TALN 2004, Fès, 19–21 avril 2004

Evaluation de méthodes de segmentation thématique linéaire non supervisées après adaptation au français

Laurianne Sitbon, Patrice Bellot
Laboratoire d'Informatique d'Avignon - Université d'Avignon
339, chemin des Meinajaries - Agroparc BP 1228
84911 AVIGNON Cedex 9 - FRANCE
Tél : +33 (0) 4 90 84 35 09, Fax : +33 (0) 4 90 84 35 01
{laurianne.sitbon, patrice.bellot}@lia.univ-avignon.fr

Résumé - Abstract

Nous proposons une évaluation de différentes méthodes et outils de segmentation thématique de textes. Nous présentons les outils de segmentation linéaire et non supervisée DotPlotting, Segmenter, C99, TextTiling, ainsi qu'une manière de les adapter et de les tester sur des documents français. Les résultats des tests montrent des différences en performance notables selon les sujets abordés dans les documents, et selon que le nombre de segments à trouver est fixé au préalable par l'utilisateur. Ces travaux font partie du projet Technolangue AGILE-OURAL [1].

This paper presents an empirical comparison between different methods for segmenting texts. After presenting segmentation tools and more specifically linear segmentation algorithms, we present a comparison of these methods on both French and English text corpora. This evaluation points out that the performance of each method heavilly relies on the topic of the documents, and the number of boundaries to be found. This work is part of the project Technolangue AGILE-OURAL.

Mots-clefs – Keywords

Segmentation thématique, métriques de Beeferman et WindowDiff, cohésion lexicale, chaînes lexicales
Topic segmentation, WindowDiff and Beeferman measures, lexical cohesion, lexical chains

1 Introduction

La segmentation d'un texte peut améliorer significativement les performances en recherche d'information (J.Callan, 1994), en donnant une partie de document correspondant à la requête,

[1] `http://www.technolangue.net/article.php3?id_article=79`

et en indexant les documents de manière plus précise. Elle peut aussi déterminer les limites entre des articles dans un flux d'informations (*broadcast news*), tâche représentée dans TDT (*Topic Detection and Tracking*). Enfin, elle peut représenter une importante part dans un processus de résumé automatique de textes, comme dans TXTRACTOR (McDonald & Chen, 2002). Après segmentation, les blocs sont évalués afin de conserver les plus pertinents, ou on produit un résumé pour chaque thème abordé, donc pour chaque segment.

Nous nous sommes intéressés aux méthodes non supervisées et produisant une segmentation linéaire (les segments trouvés sont adjacents). Nous avons testé quelques uns des outils les plus courants parmi lesquels DotPlotting (Reynar, 2000), C99 (Choi, 2000), Segmenter (Kan *et al.*, 1998), et Text Tiling (Hearst, 1997) - voir section 2. Ils ont été implémentés et comparés pour l'anglais par (Choi, 2000), mais *a priori* aucune comparaison de ces outils n'a été publiée pour des documents en français. Nous avons étudié leur comportement sur des textes français, après avoir effectué des adaptations liguistiques et créé un corpus français. Ces tests permettent de mettre en lumière d'une part les différences existant entre l'efficacité des outils, et d'autre part des différences entre les langues, les catégories de documents traités, et la taille des segments.

2 Méthodes de segmentation linéaire non supervisée

Les méthodes utilisées s'appuient sur la notion de cohésion lexicale (Halliday & Hasan, 1976), c'est à dire la répétition des mots comme indicateur d'homogénéité thématique. Nous présentons tout d'abord deux méthodes basées sur la notion de chaîne lexicale, puis deux autres fondées sur la seule répartition des mots.

