voudrait créer différents type d'extractions. Par exemple, si on créer un extracteur qui considère différent verbe, autre que le verbe *être*, en combinant différentes extractions, on pourrait recréer différentes informations intéressante qui se compléteraient.

3 Conclusion

En conclusion, il y a une panoplie d'informations qu'on peut extraire d'un texte. Nous nous sommes particulièrement intéressé aux tâches de recherche d'acronymes, d'entités nommé et l'extraction d'information ouverte sur les relations d'hyponymie. L'extraction d'information est un domaine extrêmement vaste. Une étude plus approfondi nous fait comprendre que c'est une tâche très complexe. Il existe une énorme quantité de conditions à prendre en compte pour créer de bons extracteurs. Ici, nous nous sommes restreint à des problématiques simple afin de pouvoir les explorer le plus profondément possible.

La densité d'information extraite pour la tâche d'acronymes semble raisonnable. En effet, plus de 1300 acronymes en été extraits: soit un acronyme pour chaque dix articles. Étant donné la fréquence assez basse d'acronyme dans un article ou même dans un corpus (estimé ici à un maximum de 3286 acronymes), la qualité dans le nombre d'extractions faites, mais aussi de la qualité de chacune de ces extractions est en effet raisonnable. Par ailleurs, en plus de combiner le CNN et l'algorithme de Schwartz-Hearst afin d'obtenir de meilleurs résultats, une amélioration possible à notre système serait de tenter, basé sur le contexte de la phrase, de définir les acronymes utilisés, mais non défini dans le corpus.

Pour la recherche d'entités nommées, la densité d'information obtenu est proche de 95% (densité estimée a partir de nos 100 articles test). Mais il existe des erreurs dans la qualité des productions malgré nos corrections. Une amélioration intéressante serait d'explorer l'utilisation d'un modèle statistique plus grand qui devrait donner des meilleures productions de base puis elle permettrait d'avoir un meilleur correcteur. En effet, la correction dépend du $tagger$ qui lui aussi dépend ce modèle.

Sur le corpus total, pour l'extraction de relation d'hyponymie, étant donné qu'on utilise une analyse manuelle, on peut extrapoler le résultat obtenu (2.33% d'information pertinente sur 2 483 phrases, soit 100 articles) et poser qu'on serait en mesure d'extraire 2.33% extractions pertinente des articles totales, soit 6 440 extractions. Cette densité semble plutôt basse, mais c'est le résultat auquel on s'attend considérant qu'on s'intéresse à une très petite tâche. L'amélioration principale aurait été d'identifier la nature des pronoms. Une telle amélioration permettrait d'accepter un plus grand nombre d'extractions jugé inutile. De plus, il aurait été intéressant de s'aventurer vers différents type de relation pour l'extraction d'informations ouvertes.

Annexe

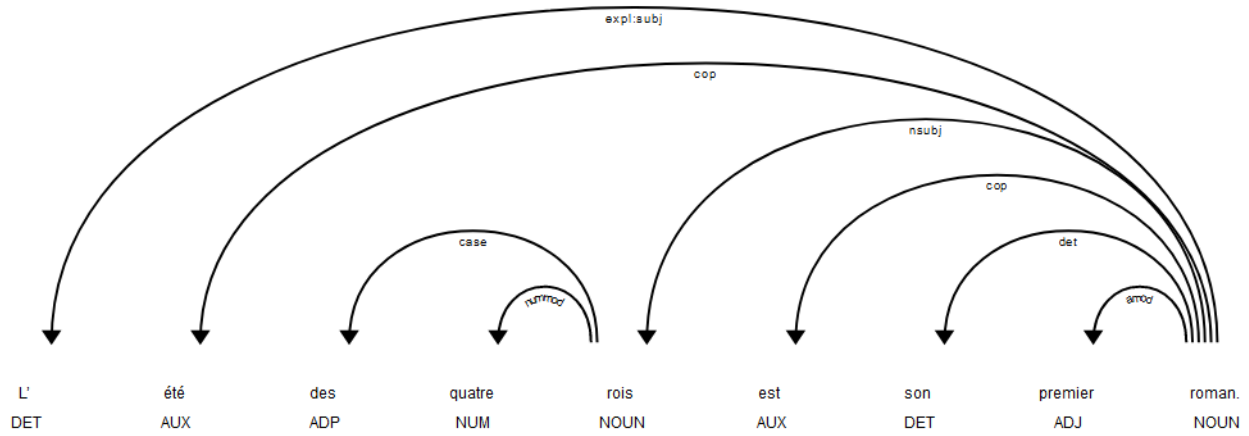

Figure 1: Dépendance syntaxique visualisé à l'aide de `Displacy`

References

[1] A. S. Schwartz and M. A. Hearst, "A simple algorithm for identifying abbreviation definitions in biomedical text," in *Biocomputing 2003*, pp. 451–462, World Scientific, 2002.

[2] R. P. Brent, "Fast training algorithms for multilayer neural nets," *IEEE Transactions on Neural Networks*, vol. 2, no. 3, pp. 346–354, 1991.

[3] D. Nguyen and B. Widrow, "Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights," in *1990 IJCNN International Joint Conference on Neural Networks*, pp. 21–26, IEEE, 1990.

[4] S. V. Kamarthi and S. Pittner, "Accelerating neural network training using weight extrapolations," *Neural networks*, vol. 12, no. 9, pp. 1285–1299, 1999.

[5] "French: Available pretrained statistical models for french." `https://spacy.io/models/fr`, 2016-2020. Accessed: 2020-12-19.

[6] M. Hatmi, "Reconnaissance des entités nommées dans des documents multimodaux," *Université de Nantes*, 2014.

[7] Y. Benajiba and P. Rosso, "ANERsys 2.0: Conquering the NER Task for the Arabic Language by Combining the Maximum Entropy with POS-tag Information," *Universidad Polit ecnica de Valencia*, 2007.

[8] L. S. A. Alireza Mansouri and A. Mamat, "Named Entity Recognition Approaches," *University Putra Malaysia*, 2008.

[9] L. D. Corro and R. Gemulla, "ClausIE: Clause-Based Open Information Extraction," *Max-Planck-Institute für Informatik*, 2005.

[10] S. Liao and R. Grishman, "Using Document Level Cross-Event Inference to Improve Event Extraction," *New York*, 2010.