Mot vide, mot plein ?
Comment trancher localement

Frédérick HOUBEN

GREYC – UMR 6072 – Université de Caen
Bureau S2 301 – campus II – BP 5186
F-14032 CAEN cedex
fhouben@info.unicaen.fr

Mots-clefs – Keywords

Traitements multilingues, découverte de mots vides, alternative à une stop-list, extraction de règles et de motifs fréquents.

Multilingual NLP, function words discovery, stop-list alternation, rules and frequent pattern mining.

Résumé – Abstract

Nous présentons une méthode multilingue de catégorisation en mot vide / mot plein à partir de corpus brut. Cette méthode fait appel à des propriétés très générales des langues ainsi qu'à des techniques issues de la communauté de la fouille de données.

We are presenting a NLP multilingual method for function word / content word categorization using no other resource than the raw text itself. This method uses very general linguistic properties and also engineering from data mining community.

1 Introduction

Ce travail est une tâche préliminaire dans le cadre de traitements linguistiques sans autres ressources que le texte à analyser lui-même. Cette absence de ressources permet d'analyser un texte sans identification préalable de la langue dans laquelle il a été écrit et permet aussi d'éviter le recours à une stop-list et le problème lié des mots homographes pouvant être à la fois mot vide ou mot plein selon le contexte (*son, vers, pas* …). Pour cela, nous utilisons des connaissances très générales (qualifiées parfois de méta-connaissances) sur les langues. Nous ne prétendons cependant pas qu'une telle méthode soit totalement universelle et nous nous contenterons de la réduire à certains groupes de langues aux propriétés communes (ici, les langues de type alphabétique non agglutinées).

Dans le cadre de cet article, nous nous intéresserons à la discrimination locale mot vide / mot plein. Cette tâche peut servir d'amorce dans des travaux d'extraction de candidats termes (Vergne, 2003) ou d'indexation automatique.

2 Mots vides, mots grammaticaux, function words

On trouve plusieurs définitions de ces mots vides qui varient dans l'inclusion ou la non inclusion des mots lexicaux reliés au thème d'un corpus donné (ce mot n'est pas discriminant dans le cadre de l'informatique documentaire et augmente la taille d'un index de manière inutile). Dans notre cas, nous nous situons plus dans l'optique de (Tesnière, 1969) et considérons comme mots vides l'ensemble des mots grammaticaux ainsi que les auxiliaires. Les mots pleins sont donc tous les autres.

Il est important de noter ici que les déductions étant faites à l'aide du contexte, elles sont totalement locales et peuvent donc différer pour une même graphie.

3 Propriétés linguistiques utilisées

Nous utilisons deux propriétés linguistiques très générales :

- Le « principe du moindre effort » de Zipf consiste à dire que le locuteur a tendance à raccourcir ce qui est d'usage courant (Zipf, 1949). En d'autres termes, les mots grammaticaux sont à la fois fréquents et de petite taille.
- La seconde propriété est une conséquence du principe de Saussure (Saussure, 1922) selon lequel « dans la langue, il n'y a que des différences ». Entre autre, nous pouvons calculer des différences locales pour discriminer les mots vides des mots pleins, différences entre les critères de taille et d'effectif tels que définis par Zipf.
- Enfin, nous faisons l'hypothèse que les mots vides et les mots pleins ne se succèdent pas n'importe comment et que certaines chaînes sont plus fréquentes que d'autre.

4 Processus de marquage

4.1 Une vue générale

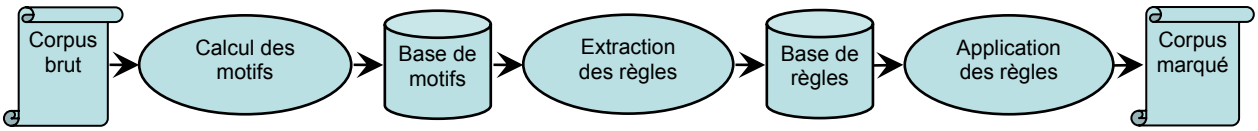

Figure 1: Processus de marquage mot vide / mot plein

Notre processus se décompose en trois étapes distinctes. Une première étape permet de collecter les motifs après un découpage du corpus en mots (et un comptage de l'effectif de chaque graphie). Dans la deuxième étape, nous utilisons le logiciel *mvminer*[1], pour extraire des règles à partir des motifs obtenus précédemment et enfin nous cherchons à appliquer ces règles pour affiner nos résultats. En définitive, nous obtenons notre corpus avec, pour chaque mot, l'indication du type calculé.

