RECITAL 2009, Senlis, 24–26 juin 2009

Induction de sens de mots à partir de multiples espaces sémantiques

Claire Mouton[1,2]
(1) CEA/LIST/LIC2M - BP 6 92265 Fontenay-aux-Roses Cedex
(2) Exalead S.A. - 10 place de la Madeleine - 75008 Paris
Claire.Mouton@cea.fr, Claire.Mouton@exalead.com

Résumé. Les mots sont souvent porteurs de plusieurs sens. Pour traiter l'information correctement, un ordinateur doit être capable de décider quel sens d'un mot est employé à chacune de ses occurrences. Ce problème non parfaitement résolu a généré beaucoup de travaux sur la désambiguïsation du sens des mots (Word Sense Disambiguation) et dans la génération d'espaces sémantiques dont un des buts est de distinguer ces différents sens. Nous nous inspirons ici de deux méthodes existantes de détection automatique des différents usages et/ou sens des mots, pour les appliquer à des espaces sémantiques issus d'une analyse syntaxique effectuée sur un très grand nombre de pages web. Les adaptations et résultats présentés dans cet article se distinguent par le fait d'utiliser non plus une seule représentation mais une combinaison de multiples espaces de forte dimensionnalité. Ces multiples représentations étant en compétition entre elles, elles participent chacune par vote à l'induction des sens lors de la phase de clustering.

Abstract. Words can have many senses. In order to process information correctly, a computer should be able to decide which sense of a word is used in a given context. This unsolved problem has generated much research in word sense disambiguation and in the generation of semantic spaces in order to separate possible meanings. Here, we adapt two existing methods to automatically distinguish words uses and senses. We apply them to multiple semantic spaces produced by a syntactic analysis of a very large number of web pages. These adaptations and the results presented in this article differ from the original methods in that they use a combination of several high dimensional spaces instead of one single representation. Each of these competing semantic spaces takes part in a clustering phase in which they vote on sense induction.

Mots-clés : espace sémantique, réduction de dimensions, Locality Sensitive Hashing, induction de sens, clustering de mots, objets multi-représentés.

Keywords: semantic space, dimensionality reduction, Locality Sensitive Hashing, Word Sense Induction, words clustering, multi-represented data.

1 Introduction

Un même mot peut parfois être porteur de différents sens (polysémie) tandis que deux mots distincts sont parfois porteurs du même sens (synonymie). L'interprétation que les humains font à l'aide du contexte discursif ou environnemental leur permet de distinguer l'emploi d'un sens par rapport à un autre ainsi que la référence identique à un même concept par deux mots différents. Différentes approches permettent aux machines de modéliser cette distance entre

les sens d'un même mot ou cette proximité sémantique entre les mots. Ces approches peuvent être fondées sur des ressources constituées manuellement (ontologies, réseaux lexicaux) ou sur des ressources automatiques comme les *espaces sémantiques* auxquels nous nous intéressons dans ce travail. Nous pensons que la projection des mots dans des espaces vectoriels permet non seulement de retrouver cette distance mais aussi de regrouper des ensembles de mots qui pourront définir les différents sens des mots polysémiques. C'est l'induction de ces sens (*Word Sense Induction, WSI*) qui fait l'objet de notre travail.

Le paradigme des espaces sémantiques est fondé sur l'hypothèse que le sens porté par les mots dépend de leur contexte (Harris, 1985). Un contexte peut se concevoir à différentes échelles, on peut donc définir différents types de contextes. Dans (Salton *et al.*, 1975), les contextes du *Vector Space Model* sont les documents dans lesquels apparaissent les termes du vocabulaire. On stocke donc dans une matrice de similarité de terme général f_{ij} la fréquence d'apparition du terme i dans le document j. Les termes du vocabulaire représentés par les lignes peuvent ainsi être projetés dans un espace sémantique où les dimensions correspondent aux différentes colonnes de la matrice et les coordonnées sont données par les valeurs de la matrice. Ainsi, plus deux termes apparaissent fréquemment dans les mêmes documents, plus ils seront proches dans l'espace sémantique. Dans (Lund & Burgess, 1996), les colonnes correspondent aux termes les plus fréquents du vocabulaire et la matrice stocke les comptes de cooccurrences entre les termes des lignes et ceux des colonnes sur des fenêtres de taille fixe (*n* mots consécutifs après avoir retiré les *mots vides* tels que les prépositions, pronoms, déterminants...). Plus deux termes apparaissent à proximité des mêmes termes, plus ils seront proches dans l'espace sémantique. Dans (Padó & Lapata, 2007) et (Grefenstette, 2007), un contexte est une relation syntaxique associée à un mot du vocabulaire. Nous donnons pour exemple le contexte *sujet de manger*. Tous les mots apparaissant dans ce contexte ont une certaine similarité sémantique (*entités capables de manger*) qui sera nuancée par l'ensemble des autres contextes.

