Reconnaissance d'entités nommées itérative sur une structure en dépendances syntaxiques avec l'ontologie NERD

Cédric Lopez, Melissa Mekaoui, Kevin Aubry, Jean Bort, Philippe Garnier

Emvista
https://www.emvista.com
Cap Oméga, Rond-point Benjamin Franklin, 34960 Montpellier
prenom.nom@emvista.com

Résumé. La reconnaissance des entités nommées (REN) consiste à repérer des éléments textuels et à les classer dans des catégories prédéfinies (noms de personnes, d'organisations, de marques, d'équipes sportives, *etc.*). La REN est souvent considérée comme l'une des briques de fondation des systèmes visant à structurer un texte tout-venant. Dans cet article, nous décrivons notre système symbolique de REN qui se caractérise par 1) l'utilisation de ressources dictionnairiques limitées et 2) la prise en compte de résultats provenant d'autres briques telles que la résolution de coréférences et l'extraction de relations. Le système est basé sur la sortie d'un analyseur syntaxique en dépendances qui adopte un flot d'exécution itératif intégrant des résultats d'autres briques d'analyse. À chaque itération, des catégories candidates sont générées et sont toutes prises en compte dans les itérations suivantes. L'intérêt d'un tel système est de sélectionner définitivement le meilleur candidat uniquement à la fin du traitement afin de tenir compte de l'ensemble des éléments fournis par les différentes briques. Le système est comparé à des systèmes académiques et industriels.

1 Introduction

La reconnaissance des entités nommées (REN) dans un texte est une tâche consistant à repérer des éléments textuels et à les classer dans des catégories/types prédéfinies (*i.e.* noms de personnes, d'organisations, de marques, d'équipes sportives, *etc.*). La REN est souvent considérée comme l'une des briques de fondation des systèmes visant à structurer un texte tout-venant. Cette brique est généralement conçue de façon indépendante des autres briques du Traitement Automatique du Langage Naturel (TALN) et s'inscrit dans un flot d'exécution linéaire : les briques situées après la REN ne peuvent plus intervenir sur la reconnaissance des entités nommées, et celles situées avant ne peuvent bénéficier des résultats de la REN. Par exemple, la résolution des coréférences et l'extraction des relations sont généralement situées après la REN ce qui impose *de facto* des limites aux systèmes. Soient trois exemples :

(1) Paris visite Paris.
(2) Paris est triste. Elle pleure dans le salon.

(3) Firadsicht est une ville.

Dans l'exemple (1), les systèmes académiques et industriels testés [1] reconnaissent les deux mentions de "Paris" comme des lieux. Pourtant, l'application d'un module d'extraction de relations permettrait de construire le triplet "Paris_*lieu*, rencontrer, Paris_*lieu*" et de mettre en évidence l'inconsistance de l'annotation (un lieu ne peut rencontrer un lieu) pour conduire le système à réviser les solutions proposées. Dans l'exemple (2), les systèmes annotent "Paris" comme un lieu. La résolution de la coréférence "Elle" (*i.e.* "Paris") conduirait le système à mettre en évidence que la ville ne peut pas "pleurer dans le salon", conduisant ainsi le système à revoir son annotation. L'exemple 3 met en évidence les limites des systèmes dès lors que l'entité à annoter est absente des ressources dictionnairiques ("Firadsicht" dans l'exemple n'est pas reconnue). Un constat plus global est que les briques élémentaires à la compréhension du langage sont aujourd'hui encore (trop) cloisonnées tel que récemment souligné par Cartier (2015).

Nous nous focalisons dans ce qui suit sur une approche de REN symbolique (à base de règles). Les systèmes existants sont confrontés à plusieurs contraintes, en sus des contraintes d'ordre analytique présentés ci-avant. Tout d'abord, ils reposent généralement sur un ensemble de règles pour lesquelles la priorité d'application est cruciale (Maurel et al., 2011), impactant directement la qualité des résultats. Ensuite, les systèmes symboliques actuels s'appuient généralement sur des ressources de grand volume qu'il faut alors développer ou enrichir (Sagot et Stern, 2012)(Nouvel et al., 2012)(Nouvel et al., 2016). Enfin, le texte à analyser est généralement traité comme une suite de phrases indépendantes, ce qui réduit considérablement le contexte mis à disposition du système.

Nous proposons une façon de décloisonner à la fois les briques d'analyse du TALN et les phrases en mettant en place une approche itérative pour la reconnaissance d'entités nommées : chaque brique du TALN sollicitée lors d'une itération apporte à la REN de nouveaux éléments de contexte qui la mènent à des solutions plus pertinentes. Ces nouvelles solutions impliquent potentiellement les briques à réviser les éléments précédemment fournis, et ainsi de suite. Par ailleurs, le système tisse des liens entre les phrases grâce à la résolution des coréférences dans le but de propager à travers le texte les solutions identifiées par le système. Il s'ensuit que les types d'entités proposés par le système demeurent des candidats jusqu'à la dernière itération. Dans la suite, nous décrivons le système (section 2) et présentons les résultats de l'évaluation (section 3).