2.1 Segmenter : chaînes lexicales

Segmenter (Kan *et al.*, 1998) effectue une segmentation linéaire basée sur les chaînes lexicales présentes dans le texte. Ces chaines relient les occurrences des termes dans les phrases. Une chaîne est rompue si le nombre de phrases séparant deux occurrences est trop important. Ce nombre dépend de la catégorie syntaxique du terme considéré. Une fois tous les liens établis, un poids leur est assigné en fonction de la catégorie syntaxique des termes en jeu et de la longueur du lien. Un score est ensuite donné à chaque paragraphe en fonction des poids et des origines des liens qui le traversent ou qui y sont créés. Les marques de segmentation sont alors apposées au début des paragraphes ayant les scores maximaux.

Etant donné qu'un concept peut être désigné par un ensemble de mots, le concept de chaînes lexicales a été élargi aux chaînes conceptuelles à l'aide de WordNet (Fellbaum, 1998) ou d'autres ressources sémantiques. (Kan *et al.*, 1998) montre que l'amélioration est très peu significative.

2.2 Text Tiling : multicritères sur la répartition des termes

Text Tiling (Hearst, 1997) étudie la distribution des termes selon plusieurs critères. Un score de cohésion est attribué à chacun des blocs de texte en fonction du bloc qui le suit. Ce score dépend lui-même d'un second score attribué à chaque paire de phrases suivant la paire de phrases qui la suit. Ce second score est calculé en tenant compte des mots communs, du nombre de mots

nouveaux, et du nombre de chaînes lexicales actives dans les phrases considérées.

Le score d'un segment de texte est alors le produit scalaire normalisé des scores de chaque paire de phrases qu'il contient. Si l'écart entre le score d'un segment et les scores du segment qui le précède et du segment qui le suit est grand, une frontière est apposée à l'intérieur de ce segment. La rupture entre deux unités thématiques est située dans une zone du texte entourée de zones présentant des valeurs de cohésion très différentes de la sienne.

2.3 Dot Plotting : répétition de termes

L'algorithme que nous utilisons est proposé par (Reynar, 2000), et est en fait une adaptation pour la segmentation de la méthode des nuages de points présentée par (Helfman, 1994) pour la recherche d'information. Il se base sur une représentation graphique du texte par les positions des occurrences des termes du texte à segmenter. Lorsqu'un terme apparaît à deux positions du texte x et y, les quatre points $(x, x), (x, y), (y, x)$ et (y, y) sont représentés sur un graphe, ce qui permet de déterminer visuellement les zones du texte où les répétitions sont nombreuses.

Cette méthode est adaptée par (Reynar, 2000) à la segmentation thématique de textes. Les positions de début et de fin des zones les plus denses du graphe sont les limites des segments thématiquement cohérents. La densité est calculée pour chaque unité d'aire en divisant le nombre de points d'une région par l'aire de cette région. A partir de là, deux algorithmes peuvent déterminer les frontières thématiques : indentifier les limites en maximisant la densité au sein des segments, ou repérer la configuration qui minimise la densité des zones entre les segments.

2.4 C99 : mesure de similarité

Cet algorithme proposé par (Choi, 2000) utilise une mesure de similarité entre chaque unité textuelle. L'idée de base de cette méthode est que les mesures de similarité entre des segments de textes courts sont statistiquement insignifiantes, et que donc seul des classements locaux (voir ci-dessous) sont à considérer pour ensuite appliquer un algorithme de catégorisation sur la matrice de similarité. Dans un premier temps, une matrice de similarité est donc construite, représentant la similarité entre toutes les phrases du texte à l'aide de la mesure de similarité proposée par (Rijsbergen, 1979), calculée pour chaque paire de phrases du texte, en utilisant chaque mot commun entre les phrases, et après « nettoyage » du texte : suppression des mots vides et lemmatisation.