A partir de ces espaces sémantiques où chaque terme possède une signature (son vecteur ligne) qui le rend plus ou moins proche d'un autre, on peut regrouper les termes entre eux en fonction de leur similarité. Induire des sens à partir de mots polysémiques peut se faire à partir de différentes sélections de termes. Dans (Lin, 1998), tout le vocabulaire est réparti automatiquement (*clustering*) en ensembles de sens proches formant ainsi des classes de synonymes. Chaque mot pouvant appartenir à plusieurs ensembles, les mots polysémiques voient ainsi leurs sens discriminés. L'approche de (Schütze, 1998) consiste à effectuer un clustering sur un certain nombre d'instances du mot pour lequel on veut induire des sens, en utilisant leurs contextes d'occurrences dans un corpus donné. Enfin, certains comme (Véronis, 2003), (Ferret, 2004) effectuent un clustering sur les meilleurs cooccurrents des mots à distinguer en sens tandis que (Pantel & Lin, 2002) regroupent leurs plus proches voisins.

Notre travail s'inscrit dans la lignée de ceux de (Pantel & Lin, 2002) car nous regroupons aussi des plus proches voisins mais il se différencie par la distinction de plusieurs espaces sémantiques dont nous combinons les spécificités. En effet, jusqu'à présent les travaux faisant usage des espaces sémantiques fondés sur les cooccurrences syntaxiques traitaient tous les contextes syntaxiques de la même façon, indépendemment du type de relation syntaxique formant le contexte (la seule distinction résidait dans le fait d'utiliser une relation ou non). Notre travail montre la variabilité des distances entre les mots selon les relations syntaxiques utilisées et explore les apports de la prise en compte spécifique des différentes relations. Il reprend en outre un algorithme de recherche rapide de plus proches voisins approximatifs développé par (Ravichandran *et al.*, 2005) et le modifie pour d'une part réduire sa complexité algorithmique et d'autre part distribuer les calculs sur les noeuds d'un cluster. Nous montrons que notre version

modifiée conserve autant d'information que la version originale.

La Section 2 de cet article détaille la conception des espaces sémantiques que nous utilisons. Nous présentons dans la Section 3 une méthode rapide de recherche des plus proches voisins ainsi que la méthode de réduction de dimensions associée. La Section 4 décrit les méthodes de clustering développées pour prendre en compte les disparités de chaque espace. Enfin la Section 5 met les résultats en perspective et donne quelques pistes sur les travaux à venir.

2 Description des espaces sémantiques

Les espaces sémantiques que nous utilisons pour l'induction de sens sont issus des travaux de (Grefenstette, 2007). Au moment où nous avons effectuer nos travaux, le corpus est constitué de deux millions d'urls de pages francophones sur lesquelles a été effectuée une analyse syntaxique par le système d'analyse LIMA du CEA LIST (Besançon & de Chalendar, 2005). Ce système effectue une analyse en dépendances syntaxiques de type *sujet_verbe*, *objet_verbe*, *compl_du_nom*, (...). Ces relations sont orientées, par exemple dans le syntagme *traitement des langues*, *langues* apparaît dans le contexte de *traitement* pour la relation *compl_du_nom* tandis que *traitement* apparaît dans le contexte de *langues* pour la relation *compl_du_nom* inverse.

Le dictionnaire utilisé pour la constitution des espaces sémantiques est constitué des 68000 mots les plus fréquents de la langue française. On associe un espace distinct à chaque relation, enregistrant les fréquences de cooccurrences des 68000 mots avec les 68000 contextes ainsi définis pour cette relation. On donne ci-dessous un extrait des matrices COD_V et COMPDUNOM construites avec les phrases suivantes : *On étudie actuellement les stratégies de traitement de signaux afin d'obtenir les résultats souhaités. (...) Le responsable d'un traitement de données doit obtenir le consentement préalable des personnes concernées avant toute utilisation de ces données. (...) Il a obtenu un Doctorat en Traitement Automatique des Langues.*

	COD_V		COMPDUNOM			
	étudier	obtenir	stratégie	traitement	consentement	utilisation
stratégie	1	0	0	0	0	0
résultat	0	1	0	0	0	0
consentement	0	1	0	0	0	0
doctorat	0	1	0	0	0	0
traitement	0	0	1	0	0	0
signal	0	0	0	1	0	0
donnée	0	0	0	1	0	1
personne	0	0	0	0	1	0
langue	0	0	0	1	0	0