2 Description du système

2.1 Quelle typologie ?

Même si, dans la littérature, des centaines de types d'entités nommées ont été définies (Sekine et Nobata, 2004), les campagnes d'évaluation ou *shared tasks* les plus récentes ainsi que les solutions industrielles n'en utilisent que quelques dizaines tout au plus, selon les besoins

1. Une quinzaine au total, y compris pour l'anglais, dont : Google, Alchemy (IBM), Gate (Tablan et al., 2013), SEM (Dupont et Tellier, 2014), NERC-fr (Azpeitia et al., 2014), Polyglot-Ner (Al-Rfou et al., 2015), AllenNLP (Gardner et al., 2018)

applicatifs. Dans notre travail, nous utilisons l'ontologie NERD [2] qui propose un consensus entre les types des systèmes les plus populaires (Rizzo et Troncy, 2012). L'intérêt d'utiliser une telle ontologie réside notamment dans le fait que les solutions retournées par le système permettent d'inférer de nouvelles solutions par la relation de subsomption. Ainsi, une entité typée `nerd:SportsTeam` (*équipe de sport*) est nécessairement typée `nerd:Organization` (*organisation*). L'enjeu est donc de faire en sorte que le système retourne les types les plus spécifiques afin d'inférer, selon le cas d'application visé, des types plus génériques. Inversement, à partir d'un type identifié par le système (par exemple `nerd:Organization`), celui-ci peut être guidé par les types spécifiques possibles pour affiner la solution proposée (par exemple `nerd:Airline`, `nerd:Band` ou encore `nerd:Company`). Ce dernier cas n'est pas exploité par le système actuel et constitue une perspective à nos travaux.

Dans le cadre de ce travail, nous avons développé une extension à NERD en ajoutant des classes telles que les mesures, les noms de méthodes/théories, les récompenses, les lignes de transport, *etc* [3]. Au total, le système dispose de 115 classes et sous-classes.

2.2 Algorithmique

Le système est constitué de six phases principales dont les deux dernières sont itératives (*cf.* Fig. 1). Le texte est d'abord soumis à une analyse syntaxique (phase 1) qui fournit les dépendances entre chaque mot (*token*) de la phrase (par exemple les dépendances *sujet* et *objet*). La structure obtenue est soumise à une série d'expressions régulières permettant le repérage des mesures, e-mails, URLs, numéros de téléphone, *etc*. (phase 2). Les expressions temporelles sont ensuite reconnues dans la phase 3. La phase 4 projette les éléments du lexique de contexte-clés permettant de repérer des éléments de contexte en vue de la désambiguïsation des entités nommées. Par exemple, des termes tels que "société", "organisation", "entreprise" sont des éléments clés pour l'identification des organisations. La ressource lexicosémantique JeuxDeMots (Lafourcade et Joubert, 2008) a été utilisée pour construire de tels lexiques [4]. Les termes repérés sont des couleurs, des ingrédients, des métiers, des matériaux, des instruments de musiques, des nationalités, des sports, *etc*. La phase 4 projette également le lexique des entités nommées ce qui permet de générer les premiers candidats. La phase 5 correspond à l'application des règles en n itérations. Une fois les n itérations effectuées, la phase 6 intervient (une unique fois) dans le but d'augmenter la couverture en proposant de nouveaux candidats. La base de connaissances DBpedia est utilisée à titre expérimental : par l'extraction de n-grammes de mots, des mentions exactes et partiellement exactes sont recherchées dans la base pour augmenter la couverture du système. Il s'avère que les types fournis par la base (via la propriété `rdf:type`) sont représentés par des classes décrites dans plusieurs ontologies. Nous avons donc projeté ces classes sur celles de l'ontologie NERD en exploitant la propriété `sameAs` et les valeurs associées fournies par NERD [5]. Par exemple, `dbo:Place`, `wm:LOC`, ou encore `ner2:location` sont projetées sur `nerd:Location`. La phase 5 est ensuite à nouveau amorcée car de nouveaux éléments issus de la phase 6 pourraient impliquer le dé-

2. `http://nerd.eurecom.fr/ontology/`
3. Cette extension sera publiée prochainement.
4. `http://www.jeuxdemots.org`. Nous utilisons la relation de synonymie.
5. De nombreuses équivalences de classes ont été ajoutées manuellement pour couvrir les ontologies utilisées par DBpedia.

