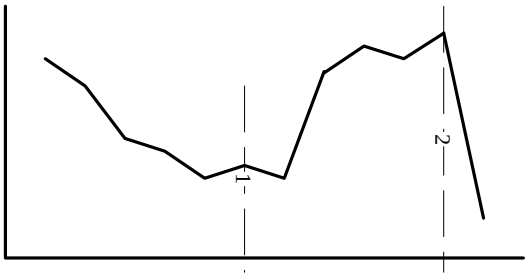
GILLES BERNARD

Laboratoire IA, Université Paris 8, France
gb@ai.univ-paris8.fr
http://ai.univ-paris8.fr/CSAR/gb.html

Détection automatique de structures syntaxiques

Il y a plus de trente-cinq ans, Z. Harris mit au point un algorithme d'analyse de chaînes signifiantes remarquable par sa simplicité et par l'absence de connaissances préalables : *recurrent stochastic dependency process*. Le principe de cet algorithme est le suivant : à mesure que s'accroît notre connaissance d'une chaîne signifiante, la prédictibilité du reste s'accroît.

Voyons l'exemple de Harris: « dogzw ri dispyuwt bliykwik r » (écriture phonologique de *dogs were indisputably quicker*). Il y a 29 phonèmes initiaux possibles, après *d* il n'y en a plus que 12, après *do* 8, après *dog* 29, après *dogz* 29, après *dogzw* 6, après *dogzw* 4, après *dogzw r* 29, etc. La remontée du nombre de symboles possibles après *dog* montre qu'il s'agit d'une chaîne signifiante, ainsi que *dogz* et *w r*. On peut donc découper en constituants une chaîne à partir de l'ensemble des chaînes, ce que montre la figure ci-contre.



Profil d'une séquence selon l'algorithme de Harris
(1 : descente, 2 : remontée et coupe)

Deux autres variantes sont proposées par Harris : examiner le nombre de successeurs de la chaîne divisé par le nombre de successeurs du dernier symbole, ou la moyenne du nombre de successeurs des successeurs.

Le but de la présente recherche était d'étudier la faisabilité de l'application de cet algorithme (apparenté aux chaînes de Markov) à l'extraction de chaînes syntaxiques à partir d'un corpus de textes de toutes provenances. Même si les résultats demandent encore à être affinés, il est aujourd'hui clair que, moyennant un certain nombre d'adaptations, cet algorithme est intégrable dans une chaîne de traitement du langage humain écrit.

Principes de traitement

Il fallait, pour pouvoir appliquer l'algorithme de Harris ici, surmonter quatre obstacles majeurs :

(1) Il suppose connu en totalité l'ensemble des symboles possibles ; même si on pouvait disposer de la liste de tous les mots possibles, leur trop grand nombre rendrait l'algorithme inapplicable.

(2) Il suppose connu l'ensemble des séquences possibles (pour déterminer le nombre de successeurs) ; même remarque.

(3) Il fonctionne mal en présence de symboles complexes, comme le graphème *qu* : le nombre de successeurs de *qu* est plus grand que le nombre de successeurs de *q*, et le mot *quatre* sera donc coupé *qu-atre*. Le même problème se pose pour les phrases, où *parce que* forme un seul mot.

(4) Si une séquence est reconnue comme constituant, toute chaîne contenant cette séquence sera coupée : ainsi, l'algorithme découpe correctement *anti-communiste*, mais aussi, à tort, *anti-que*.

Nous sommes donc loin ici de l'application relativement idéale décrite par Harris, où la liste des phonèmes est connue, ainsi que toutes les lois de séquences, et chaque phonème est représenté par un symbole — bien que le dernier obstacle ne puisse être évité. En effet, nous souhaitons extraire des constituants d'un ensemble peu homogène de textes de provenance variée, avec de nombreuses erreurs, qui font qu'on trouvera des séquences de mots qui ne devraient pas se succéder, suite à l'absence d'un point par exemple.

Le premier obstacle a été contourné en sélectionnant la liste des *marques*, ou mots grammaticaux, qui peut être totalement déterminée, et en considérant chaque mot non grammatical comme une occurrence de l'un des quatre symboles suivants : (a) mot commençant par majuscule en début de phrase, (b) séquence de mots commençant par majuscule, (c) séquence de mots commençant par minuscule, et (d) nombres.

Une séquence est donc associée à un pattern du type : « **déb* ne *min* pas le *min** ».