TAB. 1 – Remplissage des matrices

Afin que l'information fournie par tous les mots, y compris les mots rares, puisse être prise en compte, nous travaillons avec des matrices d'informations mutuelles calculées à partir des matrices de fréquence. L'information mutuelle est calculée à partir de la formule 1, où P_i est la probabilité d'occurrence du terme décrit par la ligne i dans n'importe quel contexte de la relation donnée, P_j est la probabilité d'occurrence du contexte défini par la colonne j, et Pi, j est la probabilité de cooccurrence du terme i avec le contexte j pour la relation donnée. L'information mutuelle est positive si la probabilité de cooccurrence des termes i et j est plus grande que la probabilité attendue si ces événements étaient indépendants.

$$MI = P_{i,j} * log(\frac{P_{i,j}}{P_i * P_j}) \quad (1)$$

La même procédure est effectuée pour les relations inverses, ainsi que pour les cooccurrences dans des fenêtres de taille fixe (5, 10, 20). Au final, on dispose de 71 matrices creuses et carrées de 68000 dimensions. Nous gardons ces matrices séparées, distinguant ainsi 71 espaces sémantiques dans lesquels sont représentées les mêmes données. On verra par la suite que ces espaces conservent des informations différentes et complémentaires.

3 Plus Proches Voisins Approximatifs

La taille de ces matrices nécessite une réduction du nombre de dimensions afin de travailler sur des matrices de taille raisonnable. Les décompositions en valeurs singulières des méthodes d'Analyse Sémantique Latente (*Latent Semantic Analysis, LSA*) développées par (Landauer & Dumais, 1997) devenant assez lourdes sur nos matrices (complexité quadratique), nous nous tournons vers des méthodes de réduction par Hachage Sensible à la Localité (*Locality Sensitive Hashing, LSH*), qui sont plus adaptées à la taille de nos matrices.

3.1 Réduction de dimensions : *Locality Sensitive Hashing*

(Charikar, 2002) definit une famille de fonctions LSH produisant des empreintes sur lesquelles on peut calculer une approximation de la similarité cosinus beaucoup plus rapidement que dans l'espace d'origine. De plus, (Ravichandran *et al.*, 2005) montrent que ce hachage est particulièrement adapté pour mettre en place une méthode de recherche rapide de plus proches voisins approximatifs. Nous reprenons ici les grandes lignes de la méthode de hachage.

On tire d vecteurs unitaires $\overrightarrow{r}$ selon une distribution gaussienne. Ce tirage assure une répartition équidistribuée sur l'hypersphère unitaire. Soit une famille de fonctions définies par :

$$h_{\overrightarrow{r}}(\overrightarrow{u}) = \begin{cases} 0 \; if \; \overrightarrow{r}.\overrightarrow{u} \geq 0 \\ 1 \; if \; \overrightarrow{r}.\overrightarrow{u} < 0 \end{cases} \tag{2}$$

Soient deux vecteurs $\overrightarrow{u}$ et $\overrightarrow{v}$, la probabilité de tirer un vecteur aléatoire $\overrightarrow{r}$ définissant un hyper plan qui les séparera est égale à :

$$Pr[h_{\overrightarrow{r}}(\overrightarrow{u}) \neq h_{\overrightarrow{r}}(\overrightarrow{v})] = \theta(\overrightarrow{u}, \overrightarrow{v})/\Pi \tag{3}$$

Sur un nombre d de vecteurs tirés aléatoirement on peut mesurer cette probabilité. En effet, la probabilité qu'un hyperplan tiré aléatoirement ait séparé les deux vecteurs originaux u et v est la probabilité que cet hyperplan ait donné un bit différent pour les deux résultats du hachage de u et v. La formule 4 nous donne cette probabilité :

$$Pr[h_{\overrightarrow{r}}(\overrightarrow{u}) \neq h_{\overrightarrow{r}}(\overrightarrow{v})] = distance\_de\_Hamming(\overrightarrow{u}, \overrightarrow{v})/d \tag{4}$$

En combinant 3 et 4 on obtient donc l'approximation :

$$\cos(\theta(\overrightarrow{u}, \overrightarrow{v})) \approx \cos(distance\_de\_Hamming(\overrightarrow{u}, \overrightarrow{v})/d * \Pi) \tag{5}$$

3.2 Recherche rapide des plus proches voisins

Une recherche rapide du plus proche voisin approximatif dans un espace muni d'une distance de Hamming a été proposée par (Charikar, 2002) et reprise par (Ravichandran *et al.*, 2005). La

méthode consiste à tirer aléatoirement p permutations de d éléments. Pour chaque permutation, on permute les signatures bit à bit, on procède à un tri lexicographique de tous les éléments et on garde les B plus proches éléments des n éléments source dont l'approximation du cosinus est inférieur à un certain seuil. Cela fonctionne car une signature permutée est une représentation valide du vecteur d'origine. C'est en soi une signature par une famille de hachage.