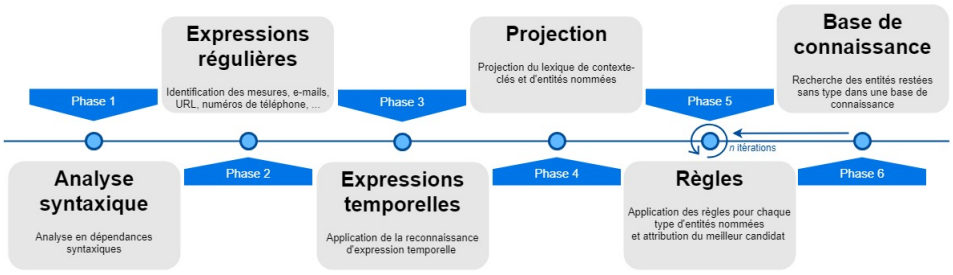

FIG. 1 – *Schéma de l'approche*

clenchement de nouvelles règles. Au bout de n itérations, le système fournit alors les résultats définitifs.

Le système exploite de nombreux éléments de contexte, y compris ceux qui pourraient être éloignés de l'entité à catégoriser, en couvrant plusieurs tâches du TALN dont l'extraction de la terminologie, l'extraction de relations et la résolution de la coréférence. Nous faisons l'hypothèse que, même si ces tâches menées de façon indépendante ne sont pas entièrement résolues, leurs traitements partiels mais combinés apportent des indices pertinents conduisant un système de REN à de meilleures performances. Ces traitements sont sollicités, sous forme de règles, lors de la phase 5 :

— *Acronymes*. Ce module identifie les acronymes entièrement en capitales par un ensemble d'heuristiques. Chaque acronyme est associé à sa forme complète lorsqu'elle existe dans le texte. Par exemple, dans "L'Organisation des Nations Unies (ONU) attire notre attention.", le sigle ONU est lié à "Organisation des Nations Unies" ; tous deux reçoivent les mêmes types candidats.
— *Complétion*. Ce module s'appuie sur les informations d'ordre morphosyntaxique pour compléter une entité nommée (notamment avec les adjectifs, noms communs et noms propres) qui serait partiellement identifiée. On évite ainsi que dans le nom du personnage "La Panthère rose", seul "La Panthère" soit reconnu.
— *Coordination*. La coordination exprime généralement une relation entre plusieurs éléments de même nature. Par exemple, dans "J'ai visité Lisbonne, Ajouda et Benfica.", le fait de connaître le type de Lisbonne (*i.e.* lieu) permet au système de proposer le même type pour les deux autres entités : Ajouda et Benfica.
— *Coréférences*. L'analyse des coréférences est actuellement limitée aux coréférences pronominales, basée sur la sortie d'un analyseur syntaxique. Par exemple, dans "Paris est triste. Elle pleure dans le salon.", "Elle" et "Paris" sont liés par un lien d'identité.
— *Descripteurs*. Les descripteurs, ou marqueurs, sont des mot-outils qui peuvent aider à désambiguïser une entité nommée. Par exemple, "à" et "en" sont plus probablement placés avant un lieu qu'avant la mention d'une équipe de sport.
— *Contexte gauche et droit*. Les relations de dépendances permettent d'identifier le contexte gauche et droit de l'entité, y compris lorsque les éléments textuels sont distants de l'entité nommée. Par exemple, dans "J'ai visité la ville, celle que j'avais toujours rêvé de visiter, Benfica.", "Benfica" est directement lié à "ville".
— *Preuve interne* (McDonald, 1996). Une entité nommée est parfois constituée d'élé-

ments permettant de déduire son type. Par exemple, "Association Valentin Haüy" et "Fédération Française de Football" contiennent "Association" et "Fédération" qui permettent au système de proposer le type candidat "organisation".

— *Appartenance* (Lopez et al., 2014). Ce module exploite l'expression d'appartenance entre deux entités nommées pour déduire une relation hiérarchique entre elles. Par exemple, dans "J'ai testé Revitalift de L'Oréal Paris.", sachant que "L'Oréal Paris" est une marque, la relation d'appartenance exprimée par la préposition "de" indique que "Revitalift" appartient à la marque et est probablement un produit.

— *Relations*. Ce module a pour objectif d'extraire des triplets de la forme "sujet, prédicat, objet". Par exemple, dans "Lilwenn est président du groupe.", le triplet "Lilwenn, être, président_*Function*" est généré. Ce triplet est soumis à une base de connaissances (générée à partir de JeuxDeMots) contenant le triplet "Person, être, Function" qui permet de déduire que le sujet est ici une personne. Les triplets sont générés en exploitant le résultat de l'analyse syntaxique (en particulier les relations sujet et objet, attribut du sujet, *etc.*), évitant ainsi le traitement proposé par Ezzat (2014) qui consiste à analyser les éléments textuels situés entre le prédicat et ses arguments.