On effectue ensuite un « classement local », en déterminant pour chaque paire d'unités textuelles, le rang de sa mesure de similarité par rapport à ses $m \times n - 1$ voisins, $m \times n$ étant le masque de classement choisi. Le rang est le nombre d'éléments voisins ayant une mesure de similarité plus faible, conservé sous la forme d'un ratio r afin de prendre en compte les effets de bord.

$$r = \frac{\text{rang}}{\text{nombre de voisins dans le masque}}$$

Enfin, la dernière étape détermine les limites de chaque segment de la même manière que l'algorithme Dotplotting (voir section 2.1) emploie la maximisation. En effet on cherche à déterminer quelle configuration offre la plus grande densité, en recherchant une nouvelle limite thématique à chaque étape. Les segments sont alors représentés par des carrés le long de la diagonale de la matrice de similarité modifiée avec les classements locaux. Pour chaque segment de la répartition proposée à une étape de la segmentation on considère son aire notée a_k et son poids s_k qui

est la somme des tous les rangs des phrases qu'il contient. On calcule alors la densité D de la configuration avec :

$$D = \frac{\sum_{k=1}^{m} s_k}{\sum_{k=1}^{m} a_k}$$

L'algorithme s'arrête lorsque la densité de la meilleure répartition proposée est suffisamment faible, ou si le nombre de frontières thématiques est déjà déterminé, lorsqu'il est atteint.

3 Adaptation au français et mise en place des tests

Nous avons adapté les implémentations effectuées par (Choi, 2000 ; `http://www.cs.man.ac.uk/~mary/choif/software.html`) au français. Une **liste de mots vides** permet de retirer du texte les mots trop communs et donc non suffisamment porteurs de sens pour les méthodes utillisées. De plus, étant très courants, ils peuvent fausser les mesures de similarité ou les chaînes lexicales en incluant des répétitions de termes supplémentaires et non pertinentes thématiquement. La liste de mots français utilisée se trouve dans (Veronis, 2004).

L'algorithme de Porter pour le ***stemming*** (Porter, 1980) est efficace pour l'anglais mais inadapté au français. Nous l'avons donc remplacé par une implémentation de Snowball (Porter, 2001), avec un fichier de paramètres pour le français livré avec l'outil.

L'étiquetage morpho-syntaxique est utilisé par l'outil Segmenter, et permet d'éliminer de l'analyse certaines catégories non spécifiquement porteuses de thème. Nous avons utilisé Tree Tagger, présenté par (Schmid, 1994). Comme les étiquettes morphosyntaxiques ne sont plus les mêmes d'une langue à l'autre, il a également fallu changer l'interprétation de l'étiquetage au niveau du code. Ce changement étant spécifique à chaque langue et non dépendant d'un fichier de paramètres, il est plus compliqué de porter Segmenter à une autre langue par la suite.

3.1 Métriques d'évaluation

Les deux standards d'évaluation que sont le rappel et la précision (Baeza-Yates & Ribeiro-Neto, 1999), utilisés en recherche d'information, ont été employés pour évaluer les premiers algorithmes de segmentation. Mais une baisse de l'un entraîne généralement une hausse de l'autre, et dans notre cas on ne cherche pas à en favoriser un, mais à évaluer l'efficacité des algorithmes pour la segmentation. De plus ils évaluent la présence ou non d'une frontière à sa place de référence, mais ne permettent pas de différencier une erreur faible (un décalage d'une phrase) d'une erreur grave (un oubli de frontière).

(Beeferman *et al.*, 1997) propose une nouvelle mesure, P_k, qui prend en compte la distance entre une limite trouvée et celle qui aurait dû être trouvée. Elle évalue une probabilité d'erreur sur la segmentation prenant en compte la probabilité pour deux phrases éloignées d'une distance k (valant la moitié de la longueur moyenne des segments du document) d'être localement dans les mêmes segments du document de référence (ref) et du document produit par l'outil (hyp), c'est à dire qu'aucune frontière thématique ne les sépare dans les deux cas.

(Pevzner & Hearst, 2002) montre que la mesure P_k proposée par Beeferman, bien que meilleure que le rappel/précision, a encore des défauts (des erreurs de même type sont pénalisées différement, les ajouts de petits segments sont ignorés, la signification de la mesure n'est pas claire).