Nous remarquons que la méthode de recherche proposée par (Ravichandran *et al.*, 2005) est optimale lorsque l'on recherche les plus proches voisins de tous les éléments de la matrice. En revanche elle ne permet pas de chercher les plus proches voisins d'un seul élément sans avoir à calculer ceux de tous les éléments. Nous avons implémenté cette méthode en MPI[1] afin de lancer l'algorithme en parallèle sur un cluster de machines et nous avons également apporté deux améliorations sur l'algorithme de recherche lui-même.

Nous nous intéressons à w_0, terme dont on cherche les plus proches voisins. D'une part, si on effectue un XOR de tous les éléments de la base avec le vecteur w_0 alors on peut se permettre de ne pas trier tous les éléments mais de n'extraire que les k premiers et de ne les trier eux-mêmes que par la suite. On a donc une complexité en $O(p.n)$ à la place de $O(p.n.log(n))$.

D'autre part, le tri lexicographique sur un vecteur de bits prend en compte en moyenne deux bits, ce qui nécessite d'avoir un très grand nombre p de permutations pour obtenir une bonne approximation. En effet, la probabilité qu'on ne prenne en compte que le premier bit est de 1/2, la probabilité pour qu'on prenne en compte deux et seulement deux bits est de $1/4$, la probabilité pour qu'on prenne en compte k et seulement k bits est de $(1/2)^k$. Le nombre moyen de bits pris en compte est donc l'espérance de la variable aléatoire de loi de probabilité $P_X(i) = (1/2)^i$. Cette espérance est donnée par la série $\sum_{k=1}^{n}(p_i.x_i) = \sum_{k=1}^{n}(k/2^k)$ convergeant vers 2.
C'est pourquoi, pour les p permutations pour lesquelles on va extraire les B plus proches voisins, on ne procède pas à une permutation bit à bit mais à une permutation de sous-parties de signature pour lesquelles le tri se fera sur le nombre de bits de chaque sous-partie. Pour exemple, si nous prenons des sous-parties de huit bits, on effectue une permutation sur ces sous-parties, puis un tri lexicographique sur des valeurs variant de 0 à 8. Le tri sur ces sous-parties conserve la propriété de rapprocher des vecteurs pour lesquelles la distance de Hamming est faible. En effet après le XOR, une sous-partie dont le compte de bits est égal à 0 correspond à une sous-partie identique à celle de la signature source.
Cela permettra de prendre en compte plus de bits lors du tri (la probabilité de prendre en compte les 8 bits de la 1ère sous-partie est de 1/2, les 16 bits de la première et deuxième sous partie 1/4, etc...) et donc de diminuer le nombre de permutations nécessaires à la précision de l'algorithme.

Prenons des sous-parties dont le nombre de bits est égal à $\alpha * d$. On obtient alors une complexité en $O(p.n\alpha.d)$ avec $\alpha.d << n$ pour l'extraction des b plus proches voisins d'un élément source et $O(p.B + k.\log(k))$ pour le tri des k meilleurs parmi les $p * B$ obtenus ce qui est négligeable car $k << n$, soit au final une complexité en $O(p_1.n\alpha.d$ au lieu de $O(p_2.n.log(n))$ pour la méthode originale avec $p_1 < p_2$ puisque notre méthode nécessite moins de permutations. Notre méthode est donc plus efficace si on souhaite rechercher les plus proches voisins d'un élément source au coup par coup. En revanche, s'il s'agit de rechercher les plus proches voisins de tous les éléments de la base (i.e. de calculer les valeurs de la matrice de similarité pour lesquels les cosinus sont supérieurs à un certain seuil), on doit répéter l'opération n fois, et on a donc une complexité en $O(p_1.n^2\alpha.d)$ qui est supérieure à celle de l'algorithme original.