— *Comparaison*. Ce module tire profit de l'expression de la comparaison. Par exemple, dans "Pierre est informaticien, comme Mart.", le système considère que Mart est probablement du même type que Pierre.

— *Expressions locales*. Les expressions locales permettent de capturer l'intégralité d'une entité complexe. Par exemple, l'expression "entre *Time* et *Time*" permet de repérer des entités temporelles telles que "entre lundi et jeudi" et "entre 2017 et 2019".

— *Projection par mémoire*. Ce module projette les candidats types reconnus à chaque itération sur l'ensemble du texte. Dans l'exemple "Quel avenir pour Tropico ? [...] Coca-Cola_*Organization* rachète Tropico.", la dernière phrase permet de reconnaître Tropico comme une organisation grâce à la relation "racheter", et propose cette solution pour la mention dans la première phrase.

— *Méta-règles*. Les méta-règles sont construites en utilisant les types proposés par le système. Par exemple, dans "Paris rencontre Marseille.", la méta-règle "Lieu, rencontrer, Lieu -> Equipe, rencontrer, Equipe" permet de typer Paris et Marseille comme des équipes de sport.

— *Propagation*. Ce module propage les types d'entités par le biais de la résolution des coréférences. Par exemple, dans "C'est Iban. Il a téléphoné à Marie.", "Il" est typé comme une personne [6] et est coréférent avec "Iban" que le système type également comme une personne.

Le nombre d'itérations est paramétrable et est pris en compte dans l'évaluation du système (section 3).

2.3 Sélection des candidats

Un candidat contient principalement trois données : le type de l'entité concernée, la règle ayant conduit à ce type, l'URI de l'entité concernée dans le cas de l'utilisation de DBpedia au cours de la phase 6. Un intérêt de l'approche proposée est de conserver les candidats pour une sélection définitive la plus tardive. À l'issue de chaque itération (phase 5), pour chaque

6. Grâce au triplet "Personne, téléphoner à, Personne"

entité nommée, un candidat est sélectionné tout en préservant les autres candidats. Si au cours du processus le candidat sélectionné est confronté à une inconsistance, celui-ci est remplacé par un candidat plus adapté. Pour ce faire, chaque règle est associée à un score de confiance déterminé empiriquement [7]. Par exemple, les candidats générés par le module *descripteurs* ont un score plus bas que les règles fondées sur le contexte gauche immédiat. Pour une entité donnée, si plusieurs candidats existent et proviennent de différentes règles, une agrégation des scores (somme) est calculée pour chaque type. Le candidat de meilleur score est retourné.

2.4 Exemple d'application

Nous décrivons pas à pas les phases du système à partir d'un exemple concret (4) :

(4) Matt Bowman (né le 31 mai 1991 à Chevy Chase, Maryland, États-Unis) est choisi par les Mets de New-York. Les Cardinals utilisent Bowman uniquement comme lanceur de relève. À sa première année, il joue 59 matchs des Cardinals. `https://fr.wikipedia.org/wiki/Matt_Bowman`

- **Phase 1 :** L'analyse syntaxique retourne la structure en dépendances qui sert de support à la suite du processus.
- **Phase 2 :** Les expressions régulières permettent d'annoter définitivement l'URL `https://fr.wikipedia.org/wiki/Matt_Bowman`.
- **Phase 3 :** Les expressions temporelles permettent d'annoter définitivement "31 mai 1991" comme une date.
- **Phase 4 :** Le lexique permet de classer New-York et États-Unis comme des lieux et les termes "lanceur_*sportif*" et "matchs_*évènement_sportif*" font désormais partie de l'ensemble des termes du contexte-clé.
- **Phase 5 : Première itération**
  - "Chevy Chase" : candidat "lieu", par déclenchement des règles suivantes : descripteurs ("*à* Chevy Chase"), coordination contenant un élément typé ("Chevy Chase, Maryland, *États-Unis*")
  - "Maryland" : même cas que "Chevy Chase".
  - "Bowman" : candidat "personne>sportif" par déclenchement de la règle *comparaison* ("Bowman [...] *comme* lanceur_*sportif*").
  - "il" : réfère à "Matt Bowman" et par extraction de relation fait partie du triplet "il, jouer, match" qui permet de déduire que "il" est un sportif, donc "Matt Bowman" est un sportif.
  - "Cardinals" : candidat "équipe de sport" grâce au contexte gauche "matchs des".
- **Phase 5 : Deuxième itération**
  - "Matt Bowman" : candidat "personne>sportif" par propagation de la coréférence "il" résolue dans l'itération précédente ; également candidat par présence de la preuve interne "Bowman" (annoté lors de l'itération précédente).
  - "Mets" : candidat "équipe sportive" par extraction du triplet "Matt Bowman_*sportif*, choisi par, Mets".