Ils ont donc proposé une nouvelle manière d'évaluer les algorithmes de segmentation, appelée WindowDiff, fortement inspirée de P_k. En fait cette nouvelle mesure prend en compte le nombre de frontières séparant deux phrases espacées d'une distance k.

$$\textit{WindowDiff(ref, hyp)} = \frac{1}{N-k}\sum(|b(ref_i, ref_{i+k}) - b(hyp_i, hyp_{i+k})|)$$

où $b(x_i, x_j)$ est le nombre de frontières entre i et j dans le texte x, contenant N phrases.

(Pevzner & Hearst, 2002) montre que cette mesure est d'une grande stabilité face aux variations des tailles des segments, et qu'elle est aussi sévère avec les ajouts qu'avec les oublis de frontières thématiques. Cependant elle peut être supérieure à 1, et ne peut donc plus être assimilée à un taux d'erreur. Il est désormais évident qu'elle n'est qu'un élément de comparaison de la fiabilité des méthodes, et non pas un indice absolu de leur qualité.

3.2 Le corpus d'évaluation.

(Hearst, 1997) et (Kan *et al.*, 1998) ont travaillé sur peu de documents segmentés manuellement. A la place, nous avons repris la méthode de (Choi, 2000) pour constituer un corpus de tests même si elle ne garantit pas que les segments assemblés aient pour autant une véritable discontinuité thématique. Cette méthode, si elle ne correspond pas à un cas réel, présente l'avantage d'être rapidement applicable à des corpus de nature différente. Nous avons ainsi choisi de l'appliquer d'une part à des articles du Monde (fortes discontinuités thématiques) et, d'autre part, à des chapitres de la Bible (relative continuité stylistique et thématique mais fortes variation sur les entités nommées).

Un premier corpus a été réalisé afin de déterminer l'influence des thèmes et de la taille des segments. Ce corpus est divisé en 5 **catégories de documents** : les articles journalistiques de n'importe quel sujet, les articles journalistiques thématiquement cohérents : art, économie, sciences ou sport, et les chapitres littéraires. Pour chacune de ces catégories, 4 ensembles de 100 documents ont été fabriqués, avec pour chaque ensemble des critères différents de **tailles de segments** : des segments courts (entre 3 et 5 phrases), des segments moyens (entre 6 et 8 phrases), des segments longs (entre 9 et 11 phrases), ou des segments de tailles variables (entre 3 et 11 phrases). Chaque document est constitué de 10 segments, qui sont autant de début d'articles choisis aléatoirement dans la catégorie correspondant au critère, et raccourcis à la longueur définie par le critère de taille. La base d'articles utilisée pour composer ce corpus est l'ensemble des articles de l'année 2001 du Monde correspondant à chaque critère, et de plus de 10 phrases. A cette base s'ajoute un ensemble de chapitres de la Bible, pour la partie du corpus concernant la littérature. Ce corpus principal permet de tester de grandes variations dans la taille des paragraphes.

Un second corpus plus petit a été constitué, où les segments sont des ensembles de paragraphes. Il est constitué de 100 documents dont les segments sont issus de n'importe quelle catégorie, et comportent entre 3 et 5 paragraphes.

4 Expériences

Une fois les transformations des outils effectuées, nous avons tout d'abord comparé les résultats trouvés sur un corpus généraliste en français à ceux trouvés sur un corpus généraliste en anglais

(*Brown Corpus*, utilisé par (Choi, 2000)), afin de vérifier que les adaptations n'ont pas affecté certains outils. Quel que soit le test effectué, les résultats sur les deux langues font apparaître une différence constante de performance, d'une valeur d'environ 0,1 en faveur de l'anglais selon la mesure WindowDiff. Cependant, cette différence étant faible, on peut considérer que les changements effectués sur les outils sont valides.