[1] Logiciel d'étude réalisé par François RIOULT (RIOULT 2003). Il extrait l'ensemble des motifs δ-libres qui constituent une représentation condensée des motifs fréquents. Les résultats fournissent l'association d'un motif libre et de sa fermeture (au sens de Galois), ce qui constitue une règle forte.

4.2 Calcul des motifs

Dans notre cas, on appellera motif une suite de 5 lettres[2] parmi les suivantes : P (Plein), s (signe de ponctuation), v (vide) ou n (non déterminé). Ces motifs représentent le type, d'après une déduction en contexte local des 5 mots d'une fenêtre.

Graphies	n'	a	connu	ni	arrêts
motif	v	v	P	n	P

Figure 2 : Exemple de motif pour une fenêtre

Dans cette partie, nous utilisons une fenêtre glissante (voir Figure 3), c'est-à-dire une suite de 5 mots consécutifs pris dans le sens de lecture. Dans cette fenêtre, nous allons effectuer des déductions locales pour déterminer le type des 5 mots puis faire glisser la fenêtre d'un cran dans le sens de lecture afin d'analyser tout le corpus.

Nous disposons, pour chaque mot de la fenêtre, de sa graphie, de son effectif total dans le corpus, ainsi que de la liste des déductions locales réalisées pour ce mot, dans les différentes positions qu'il a occupées dans la fenêtre aux précédentes positions de la fenêtre, avec donc des contextes légèrement différents à chaque fois.

Sens de lecture

[Graphies / effectif]	Déductions obtenues pour chaque graphie Ordre des déductions →				
[conquêtes 1]	P	P	P	P	P
[imprévues 1]		P	P	P	P
[. 108]			s	s	s
[La 5]				nd	v
[marche 1]					P
Motif de la fenêtre : P P s v P *conquêtes* est considéré comme un mot plein après recoupement en sortie de fenêtre					
[imprévues 1]	P	P	P	P	P
[. 108]		s	s	s	s
[La 5]			nd	v	v
[marche 1]				P	P
[triomphale 1]					P
Motif de la fenêtre : P s v P P *imprévues* est considéré comme un mot plein après recoupement en sortie de fenêtre					

Figure 3 : Exemple de fenêtre avec un glissement d'un mot

Nous allons calculer la moyenne géométrique des effectifs des mots de la fenêtre (sauf les ponctuations) ainsi que celle des tailles de ces mots. Les mots dont l'effectif est supérieur à la

[2] Nous avons déterminé la taille de ces motifs en estimant qu'un empan de 5 mots est suffisant pour garantir la présence de mots vides et pleins. En effet, puisque nous utilisons des différences locales, il faut être sûr que de telles différences existent dans notre fenêtre.

moyenne des effectifs et dont la taille est inférieure à la moyenne des tailles sont considérés comme vides. Les mots dont l'effectif est inférieur à la moyenne et dont la taille est supérieure sont considérés comme pleins. Dans tout autre cas, on considère qu'il y a indétermination.

Lorsque la fenêtre glisse d'un mot dans le sens de lecture, un nouveau mot y entre, un autre en sort. Le mot qui sort a été dans les 5 positions (sauf premiers et derniers mots du corpus) de la fenêtre et a donc fait l'objet de 5 déductions locales qui peuvent être différentes. Nous effectuons alors un recoupement de ces cinq déductions (dans l'état des travaux, nous prenons simplement la déduction majoritaire) pour attribuer un type unique « en sortie » à ce mot.

Lorsque nous avons parcouru le corpus et attribué à chacun des mots un type qui peut être soit vide, soit plein, soit encore non déterminé, nous disposons d'une liste des mots et de leur type. Nous allons donc essayer maintenant de trouver le type des mots non encore déterminés.