7. Ces scores sont les suivants : Acronyme : 0,8 ; Completion : 0,4 ; Coordination : 0,6 ; Coréférence : 0,3 ; Descripteur : 0,2 ; Contexte gauche et droit : 1,0 ; Appartenance : 0,6 ; Relation : 0,6 ; Comparaison : 0,9 ; Expression locale : 1,0 ; Projection par mémoire : 0,2 ; Méta-règles : 1,0

— **Phase 6 :** A ce stade, toutes les entités ont au moins un candidat. Cette phase peut générer de nouveaux candidats via DBpedia (paramétrable au lancement du système).

3 Évaluation

Dès l'apparition de la tâche de reconnaissance d'entités nommées, il a été démontré et largement accepté qu'une grande quantité de ressources constituait les fondations d'un système de REN (Wakao et al., 1996). Dans le contexte industriel, il a souvent été noté que le développement des ressources linguistiques pour la tâche de REN a un coût non négligeable (Ezzat, 2014). De fait, notre système limite les dépendances avec de telles ressources. Concrètement, hors DBpedia, le système dispose de ressources contenant seulement 26 000 entités nommées (dont 22 000 prénoms), de 1 500 termes (contexte-clé), et de 250 triplets (exemple d'un triplet : "Person, épouser, Person").

Les outils utilisés dans le cadre de cette évaluation sont à l'état de l'art :

— l'analyseur syntaxique Talismane (Urieli, 2013)

— l'analyseur d'expressions temporelles HeidelTime (Strötgen et Gertz, 2010) que nous avons enrichi de nouvelles règles.

Les autres modules décrits dans les sections précédentes ont été développés au sein de la société Emvista.

Pour la tâche de REN en français, les ressources libres annotées sont extrêmement rares. Dans le cadre de cette expérimentation, nous avons utilisé trois corpus : un corpus d'entraînement et deux corpus de test de genres différents :

— "Le tour du monde en quatre-vingts jours" de Jules Verne, 1872 (Lecuit et al., 2011), soient 84 976 *tokens*. Les 3 342 entités annotées dans ce corpus sont répartis en 12 types d'entités nommées (personne, organisation, lieu, place, vaisseau, bâtiment, oronyme, *etc.* [8]) que nous avons projetés sur les types de plus haut niveau de l'ontologie NERD (par exemple "place" est considérée comme `nerd:Location`) de sorte à comparer équitablement les résultats des différents systèmes.

— 280 résumés d'articles Wikipedia aléatoires, soient 10 034 *tokens*. Ce corpus a été annoté manuellement avec le format CoNLL et l'encodage BIO. Il est noté Wikipedia-train dans la suite. Il a été utilisé pour conduire le développement des règles et l'algorithmique.

— 587 résumés d'articles Wikipedia aléatoires, soient 21 855 *tokens*. Comme le précédent, ce corpus a été annoté manuellement en adoptant le format CoNLL et l'encodage BIO et est accessible sur Internet [9]. Il contient 3 125 entités nommées annotées avec les types de l'ontologie NERD. Ce corpus est noté Wikipedia-test dans la suite. Il n'a pas été utilisé lors du développement du système.

Le tableau 1 donne un aperçu de la constitution des corpus de test. La colonne T montre le nombre de tokens annotés pour chaque type, la colonne TU donne le nombre de tokens uniques. Enfin, la colonne TA donne le nombre de tokens ambigus pour chaque type (*i.e.* le nombre de tokens ayant au moins deux types différents).

Dans un premier temps, nous avons expérimenté l'apport de la phase 6 (DBpedia) sur le corpus Wikipedia-train. Les résultats obtenus montrent que l'apport de cette phase, qui in-

8. `http://tln.lifat.univ-tours.fr/Tln_Corpus80jours.html`
9. `https://www.emvista.com`