4.1 Evaluation avec un nombre de segments à trouver non prédéfini

Le tableau 1 représente, pour 5 implémentations des méthodes (avec les paramètres donnant les meilleurs résultats), la moyenne de la mesure WindowDiff sur tous les documents du corpus principal. Ces implémentations sont décrites dans le tableau 5 en annexe. Cela montre que C99 est l'outil le plus efficace, deux fois meilleur que TextTiling et Segmenter.

Le corpus utilisé, en prenant les phrases comme unités de traitement, restreint l'action de Segmenter et TextTiling pour la création de chaînes lexicales, car ils se basent habituellement sur des paragraphes, et donc les blocs unitaires étant plus courts, il est plus difficile d'y former des chaînes lexicales avec les paramètres de base. Nous avons donc comparé les résultats avec ceux sur le second corpus, composé de paragraphes. La figure 1 montre que les différences sont faibles pour les meilleures méthodes.

JSegmenter	0,78
C99 basic	0,27
Text Tiling	0,53
JTextTile	0,9
Segmenter	0,50

TAB. *1: Comparaison des méthodes pour lesquelles le nombre de segments à trouver n'est pas prédéfini, selon la moyenne de la mesure Window Diff sur l'ensemble du corpus français*

Catégories	
Test no	score
25	4
36	4
37	4
38	4
13	3
32	3
34	3
35	3

Tailles	
Test no	score
25	4
35	3
36	3
37	3
34	2
38	2

TAB. *2: Meilleures implémentations (voir Tableau 5 pour leur description) pour chaque catégorie, et pour chaque taille des segments*

4.2 Evaluation avec un nombre de segments à trouver prédéfini

Ces méthodes donnant des résultats assez proches les uns des autres, nous avons créé une sorte de tournoi afin de trouver les meilleures, selon des critères de taille, ou selon chacune des catégories. Les 5 meilleures implémentations à chaque test reçoivent un point, pour chaque catégorie d'une part, et chaque taille d'autre part. Le tableau 2 montre les résultats de ce tournoi. Il en ressort que C99 (tests 25 et 32 à 38) est plus efficace que DotPlotting (test 15) lorsque le nombre de frontières thématiques à trouver est connu à l'avance.

4.3 Utilisation d'un nombre approximatif de segments à trouver

Etant donné que les outils pour lesquels il faut fournir le nombre de segments à trouver ont naturellement des résultats significativement meilleurs que les autres, nous les avons testés en leur donnant un nombre différent de segments à trouver, à plus ou moins 5 segments du nombre réel, afin de voir leur comportement dans le cas d'un nombre approximatif de limites.

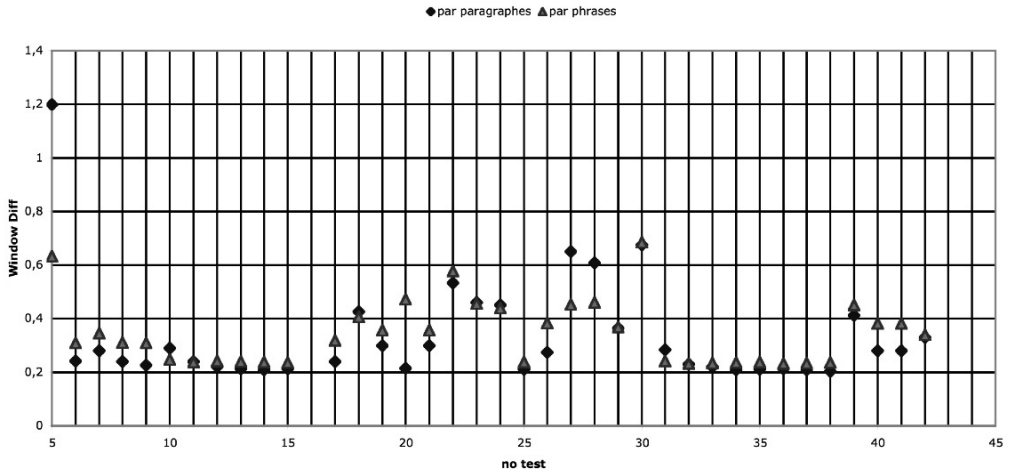

FIG. *1: comparaison entre les résultats pour le corpus principal, et le corpus secondaire, selon la mesure Window Diff*