Le deuxième obstacle est contourné en prenant un corpus suffisamment grand pour que chaque pattern y ait une chance d'être représenté. Plus le corpus sera grand, mieux l'algorithme pourra fonctionner.

Pour surmonter les deux derniers obstacles, des procédures utilisant les statistiques d'emploi des formes complètent l'algorithme de Harris ; l'idée étant que la fréquence de *u* après *q*, par exemple, nettement plus élevée que la fréquence cumulée des autres successeurs de *q*, est l'indice d'une contrainte forte.

Stratégie adoptée

Le corpus initial choisi pour les tests est un corpus de 22 textes électroniques (ANSI) en français (environ 500.000 mots, ponctuation non comprise), extraits pour la plupart du corpus de l'ABU[1], incluant de nombreuses erreurs de saisie et de format ; principalement des romans mais aussi des textes juridiques et des articles.

Notre système commence par découper les textes en paragraphes, les paragraphes en phrases et les phrases en segments ponctués. Dans les segments ponctués, les séquences de mots non grammaticaux sont remplacées par les symboles correspondant. On supprime les segments ne contenant pas de marques ; la plupart sont réduits à un seul mot (ex. *Antoine !*). Le résultat est un fichier d'environ 64.000 chaînes, avec 460.000 occurrences de symboles, dont 281.000 sont des marques (290 des 311 marques de notre liste de départ y sont présentes) et les autres sont des occurrences des quatre symboles décrits ci-dessus.

Ensuite le système construit différents arbres à partir des patterns, sur lequel sont effectués plusieurs types de calculs : l'algorithme de Harris, les fréquences des différentes branches, la fréquence des coupes, et le rapport normalisé entre la distribution locale et la distribution globale des fréquences.

Enfin le système extrait les schèmes découpés, et calcule les ressemblances entre marques (à partir du produit scalaire et de techniques de clustering).

Premiers résultats

L'algorithme de Harris a été testé dans ses trois variantes. La variante de base, décrite ci-dessus, s'avère être la meilleure ; celle qui part du nombre de successeurs de la séquence divisé par le nombre de successeurs du dernier mot repère un grand nombre de coupes (près de 80.000) mais pratiquement deux sur trois sont erronées ; celle qui part de la moyenne des successeurs de successeurs détecte environ 25.000 coupes qui forment un sous-ensemble des coupes de la variante de base.

La variante de base a coupé 27.100 segments (42% du total), et a détecté 36.100 coupes. Les segments présentant une moyenne d'un peu plus de 4 syntagmes (3 coupes), on peut estimer à 200.000 le nombre de coupes présent dans le corpus. Par conséquent l'efficacité de cet algorithme, sur le total des coupes à trouver, est de 18%. Les segments courts (moins de 4 syntagmes) sont nettement mieux découpés (autour de 40%).

Les erreurs de coupe se montent à quelques dizaines, et les plus courantes sont du type (3) : *ainsi que, parce que, tandis que* sont coupés de leur suite.

Nous avons testé différentes stratégies pour supprimer ces mauvaises coupes :

- on ne coupe pas après un symbole dont la probabilité locale (fréquence normalisée) dans la séquence en cours est supérieure à un seuil ;
- on ne coupe pas après un symbole dont la déviation (rapport de la probabilité locale à la probabilité globale) est supérieure à un seuil ;
- on ne coupe que si la probabilité des coupes attestées en ce point est supérieure à un seuil ; il est nécessaire de prendre en compte la probabilité de la coupe et non l'existence d'une coupe, parce qu'on part d'un corpus réel, avec de nombreuses erreurs.

C'est la troisième stratégie qui s'avère la plus efficace, avec un seuil de 0,3, suivie de près par la deuxième (avec un seuil de 0,5) ; en revanche la première n'a qu'un faible rendement (par ex., elle empêche la coupure après *parce que* mais non après *parce qu*).

D'autres erreurs proviennent de l'insuffisance du corpus (erreurs du type (2)) : certaines chaînes de notre corpus ne contiennent pas assez de variantes ; ainsi l'algorithme a coupé :

« *maj* *nombre* *pour le* *min* *entre* | *les* *min* *et les* *min* »

En effet le corpus ne contient pas d'autres séquences commençant par « *maj* *nombre* *pour le* *min* ». Une partie de ces erreurs proviennent en amont de la chaîne de traitement : des erreurs de format, ou quelquefois des erreurs de découpage, ont généré des segments du type « *Chap. 1 La première fois qu'Aurélien vit Bérénice* ».