[1]Message Passing Interface - http ://www-unix.mcs.anl.gov/mpi/

3.3 Résultats

Nous avons ainsi pu calculer la liste des k plus proches voisins de mots polysémiques pour chacun des espaces syntaxiques et nous voyons par exemple dans le tableau 2 les résultats à k=10 pour les mots *barrage* et *vol* dans différents espaces. Pour un mot comme *barrage* dont

barrage	COMPDUNOM	barrage,infrastructure,aménagement,amont,bâtiment,station,canal,installation,réacteur,parcours
	COD_V	barrage,barrière,pont,bâtiment,chantier,centrale,usine,station,installation,banc
	APPOS	barrage,pont,canal,empierrement,déchetterie,digue,viaduc,mousqueterie,raz,écluse
	APPOS.reverse	barrage,digue,déversoir,pont,électrolyseur,redresseur,lanier,route,lac,autoroute
vol	COMPDUNOM	vol, avion, voyage, retour, course, achat, opération, première, journée, premier
	COD_V	vol, voyage, retour, création, attaque, changement, mise, mariage, acte, fin
	APPOS	vol, meurtre, racket, fraude, chantage, assassinat, homicide, rapine, violence, crime
	APPOS.reverse	vol, meurtre, viol, prostitution, rapine, adultère, proxénétisme, idolâtrie, agression, luxure

TAB. 2 – 10 plus proches voisins des mots *barrage* et *vol*

les usages peuvent se décliner en : *construction sur un cours d'eau*, *infrastructure industrielle* et *obstacle*, les différents espaces retournent des listes de plus proches voisins où ces usages sont non différenciés. En revanche, on remarque que pour un mot comme *vol* les plus proches voisins obtenus sont assez différents selon les espaces utilisés. Certains ramènent des plus proches voisins orientés vers un sens précis. Les espaces contiennent donc des informations différentes que nous supposons utiles de garder distinctes.

4 Induction de sens de mots par clustering de mots multi-représentés

Dans notre approche, nous cherchons à regrouper les plus proches voisins d'un mot source dans des ensembles représentant chacun un usage. On souhaite parvenir à distinguer différents

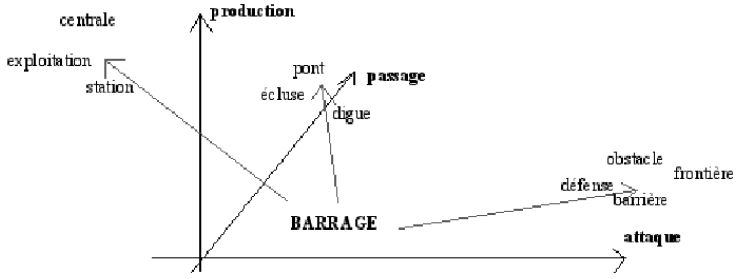

FIG. 1 – Discrimination de sens dans l'espace *complément_du_nom*

clusters comme dans l'exemple manuel de la Figure 1. Cette figure montre une projection à trois dimensions où les contextes devraient parvenir à discriminer trois significations du mot *barrage*. Dans cet exemple idéal, l'utilisation d'un unique espace vectoriel suffit. Mais nous avons vu plus haut que les différents espaces mettaient en relief différentes proximités. Cette distinction n'étant pas systématique selon les espaces et les mots traités, nous souhaitons prendre en compte la spécificité de chacun des espaces tout en permettant le regroupement inter-espaces. Pour cela, nous nous inspirons des algorithmes de clustering Shared Nearest Neighbours adaptés et utilisés par (Ertöz *et al.*, 2001) et (Ferret, 2004) ainsi que de l'algorithme Hyperlex développé par (Véronis, 2003) et nous proposons deux méthodes de clustering par vote.

Pour chacune des parties du discours pour lesquelles on souhaite induire des sens (*substantif, verbe, adjectif*), on conserve les espaces
– pertinents à ces parties (e.g. lorsqu'on traite un verbe, on délaisse l'espace COMPDUNOM qui ne concerne que les substantifs et on utilise l'espace COD_V inverse et non l'espace COD_V qui est entièrement creux pour les verbes cf. Tab 1)
– et correspondant à des relations entre mots pleins (substantif, verbe, adverbe, adjectif) (e.g. on délaisse la relation reliant un déterminant à son substantif).

Après expérimentation, nous choisissons de regrouper les mots qui apparaissent au moins deux fois dans les 100 plus proches voisins des espaces choisis. Soit E cet ensemble.

4.1 Méthode inspirée de l'algorithme Shared Nearest Neighbours

La méthode originale consiste à extraire un certain nombre de noyaux non fixé à l'avance, assigner les éléments restants à ces noyaux, et enfin à joindre les clusters dont les noyaux sont trop proches et réassigner les élements appartenant à des clusters jugés trop petits.