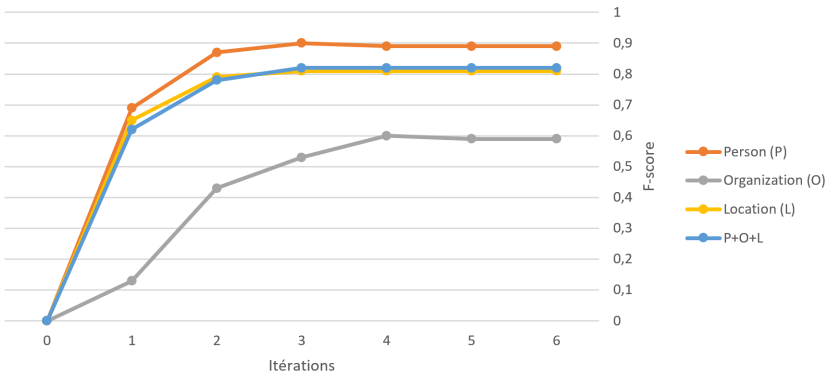

FIG. 2 – *Impact du nombre d'itérations sur Wikipedia-train*

tervient en fin de processus, est quasiment nul. Cela s'explique par le fait que l'analyse du contexte suffit à catégoriser les entités nommées : plus de 98% des entités ont été catégorisées grâce au contexte et non grâce à leur présence dans une base de données. Par conséquent, la phase 6 a été abandonnée dans la suite de l'expérimentation d'autant plus que celle-ci est coûteuse en temps d'exécution. Néanmoins, la suppression de cette phase ne doit pas être généralisée : par exemple, dans le cas des SMS et des tweets, le contexte est très limité et la syntaxe n'est pas standard : appliquer cette approche à de tels genres textuels ne serait pas pertinent.

Dans un deuxième temps, nous avons expérimenté l'impact du nombre d'itérations sur les résultats en utilisant le corpus Wikipedia-train. La Fig. 2 met en évidence que plusieurs itérations augmentent le F-score du système de façon significative. Sur ce corpus, une stabilité apparaît à partir de la quatrième itération. Nous montrons ainsi que le système est en mesure de proposer de nouvelles solutions à chaque itération. Une perspective à ce travail consiste à définir le cas d'arrêt optimal, automatiquement. Dans la suite de l'expérience, nous avons fixé un nombre d'itérations égal à 6.

Enfin, une expérience a consisté à attribuer le même score à chaque module de règles. Sur le corpus Wikipedia-train, la performance du système est diminuée de moitié ce qui montre la pertinence des scores attribués empiriquement. Une perspective consiste à expérimenter plus finement ces scores qui ont un impact immédiat sur les résultats.

Nous avons comparé les résultats de sept systèmes industriels et académiques sur Wikipedia-test et 80 jours : Google, Alchemy (IBM), Gate (Tablan et al., 2013), NERC-fr (Azpeitia et al., 2014), SEM (Dupont et Tellier, 2014), mXS (Nouvel et al., 2011) et notre système symbolique noté "Emvista". Les mesures classiques de précision (P), rappel (R) et F-score (F) sont utilisées. Le tableau de synthèse (*cf.* Tab. 4) présente les résultats en termes de micro mesures afin de tenir compte du déséquilibre des classes.

Sur le corpus "80 jours", le tableau 2 donne la première et la deuxième places (en terme de F-score) à Google et à Emvista. Avec un F-score global de 0,76, notre système obtient la première place (*cf.* Tab. 4). Le même scénario est observé sur le corpus Wikipedia-test (*cf.* Tab. 3). Notre système obtient le meilleur F-score global (0,81), proche du deuxième (0,80).

Il est intéressant de remarquer que, malgré la différence stylistique entre les deux corpus, le système d'Emvista est le plus robuste en ce sens qu'il ne montre que 0,05 point de différence

entre les deux corpus (*cf.* Tab. 4). Cela peut s'expliquer par le fait qu'il est plus autonome vis-à-vis des ressources que les autres systèmes. Enfin, les types de situations induisant le système en erreur sont majoritairement 1) les problèmes de frontières à gauche et à droite des entités nommées, 2) les erreurs de l'analyse syntaxique qui impacte directement la qualité du NER, 3) les erreurs relatives à la résolution des coréférences.

	80 jours			Wikipedia-test		
	T	TU	TA	T	TU	TA
nerd:Organization	387	121	61	373	277	103
nerd:Person	3 435	165	44	820	624	54
nerd:Location	2 041	438	100	1886	923	141
nerd:Time	-	-	-	1139	264	39
nerd:Product	-	-	-	318	267	110
nerd:Species	-	-	-	180	153	14
nerd:Function	-	-	-	421	248	103
nerd:Nation	-	-	-	316	74	26
nerd:Facility	-	-	-	108	75	38
nerd:Event	-	-	-	266	130	80

TAB. 1 – *Aperçu du contenu des corpus de test (T : nombre de tokens ; TU : nombre de tokens uniques ; TA : tokens ambigus)*