Ces coupes sont également supprimées avec la stratégie mentionnée. Les erreurs du type (4) sont plus difficiles à détecter ; un travail de contrôle manuel considérable est nécessaire pour les chiffrer et adopter une stratégie d'attaque, sans doute à l'aide de la déviation ; un successeur dont la probabilité locale est largement supérieure à celle des autres (mieux qu'à un seuil, comme dans la version implémentée ici) ne doit pas être coupé.

En revanche, le fait que certaines marques soient en même temps des mots non grammaticaux (ex. *ton, son, pas, point, personne*...), n'a pas gêné l'algorithme.

Augmentation des coupures

L'algorithme de Harris découpe mieux les débuts de segments (et donc les segments courts), parce que l'arbre construit est beaucoup plus riche vers sa racine que dans les branches. Une fois éliminées la plupart des erreurs constatées, en combinant l'algorithme de Harris avec la probabilité des coupures, on peut rendre récursif l'algorithme de Harris de la manière suivante : on n'effectue qu'une seule coupe, puis on réinsère ce qui suit la coupe dans l'arbre pour en enrichir la structure en largeur et diminuer sa profondeur. On continue jusqu'à ce qu'il n'y ait plus de nouvelles coupures.

---

[1] L'Association des Bibliophiles Universels offre un corpus de textes en français d'accès libre à http://cedric.cnam.fr/ABU.

Dans la partie récursive la probabilités des coupes n'est plus très efficace, car moins l'arbre est riche et moins il y a de coupes ; on a dû donc recourir à la déviation, avec un seuil de 0.2. L'efficacité de l'algorithme monte à près de 152.200 constituants repérés, soit près de 60% du nombre total de coupes du corpus initial (qui est une estimation, le corpus n'ayant pas été étiqueté à la main).

Ce résultat a été obtenu en utilisant la probabilité de coupure en parallèle à l'algorithme de Harris ; elle rajoute quelques milliers de coupes que l'algorithme de Harris ne détecte pas détecté (son efficacité seule est beaucoup moins grande).

Dans les deux cas on empêche la coupe si le symbole qui suit la coupe est de déviation locale (non normalisée) supérieure à 1.

Relations probabilistes intra-constituants

Les constituants générés forment des mini-patterns aux frontières floues, qui peuvent fonctionner comme de mini-règles d'analyse syntaxique. Les frontières sont floues quand le constituant se termine ou commence par un symbole représentant une séquence de mots non grammaticaux. La plupart du temps, pour ce que nous avons pu dépouiller, la séquence de droite est inclue dans le constituant.

La séquence de gauche appartient rarement au constituant, ce qui est attendu, puisqu'en français, comme dans beaucoup de langues, les marques se situent presque toutes au début du constituant. Celles qui se situent à droite sont en général précédées d'un tiret (ex. *-ci*), et on pourrait compléter l'algorithme en testant ce caractère. Nous ne l'avons pas fait, pour que l'algorithme ne soit pas spécifique au français. Pour ces marques, on peut aussi appliquer l'algorithme de Harris (avec nos modifications) en commençant par la fin des segments, pour découvrir les coupures à gauche. Par exemple, les coupes que l'algorithme actuel introduit parfois avant *pas* (erreurs dues à l'utilisation de la probabilité de coupure) sont éliminées par l'analyse à rebours[1]

Enfin, dans une petite proportion de cas, on a des constituants «invisibles», car ne contenant aucune marque grammaticale. La différenciation entre constituants minuscules et majuscules permet de repérer des frontières (il s'agit souvent d'un verbe suivi d'un nom propre, formant deux constituants), mais elles ne sont pas sûres (il peut s'agir, bien que beaucoup plus rarement, d'un adjectif suivi d'un nom propre), à moins de raffiner l'analyse : l'adjectif précédant le nom propre est obligatoirement précédé d'un déterminant (*le grand Alexandre*) , alors que le verbe peut être précédé d'une marque verbale (*il voit Alexandre*). Nous n'avons pas utilisé de connaissances de ce type dans ce travail.

L'appartenance au constituant des séquences de mots non grammaticaux doit donc être affectée d'un coefficient décroissant ; le premier mot y appartient à coup sûr, mais les suivants, surtout si la séquence est longue, peuvent contenir un constituant invisible.