L'algorithme que nous proposons reprend la méthode de choix de noyaux de l'algorithme SNN de (Ertöz *et al.*, 2001) pour lequel on n'a pas à choisir le nombre de clusters a priori. Cependant l'expérimentation nous donnant de meilleurs résultats pour un graphe de voisins directs qu'avec le graphe des Plus Proches Voisins Partagés, nous garderons le premier. Une explication possible concernant ce résultat surprenant comparé aux résultats exposés dans (Ertöz *et al.*, 2001) est le fait que nous n'utilisons qu'une petite partie des élements de l'espace d'origine, on a donc trop peu de données pour exploiter ce second degré d'information. Nous conservons également l'idée de définition des noyaux à partir des *liens forts*. Le nom Shared Nearest Neighbours n'ayant plus de raison d'être, nous nous référerons désormais à cet algorithme sous le nom de MultiNN. Dans chacun des espaces (COD_V, SUJ_V...) :

1. On établit le graphe des plus proches voisins dans lequel les noeuds sont les élements de E et les arcs relient tous les noeuds entre eux et ont pour valeur la distance cosinus approchée que l'on calcule entre les deux noeuds extrémités.
2. On définit un seuil de lien fort (e.g. 0,2 de la distance maximale dans l'espace) et on élimine les arcs dont les valeurs sont inférieures à ce seuil.
3. Pour chaque noeud du graphe on calcule la somme des valeurs de ses liens restants.
4. Si cette somme est plus grande qu'un certain seuil (e.g. 0.3 de la somme maximale dans l'espace), on dit que ce point est un *noyau local* pour cet espace.

Pour tout élément de E, on sait le nombre d'espaces dans lesquels il est *noyau local*. Si ce nombre est supérieur à un seuil (e.g. 0.8 des espaces utilisés), cet élément est dit *noyau global*.

Nous constituons des clusters autour de chaque *noyau global* de la façon suivante. Dans chaque espace, on retire de E les éléments qui ont été désignés *noyaux globaux*. Puis, dans chacun des espaces et pour chaque élément de E, on enregistre un vote pour le *noyau global* le plus proche (cosinus approché le plus élevé). On somme les votes sur tous les espaces et on assigne chaque élément à son noyau le plus populaire (éventuellement à plusieurs en cas d'égalité).

Dans chaque espace, si deux *noyaux globaux* ont une valeur supérieure à un seuil donné, l'espace vote pour la jonction des deux clusters associés. Si un nombre suffisant d'espaces votent pour cette jonction, les deux clusters sont regroupés. Les éléments des clusters considérés trop petits par rapport au nombre d'éléments à regrouper sont réassignés un à un aux gros clusters.

4.2 Méthode fondée sur l'algorithme HyperLex

Nous adaptons une seconde méthode à notre multi-représentation des mots. Il s'agit de l'algorithme Hyperlex présenté par (Véronis, 2003). Celui-ci est à l'origine appliqué à une liste de cooccurrents mais dans notre cas nous l'appliquons à une liste de plus proches voisins.

L'algorithme original est le suivant. Soit E l'ensemble des mots à regrouper. Pour chacun d'entre eux, on calcule la distance au mot source. Le mot le plus proche devient noyau d'un cluster. On attribue à ce cluster tous les éléments appartenant à la sphère de rayon φ (si leur nombre est supérieur à un paramètre k) et on retire ces éléments de E. Le mot de E le plus proche du mot source devient noyau d'un autre cluster. On réitère jusqu'à ce que E soit vide.

Avec nos multiples représentations, le calcul des distances des éléments de l'ensemble E par rapport au mot source devient la moyenne des distances de tous les espaces. La spécificité des espaces n'est pas prise en compte à ce niveau-là. En revanche, les éléments sont attribués au cluster uniquement si un certain nombre d'espaces valide cette attribution (vote).

4.3 Résultats

Les résultats n'ont été évalués que manuellement dans cette première étape. Pour l'exemple du mot *barrage*, nous disposons des résultats fournis par les méthodes dont nous nous sommes inspirées. Nous présentons ainsi nos résultats pour ce mot dans le tableau 3. Nous avons privilégié un grand nombre d'éléments à regrouper afin de rassembler le maximum de sens possibles. Nous avons aussi opté pour un résultat contenant un grand nombre de clusters afin d'en obtenir le maximum possible de précis, quitte à en obtenir certains qui pourraient encore être regroupés. Un classifieur destiné à faire de la désambiguïsation et apprenant sur ces données saura ne pas classifier de mot dans l'une de ces classes si les éléments de celle-ci sont trop clairsemés et ont des équivalents dans d'autres clusters.
La comparaison de nos clusters avec ceux des algorithmes d'origine reste difficile puisque nous ne cherchons pas à regrouper le même type de terme (cooccurrents vs. plus proches voisins syntaxiques). Cependant nous pouvons voir que les sens distingués ne sont pas toujours les mêmes. Nous retrouvons d'une part en 3.1, 3.5 et 3.10 et d'autre part en 4.4, 4.11 et 4.12 les sens de *barrage* correspondant au *barrage routier, frontalier ou policier*, que l'on ne distingue pas bien entre eux. Le Petit Larousse utilisé dans la campagne ROMANSEVAL (Segond, 2000) ne fait lui-même pas la distinction et considère simplement ceux-ci comme *obstacle*. En 3.4, 4.5, 4.6, et 4.9, l'usage du *barrage hydraulique* est distingué en tant qu'*infrastructure industrielle* et en 3.7 et 4.8 en tant que *construction sur un cours d'eau*. Il s'agit du même objet physique mais l'usage est différent. En revanche on ne retrouve pas le sens du *match de barrage* pour lequel il existe très peu de mots partageant les mêmes contextes syntaxiques. Pour le mot *vol*, notre algorithme extrait correctement les sens *délit de vol* et *vol aérien*, ce qui n'était pas le cas pour l'algorithme HyperLex.