Systèmes	nerd:Person			nerd:Location			nerd:Organization		
	P	R	F	P	R	F	P	R	F
Alchemy	0,94	0,60	0,73	0,65	0,42	0,51	0,17	0,10	0,12
Gate	**0,95**	0,55	0,69	0,71	0,34	0,45	0,45	0,05	0,09
Google	0,88	0,64	0,74	0,81	**0,62**	**0,70**	0,37	**0,50**	**0,42**
NERC-fr	0,91	0,17	0,28	0,79	0,35	0,48	**0,75**	0,14	0,23
SEM	**0,95**	0,44	0,60	**0,85**	0,38	0,52	0,11	0,45	0,17
mXS	0,65	0,28	0,39	0,84	0,45	0,58	0,16	0,04	0,06
Emvista	0,92	**0,81**	**0,86**	0,74	0,58	0,65	0,61	0,33	**0,42**

TAB. 2 – *Résultats pour le corpus "Le tour du monde en quatre-vingts jours"*

4 Conclusion

Nous avons expérimenté un système visant à compenser l'utilisation d'une très faible quantité de ressources par une analyse du contexte faisant intervenir différentes briques du TALN au-delà des frontières imposées par les phrases : gestion de la coordination, extraction de relations, identification des acronymes, analyse des coréférences, *etc*. Le point essentiel du système réside dans le fait qu'il repose sur un flot d'exécution itératif, évitant ainsi de fixer un ordre d'exécution des différentes briques et règles. À chaque itération, le système est en mesure de

Systèmes	nerd:Person			nerd:Location			nerd:Organization		
	P	R	F	P	R	F	P	R	F
Alchemy	0,78	0,80	0,79	0,63	0,23	0,34	0,46	0,20	0,28
Gate	0,81	0,68	0,73	0,86	0,27	0,41	0,46	0,07	0,12
Google	0,77	**0,95**	0,85	**0,93**	0,74	0,82	0,68	**0,61**	**0,64**
NERC-fr	0,69	0,32	0,43	0,90	0,30	0,45	0,54	0,06	0,01
SEM	0,82	0,69	0,74	0,90	0,46	0,60	0,24	0,37	0,29
mXS	0,77	0,68	0,72	0,72	0,47	0,56	0,47	0,05	0,09
Emvista	**0,83**	0,93	**0,87**	0,91	**0,80**	**0,85**	**0,72**	0,38	0,49

TAB. 3 – *Résultats pour le corpus "Wikipedia-test"*

Systèmes	80 jours			Wikipedia-test		
	P	R	F	P	R	F
Google	0,82	0,62	0,71	0,85	**0,77**	0,80
Emvista	0,84	**0,70**	**0,76**	**0,86**	**0,77**	**0,81**
SEM	**0,87**	0,42	0,55	0,80	0,51	0,61
mXS	0,69	0,33	0,44	0,70	0,48	0,55
Gate	0,83	0,44	0,57	0,80	0,34	0,45
NERC-fr	0,86	0,23	0,35	0,81	0,28	0,41
Alchemy	0,79	0,50	0,62	0,64	0,36	0,43

TAB. 4 – *Synthèse des résultats sur les deux corpus pour les organisations, lieux et personnes*

proposer de nouveaux candidats, dont certains plus pertinents que ceux identifiés jusqu'alors. Ce système obtient des résultats supérieurs aux systèmes académiques et industriels testés. Les résultats sont très encourageants d'autant plus que le système est évolutif puisque de nouvelles règles peuvent être facilement ajoutées.

Une perspective immédiate à ce travail est l'amélioration des briques de TALN, particulièrement la brique de résolution des coréférences de sorte à tisser plus de liens entre les phrases. Il va de soi que cette brique est dépendante des résultats de la REN et d'extraction des relations, entre autres. De fait, nous développons un système itératif couvrant l'ensemble des briques du Traitement Automatique du Langage Naturel, dans le but d'expérimenter le décloisonnement total des différentes tâches du TALN.

Références

Al-Rfou, R., V. Kulkarni, B. Perozzi, et S. Skiena (2015). Polyglot-ner : Massive multilingual named entity recognition. In *Proceedings of the 2015 SIAM International Conference on Data Mining*, pp. 586–594. SIAM.

Azpeitia, A., M. Cuadros, S. Gaines, et G. Rigau (2014). Nerc-fr : supervised named entity recognition for french. In *International Conference on Text, Speech, and Dialogue*, pp. 158–

	P	R	F
nerd:Organization	0,72	0,38	0,49
nerd:Person	0,83	0,92	0,87
nerd:Location	0,91	0,80	0,85
nerd:Time	0,95	0,97	0,95
nerd:Product	0,84	0,41	0,55
nerd:Species	0,99	0,56	0,71
nerd:Function	0,63	0,85	0,72
nerd:Nation	0,94	0,92	0,93
nerd:Facility	0,51	0,45	0,47
nerd:Event	0,88	0,57	0,69
Total (micro-mesures)	0,86	0,78	0,81

TAB. 5 – *Détails des résultats pour quelques types du système d'Emvista (sur Wikipedia-test)*

165. Springer.

Cartier, E. (2015). Extraction automatique de relations sémantiques dans les définitions : approche hybride, construction d'un corpus de relations sémantiques pour le français. In *Conférence annuelle Traitement Automatique des Langues Naturelles*.