D'autre part, les contraintes entre deux symboles successifs d'un même pattern, exprimées par la déviation probabiliste du second au contact du premier, sont un indice important de la solidité de la relation. Ainsi, pour *de*, la marque la plus fréquente du corpus avec près de 23.000 occurrences, les termes qui présentent la contrainte la plus nette (la déviation la plus forte) sont les déterminants (déviation comprise entre 10,4 et 1,3). Sur les 37 plus fortement liés (déviation supérieure à 2,3), 34 sont des déterminants, plus *moi, toi* et *presqu*'. Les termes qui ne sont là que par coïncidence (ex. *sommes*, à la faveur de la confusion entre la marque - le verbe être - et le pluriel du nom *somme*, ou *je* à cause de « *de je ne sais quoi* ») ont une déviation négative (0.2 pour *sommes*, 0.02 pour *je*).

Il y a toutefois une précaution à prendre : une déviation élevée, rapport entre probabilité locale et probabilité globale, peut être due à une probabilité locale élevée, ou à une probabilité globale très basse, auquel cas la validité du résultat doit être pondérée. Pour l'extraction des candidats à un constituant unique, citée plus haut, et plus généralement pour l'estimation des contraintes, nous avons filtré les déviations. Le principe était le suivant : plus le profil d'un symbole (nombre de successeurs ayant une fréquence supérieure à un certain seuil) est long, plus le symbole a des chances d'être polysémique, et par conséquent, pour que les résultats soient valables, plus il faut d'occurrences de ce symbole.

Nous avons donc éliminé les marques dont la valeur significative, donnée par la formule suivante, était inférieure à -0,1 :

((<Nombre d'occurrences> / 20) - <longueur du profil>) / <longueur du profil>

Autrement dit, un symbole qui a 20 occurrences ne sera retenu que s'il n'est suivi que par un symbole, s'il a 40 occurrences, il ne sera retenu que s'il n'est suivi que par un ou deux symboles, etc.

Construction des catégories

Les patterns construits sont en grand nombre ; pour faciliter leur interprétation, il convient de les regrouper par famille, pour générer des étiquettes à partir du corpus, qui seront ensuite éventuellement associées à des étiquettes linguistiques, comme « syntagme nominal », « syntagme verbal ».

Pour étudier la faisabilité de cette génération d'étiquettes, nous avons analysé différentes possibilités de catégorisation automatique. En utilisant les vecteurs de probabilité et de déviation établis à partir de l'ensemble

---

1. Elles seraient aussi éliminées en grande partie si on traitait les formes de *pouvoir* comme des marques, car l'absence de *pas* n'est fréquente que dans ce cas : *je ne puis*.

des successeurs de chaque symbole, nous avons construit des clusters de marques en groupant autour de chaque symbole l'ensemble des symboles qui présentent une forte similitude (mesurée par le produit scalaire).

La clusterisation à l'aide des probabilités donne de très mauvais résultats (un seul cluster ou presque pour l'ensemble des marques), à cause de la très grande fréquence relative des symboles représentant des séquences de mots non grammaticaux. On pourrait penser à les éliminer, mais on perdrait une information importante : par exemple, les marques {*M. MM. Mme Mmes Mlle*} sont en général suivies par une majuscule. En revanche, la clusterisation à l'aide du vecteur des déviations normalisé donne d'excellents résultats.

Voici quelques exemples :

- un agrégat des déterminants, répartis en différents sous-groupes de similitude supérieure à 0.8, mais fortement interconnectés par des similitudes autour de 0,6 en moyenne (quoiqu'on puisse trouver, entre les extrêmes des groupes, une similitude au plus bas à 0,35, entre *ma* et *cette*). *Autre*, isolé, se rattache à ce groupe avec une similitude de 0,45 en moyenne.
- à part, bien qu'ayant des liens avec le précédent, un agrégat des déterminants définis (*du, le, les, aux*...), avec une similitude moyenne de 0.7 ;
- un agrégat de l'ensemble des déterminants du nom propre : {*M. Mme Mlle MM.*}, avec une similitude moyenne de 0,7.
- l'agrégat des pronoms de troisième personne : {*il elle on y*}, similitude comprise entre 0.71 et 0.61 (sauf pour la similitude *il - y* : 0.41)
- l'agrégat des différentes formes du verbe *avoir*, où l'adverbe *jamais* s'est inséré, avec une similitude moyenne de 0,6.
- à remarquer la synonymie de *et, mais* et du début absolu de segment : 0,6 pour *et*, 0,56 pour *mais*, et 0,49 entre *et* et *mais*.