4.3 Extraction des règles

Les règles sont calculées par le logiciel *mvminer* à partir des motifs collectés dans la phase précédente. Les règles se présentent sous la forme classique « prémisses => conclusion ». Il s'agit de règles à prémisses minimales. Dans l'exemple ci-dessous, les prémisses sont le contexte suivant : un mot vide, un mot de type quelconque, un signe de ponctuation puis un mot vide. La conclusion nous indique que le mot de type quelconque est sans doute un mot plein. Les règles peuvent avoir des exceptions (tolérance en erreur) dont la quantité est paramétrable lors de l'exécution de *mvminer*.

v		s	v		=>	v	⇨ **P**	s	v	

Figure 4 : Exemple de règle

Nous ne gardons que les règles avec un support minimal qui est fonction de la langue étudiée afin de ne pas avoir des « règles d'exceptions ». De même, seules les règles avec au moins deux éléments dans les prémisses sont conservées. En effet, nous cherchons à déduire le type d'un mot d'après son contexte et les prémisses correspondent à ce contexte. Enfin, les règles contenant un élément de type indéterminé sont aussi rejetées puisque notre objectif est justement de lever cette indétermination.

Notons enfin, et cela est essentiel, que la manière dont les règles sont extraites garantit qu'il ne peut pas y en avoir deux contradictoires (prémisses identiques, conclusions opposées).

4.4 Application des règles

	des	conquêtes	imprévues	.	La	marche
Effectif	31	1	1	108	5	1
Type initial	v	P	P	s	n	P
Règle appliquée		P		s	⇨ v	P
Type final					**v**	

Figure 5 : Application d'une règle

Pour appliquer les règles, nous sortons du cadre de la fenêtre pour modifier le type des mots tel que défini après recoupement, type que nous avions appelé « en sortie ». Nous cherchons simplement à retrouver automatiquement un contexte qui correspond aux prémisses d'une

règle et nous vérifions que la conclusion de cette règle permettra de remplacer un mot dont le type n'avait pas encore été déterminé.

5 Evaluation et discussion des résultats

Nous avons cherché à évaluer ces résultats en deux temps. Tout d'abord le degré de catégorisation des mots dès la première étape, basée sur le simple calcul de double différence taille / effectif. Ensuite, la part d'amélioration apportée par l'utilisation des règles.

Le tableau ci-dessous présente les résultats sur un corpus en français (article de Coubertin sur le football). Ce corpus est de taille volontairement limitée pour permettre une validation à la main des résultats obtenus. Dans chaque colonne, nous mettons le nombre de graphies ayant été considérées comme des mots vides, comme des mots pleins ou de type non déterminé.

Les déductions locales sont les calculs effectués au cours de la première étape, à l'intérieur d'une fenêtre. Chaque mot apparaissant dans les 5 positions de la fenêtre, il y a approximativement 5 fois plus de déductions que de graphies.

Les recoupements sont les types définitifs, quand le mot est sorti de la fenêtre et que nous avons fait la synthèse des déductions locales effectuées sur ce mot. Là encore, ce sont des calculs effectués lors de la première étape. Cette fois-ci, il y a un type par graphie du corpus.

La dernière colonne montre les résultats obtenus après avoir extrait et appliqué les règles. Nous retrouvons les résultats précédents auxquels nous ajoutons les graphies dont le type n'était pas encore déterminé et qui l'a été grâce à l'application des règles.

3287 graphies dont 1118 différentes	Après déductions locales	Après recoupements en sortie de fenêtre	Après application des règles
Vides	6265	1262	1277
Pleins	6252	1256	1277
Indéterminés	2178 (24 %)	423 (14,4 %)	387 (13,2 %)

Figure 6 : Résultats sur un corpus français

De manière plus précise, nous remarquons que des cas difficiles sont ainsi résolus. Par exemple, *os, ans, vie, jeu* sont reconnus pleins malgré leur très petite taille (voir Figure 7) mais grâce au contexte dans lequel nous les avons lus. *Jeu* présentait même une deuxième difficulté puisque le texte analysé portait sur le football et que *jeu* faisait partie des mots liés au thème et donc à grand effectif (11 occurrences).