Dupont, Y. et I. Tellier (2014). A named entity recognizer for french (un reconnaisseur d'entités nommées du français)[in french]. *Proceedings of TALN 2014 (Volume 3 : System Demonstrations) 3*, 40–41.

Ezzat, M. (2014). *Acquisition de relations entre entités nommées à partir de corpus*. Ph. D. thesis, Paris, INALCO.

Gardner, M., J. Grus, M. Neumann, O. Tafjord, P. Dasigi, N. Liu, M. Peters, M. Schmitz, et L. Zettlemoyer (2018). Allennlp : A deep semantic natural language processing platform. *arXiv preprint arXiv :1803.07640*.

Lafourcade, M. et A. Joubert (2008). Jeuxdemots : un prototype ludique pour l'émergence de relations entre termes. In *JADT'08 : Journées internationales d'Analyse statistiques des Données Textuelles*, pp. 657–666.

Lecuit, É., D. Maurel, et D. Vitas (2011). Les noms propres se traduisent-ils ? étude d'un corpus multilingue. *Corpus 10*, 201–218.

Lopez, C., F. Segond, O. Hondermarck, P. Curtoni, et L. Dini (2014). Generating a resource for products and brandnames recognition. application to the cosmetic domain. In *LREC*, pp. 2559–2564.

Maurel, D., N. Friburger, J.-Y. Antoine, I. Eshkol, et D. Nouvel (2011). Cascades de transducteurs autour de la reconnaissance des entités nommées. *Traitement automatique des langues 52*(1), 69–96.

McDonald, D. D. (1996). Internal and external evidence in the identification and semantic categorization of proper names, corpus processing for lexical acquisition.

Nouvel, D., J.-Y. Antoine, N. Friburger, et A. Soulet (2011). Recognizing named entities using automatically extracted transduction rules. In *Language & Technology Conference*

(LTC'11).

Nouvel, D., J.-Y. Antoine, N. Friburger, et A. Soulet (2012). Coupling knowledge-based and data-driven systems for named entity recognition. In *Proceedings of the Workshop on Innovative Hybrid Approaches to the Processing of Textual Data*, pp. 69–77. Association for Computational Linguistics.

Nouvel, D., M. Ehrmann, et S. Rosset (2016). *Named Entities for Computational Linguistics*. John Wiley & Sons.

Rizzo, G. et R. Troncy (2012). Nerd : a framework for unifying named entity recognition and disambiguation extraction tools. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 73–76. Association for Computational Linguistics.

Sagot, B. et R. Stern (2012). Aleda, a free large-scale entity database for french. In *LREC 2012 : eighth international conference on Language Resources and Evaluation*, pp. 4–pages.

Sekine, S. et C. Nobata (2004). Definition, dictionaries and tagger for extended named entity hierarchy. In *LREC*, pp. 1977–1980. Lisbon, Portugal.

Strötgen, J. et M. Gertz (2010). Heideltime : High quality rule-based extraction and normalization of temporal expressions. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pp. 321–324. Association for Computational Linguistics.

Tablan, V., I. Roberts, H. Cunningham, et K. Bontcheva (2013). Gatecloud.net : a platform for large-scale, open-source text processing on the cloud. *Phil. Trans. R. Soc. A 371*(1983), 20120071.

Urieli, A. (2013). *Robust French syntax analysis : reconciling statistical methods and linguistic knowledge in the Talismane toolkit*. Ph. D. thesis, Université Toulouse le Mirail-Toulouse II.

Wakao, T., R. Gaizauskas, et Y. Wilks (1996). Evaluation of an algorithm for the recognition and classification of proper names. In *Proceedings of the 16th conference on Computational linguistics-Volume 1*, pp. 418–423. Association for Computational Linguistics.

Summary

Named entity recognition (NER) seeks to locate and classify named entities into predefined categories (persons, organizations, brandnames, sports teams, *etc.*). NER is often considered as one of the main modules designed to structure a text. In this article, we describe our symbolic system which is characterized by 1) the use of limited resources, and 2) the embedding of results from other modules such as coreference resolution and relation extraction. The system is based on the output of a dependency parser that adopts an iterative execution flow that embeds results from other analysis blocks. At each iteration, candidate categories are generated and are all considered in subsequent iterations. The advantage of such a system is to select the best candidate only at the end of the process in order to take into account all the elements provided by the different modules. The system is compared to academic and industrial systems.