Une évaluation plus pertinente de ces différents types de clusters serait de les utiliser dans une tâche applicative (désambiguïsation de sens ou recherche d'information) et d'évaluer les résultats de celle-ci. Une telle évaluation automatique doit encore être mise en place pour estimer la meilleure approche et évaluer la qualité de discrimination des clusters produits.

Mot-source	barrage
HyperLex (Véronis, 2003)	1.1 : eau, construction, ouvrage, rivière, projet, retenue, crue
	1.2 : routier, véhicule, camion, membre, conducteur, policier, groupement
	1.3 : frontière, Algérie, militaire, efficacité, armée, Suisse, poste
	1.4 : match, vainqueur, victoire, rencontre, qualification, tir, football
SNN (Ferret, 2004)	2.1 : manifestant, forces_de_l'ordre, préfecture, agriculteur, protester, incendier, calme, pierre
	2.2 : conducteur, routier, véhicule, poids_lourd, camion, permis, trafic, bloquer, voiture, autoroute
	2.3 : fleuve, lac, rivière, bassin, mètre_cube, crue, amont, pollution, affluent, saumon, poisson
	2.4 : blessé, casque_bleu, soldat, milicien, tir, milice, convoi, évacuer, croate, milicien, combattant
MultiHyperLex	3.1 : **conflit, barrière, liaison, frontière, transport, environnement, division, combat, négociation, échange**
	3.2 : instrument, matériau, médicament, matériel, technologie, équipement, dispositif
	3.3 : poste, autoroute, mine, train, infrastructure, usine, distribution, logement, consommation, bateau, pression, commerce, bâtiment, installation, accord, marché, contrôle
	3.4 : **quartier, tunnel, centrale, port, station, commune**
	3.5 : **regroupement, arme, défense, accès, établissement, aménagement**
	3.6 : réseau, immeuble, proximité, financement, secteur
	3.7 : **ruisseau, écluse, digue, ville, route, île, parc, pont, sol, chemin, plage, forêt, canal, étang, lac, rivière**
	3.8 : tir, stockage, lutte, foyer, contrainte, chantier, édifice, bassin, coût, étape, développement, succession, possibilité, construction, dépôt, sortie, cours, marche
	3.9 : tour, hôpital, habitation, obstacle, concurrence, impact, appui, exploitation, présent, terrain
	3.10 : **zone, périmètre, ouvrage, limite, sécurité, unité**
MultiNN	4.1 : aménagement, amont, implantation, périmètre, proximité, édifice, emplacement, habitation, stockage
	4.2 : commerce, couverture, déchet, trafic, chantier, conflit, frontière, distribution, environnement, infrastructure, logement, transport, concurrence, mine, pêche, poste
	4.3 : développement, avantage, conflit, division, lutte, phase, présent, proximité, accord, construction, contrôle, cours, départ, environnement, essai, technologie
	4.4 : **échange, combat, conflit, liaison, ouverture, retrait**
	4.5 : **équipement, dispositif, instrument, opération, station, matériel, déplacement, emplacement, installation, matériau, médicament, ressource, stockage, train, usine, véhicule**
	4.6 : **établissement, division, fondation, réserve, station, commune, dépôt, exploitation, foyer, hôpital, immeuble, port, usine**
	4.7 : financement, couverture, entente, impact, regroupement, transfert, appui, concurrence, consommation, coût, investissement, négociation, revenu, soutien, succession
	4.8 : **forêt, bloc, ruisseau, étang, plantation, lac, rivière, bassin, édifice, parc, plage, sol**
	4.9 : **pont, chaussée, colonne, digue, tunnel, autoroute, canal, port**
	4.10 : possibilité, contrainte, étape, impact, limite, opération, obstacle
	4.11 : **secteur, division, impact, proximité, section, zone, environnement, exploitation, marché, réseau, terrain**
	4.12 : **sécurité, couverture, opération, présent, protection, proximité, unité, accès, défense, marche, poste**
	4.13 : ville, arme, centre, environ, grande, proximité, tour, bateau, bâtiment, chemin, feu, île, port, quartier, route, village