Problèmes et perspectives

Dans l'ensemble, les résultats obtenus (plus de trois coupures sur cinq détectées) sont extrêmement intéressants, pour deux raisons : (a) notre système n'a besoin que d'un très petit nombre de connaissances : la liste des marques grammaticales (b) il est instantanément portable dans n'importe quelle langue pour laquelle on dispose à la fois d'un corpus conséquent et de la liste de ses marques (inflexions non comprises). Nos patterns devraient pouvoir servir au tagging et à l'analyse syntaxique.

Notre corpus, bien qu'il ait permis d'affirmer la validité de notre approche, est trop petit pour éviter l'influence de caractéristiques propres à un texte sur les résultats. Un corpus de quelques millions de mots apparaît comme un minimum ici. Nous envisageons maintenant d'intégrer dans la chaîne de traitement des améliorations, dont certaines ont été évoquées dans le cours de l'exposé, sur un corpus beaucoup plus conséquent : faire fonctionner l'algorithme à partir de la fin des segments, étudier les séquences de marques qui sont en fait une marque unique (ex. *parce que*) de plus de deux mots (ex. *à peu près, peu à peu*). Le problème du contrôle de la qualité des résultats se pose encore plus crucialement dans ce cas, car l'estimation du nombre de coupures à découvrir peut difficilement reposer sur une analyse manuelle.

La bonne qualité des résultats en catégorisation automatique des marques nous permet d'envisager un apprentissage non supervisé de ces catégories par un réseau neuro-mimétique de type *Self Organizing Maps* (T. Kohonen), sur la base des déviations observées ; en effet ce type de réseau permet non seulement de générer des catégories, mais également d'évaluer la distance entre catégories : dans les observations que nous avons pu faire, il ressort que certaines classes de marques, comme les déterminants, sont organisées en plusieurs catégories et il serait intéressant de faire émerger des méta-catégories couvrant l'ensemble de ces marques (et donc des règles qu'elles impliqueraient).

Une fois les constituants catégorisés, on pourra envisager d'appliquer l'algorithme de Harris non plus aux mots mais aux constituants, pour repérer des méta-constituants, à deux conditions, dont nous ne pouvons pas savoir à l'heure actuelle si elles seront remplies : d'une part que le nombre de catégories de constituants soit raisonnablement petit, et qu'à l'inverse le corpus contiennent suffisamment d'instances de ces catégories.

Bibliographie

Bernard (Gilles) : « Experiments on distributional categorization of lexical items with Self Organizing Maps ». Proceedings of WSOM'97, Helsinki 1997 (voir aussi sur mon site : http://ai.univ-paris8.fr/CSAR/gb.html)
Bernard (Gilles) : « La linguistique et l'intelligence artificielle » in *Manuel d'Histoire des Sciences du Langage* [S. Auroux], Walter de Gruyter, Amsterdam, à paraître (id.)
Brill (Eric) : « Unsupervised Learning of Disambiguation Rules for Part of Speech Tagging » in *Natural Language Processing Using Very Large Corpora*. Kluwer Academic Press.1997
Briscoe (J.) : « Prospects for practical parsing: robust statistical techniques» in *Corpus-based Research into Language : A Feschrift for Jan Aarts* (de Haan & Oostdijk, eds), Rodopi, Amsterdam 1994.
Harris (Zelig) : *Mathematical Structures of Language*, John Wiley & Sons, New York 1968
Joshi (Aravind K.) : « The convergence of mildly context-sensitive grammatical formalisms » in Processing of Linguistic Structure, Santa Cruz 1987
Jones (B.). «Can punctuation help parsing?» in *Proceedings of the 15th International Conference on Computational Linguistics*, Kyoto, Japan, 1994.
Kohonen (Teuvo) : *Self Organizing Maps*, Springer, Heidelberg 1995
Lari (K.) & Young (S. J.) : « The estimation of stochastic context-free grammars using the Inside-Outside algorithm» in *Computer Speech and Language Processing*, 1990.
Joshi (Aravind K.) : «How much context-sensitivity is necessary for characterizing structural descriptions---Tree adjoining grammars» in *Natural Language Processing---Theoretical, Computational and Psychological Perspectives* (Dowty, Karttunen, Zwicky, eds), Cambridge University Press, New York 1985