Graphies	règles	du	jeu	;	Il	les
Effectif / Type initial	5 / P	42 / v	11 / n	31 / s	39 / v	58 / v
Règle appliquée		v	⇨ **P**	s	v	

Figure 7 : Exemple de déduction réussie

Nous avons aussi cherché à calculer la valeur de ces résultats. Nous avons donc procédé à une annotation manuelle du corpus en mot vide / mot plein à laquelle nous avons comparé nos résultats de calculs. Nous avons regardé le nombre de mots auxquels un type a été donné (vide ou plein) pour calculer un taux de rappel ainsi que le nombre de ces mots dont le type donné est le bon afin d'établir une mesure de précision. Nous avons effectué ces deux mesures avant et après l'application des règles extraites, afin de pouvoir déterminer

l'amélioration apportée par cette méthode. Nous avons ainsi obtenu un rappel de 85,6 % avant et 86,84 % après application des règles pour une précision de 93,37 % avant et 93,11 % après.

Nous avons enfin cherché à évaluer la précision des modifications effectuées grâce aux règles. Nous avons simplement regardé quels étaient les mots ayant subit une modification de type (indéterminé avant l'application des règles, vide ou plein après) et dont le nouveau type était correct. Pour plus de détail, nous avons séparé ceux dont le nouveau type était vide de ceux dont le nouveau type était plein. 15 mots ont été déterminés vides dont 14 correctement et 21 mots déterminés pleins dont 15 correctement.

6 Conclusion

Cette méthode présente de bons résultats et permet d'obtenir, sans ressources préalables, une discrimination en mots vides / mots pleins y compris sur des cas difficiles comme les mots dont la taille ou l'effectif ne permettrait pas que nous les détections en utilisant simplement la loi de Zipf. Nous résolvons le problème des mots pleins de petite taille, des mots pleins de grand effectif (mots liés au thème du corpus notamment), le problème des mots vides avec majuscule (et donc faible effectif) et aussi certains cas d'homographie…

Cette méthode peut être vue comme un classifieur naïf dans son fonctionnement actuel. Elle présente surtout l'originalité de faire appel à des techniques issues de la communauté de la fouille de données qui peuvent s'appliquer à notre domaine avec une certaine réussite. Il faudrait maintenant comparer cette méthode avec des méthodes d'apprentissage déjà existantes comme les réseaux connexionnistes afin d'en voir la réelle efficacité dans l'optique de traitements sans ressource.

Nous envisageons bien sûr de développer et d'améliorer cette méthode notamment au travers d'une collaboration accrue entre François RIOULT et moi-même, chacun de nous amenant ses connaissances spécifiques et son regard neuf sur le domaine de l'autre.

Références

Ben Yahia S., Cherif C. L., Mineau G., Jaoua A. (2003), Actes des *3ème EGC*, 131-143.

Bourigault Didier (2002), Upery : un outil d'analyse distributionnelle étendue pour la construction d'ontologies à partir de corpus, *Actes TALN 2002, Nancy*, pp.75-84.

Do C. (2003), Posterior Decoding for Generative Constituent-Context Grammar Induction,

Grammatical Induction Community website : http://eurise.univ-st-etienne.fr/gi/

Klein D., Manning C. D.(2002), A Generative Constituent-Context Model for Improved Grammar Induction, *Proceedings of the 40th Annual Meeting of ACL*, 128-135.

Rioult F., Crémilleux B. (2003), Condensed Representations in Presence of Missing Values, *Proceedings of IDA'03, LNCS 2810*, p. 578-588.

Tesnière L. (1959), *Éléments de syntaxe structurale, Paris,* Klincksieck.

Vergne J. (2003), Un outil d'extraction terminologique endogène et multilingue, *Actes de TALN 2003*, tome 2, 139-148.

Wolff J. G. (2003), Unsupervised Grammar Induction in a Framework of Information Compression by Multiple Alignment, Unification and Search

Zipf G. K. (1949), Human Behaviour and the Principle of Least Effort, New York, Harper.