TAB. 3 – Comparatif des clusters construits à partir du mot *barrage*

5 Discussions et Perspectives

Bien que nous nous soyons dégagés du choix du nombre de clusters à obtenir afin de permettre un nombre de sens différents selon les mots, le choix des paramètres de ces deux algorithmes reste un problème dans le sens où nous n'avons pour l'instant pas de méthode pour apprendre les paramètres optimaux et nous restons ainsi contraints de les choisir par l'expérimentation.

Un avantage des méthodes présentées peut être mis en avant : nous supposons que le fait d'utiliser les plus proches voisins comme ensemble d'éléments à regrouper (et non les cooccurrents), permettra d'utiliser ces voisins comme données d'apprentissage pour un classifieur destiné à la désambiguïsation de termes ambigüs apparaissant dans un nouveau texte. Cette hypothèse sera expérimentée très prochainement dans la suite de ces travaux.

Par ailleurs, la finalisation de ces travaux nécessite diverses études à mener rapidement. Afin de mettre en valeur les apports de la méthode présentée ici, nous souhaitons former des clusters à partir des mêmes ensembles de plus proches voisins que dans cette étude mais en utilisant d'une part l'espace de cooccurrences de fenêtres seul ainsi que chacun des espaces syntaxiques seul, et en utilisant d'autre part les matrices concaténées.

Enfin, nous projetons d'utiliser ces clusters de sens pour désambiguïser et indexer une collection

de documents et étudier l'apport de cette désambiguïsation dans la recherche d'information.

Références

BESANÇON R. & DE CHALENDAR G. (2005). L'analyseur syntaxique de lima dans la campagne d'évaluation easy. In M. JARDINO, Ed., *Actes de TALN 2005 (Traitement automatique des langues naturelles)*, Dourdan : ATALA LIMSI.

CHARIKAR M. (2002). Similarity estimation techniques from rounding algorithms. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*.

ERTÖZ L., STEINBACH M. & KUMAR V. (2001). Finding topics in collections of documents : A shared nearest neighbor approach. In *Text Mine 01, Workshop of the 1st SIAM International Conference on Data Mining*.

FERRET O. (2004). Discovering word senses from a network of lexical cooccurrences. In *COLING '04 : Proceedings of the 20th international conference on Computational Linguistics*, p. 1326, Morristown, NJ, USA : Association for Computational Linguistics.

GREFENSTETTE G. (2007). Conquering language : Using nlp on a massive scale to build high dimensional language models from the web. In *Proceedings of the 8th CICLing Conference*, p. 35–49, Mexico.

HARRIS Z. (1985). Distributional structure. In J. J. KATZ, Ed., *The Philosophy of Linguistics*, p. 26–47. New York : Oxford University Press.

LANDAUER T. K. & DUMAIS S. (1997). A solution to plato's problem : The latent semantic analysis theory of the acquisition, induction, and representation of knowledge. *Psychlogical Review*, **104**(2), 211–240.

LIN D. (1998). Automatic retrieval and clustering of similar words. In *Proceedings of COLING-ACL98*, p. 768–774.

LUND K. & BURGESS C. (1996). Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments, and Computers*, **28**, 203–208.

PADÓ S. & LAPATA M. (2007). Dependency-based construction of semantic space models. *Comput. Linguist.*, **33**(2), 161–199.

PANTEL P. & LIN D. (2002). Discovering word senses from text. In *Proceedings of ACM SIGKDD Conference on Knowledge Discovery and Data Mining 2002*, Edmonton, Canada.

RAVICHANDRAN D., PANTEL P. & HOVY E. (2005). Randomized algorithms and nlp : Using locality sensitive hash functions for high speed noun clustering. In *Proceedings of ACL*, Ann Arbour(MI).

SALTON G., WONG A. & YANG C. S. (1975). A vector space model for automatic indexing. **18**(11), 613–620.

SCHÜTZE H. (1998). Automatic word sense discrimination. *Computational Linguistics*, **24**(1), 97–123.

SEGOND F. (2000). Framework and results for french. *Computers and the Humanities, Special Issue on SENSEVAL*, **34**(1).

VÉRONIS J. (2003). Cartographie lexicale pour la recherche d'information. In B. DAILLE, Ed., *Actes de TALN 2003 (Traitement automatique des langues naturelles)*, p. 265–274, Batz-sur-mer : ATALA IRIN.