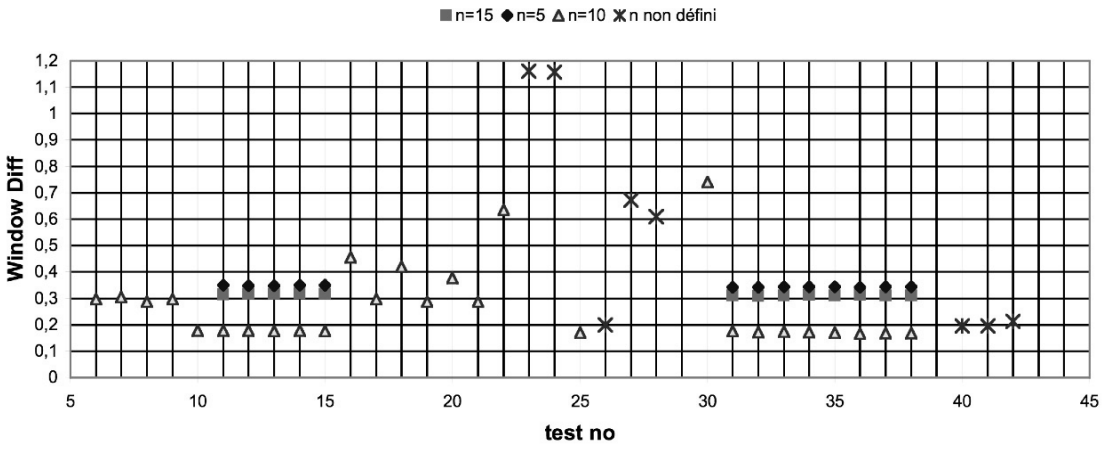

FIG. *2: résultats suivant un nombre n variable de segments prévus*

La figure 2 montre que même avec une erreur d'un ordre de la moitié du nombre réel de segments, les résultats sont aussi bons que les méthodes statistiques avec un nombre de segments à trouver connu et certains paramètres, et sont meilleurs que TextTiling ou Segmenter dans cette expérimentation. Cela suggère qu'on pourrait utiliser une granularité de segmentation définie par l'utilisateur, sans baisser trop les performances par rapport à des systèmes entièrement automatiques qui ne permettent pas ce paramétrage.

4.4 Efficacité par catégorie ou par taille

Le tableau 3 résume l'évaluation pour les trois implémentations les plus performantes de chaque catégorie toutes tailles de segments confondues, et de chaque taille de segments toutes catégories confondues (moyenne arithmétique des scores). Les résultats pour les segments entre 3 et 5 phrases sont les moins bons, cela montre donc qu'il est plus difficile de déterminer les frontières des petits segments. De plus les résultats pour des segments entre 3 et 11 phrases sont moins

bons que ceux entre 6 et 8 phrases, alors que la taille moyenne des segments est 7 phrases dans les deux cas. On peut en déduire qu'une forte variation dans les tailles des segments est également un frein au bon fonctionnement des outils.

A propos des catégories, les résultats montrent que les données littéraires sont plus difficiles à traiter que les données journalistiques. Pour ces dernières, le thème de l'économie ainsi que les articles d'ordre général sont les plus aisés à segmenter. On peut supposer que les variations thématiques entre les segments de ce corpus sont plus grandes. Les documents du corpus généraliste sont en effet issus de toutes les thématiques du journal, et les documents traitant d'économie contiennent beaucoup de noms propres différents d'un segment à l'autre.

Taille	Window Diff
3-5	0,2244
3-5	0,2269
3-5	0,2270
6-8	0,1927
6-8	0,1927
6-8	0,1946
3-11	0,2131
3-11	0,2131
3-11	0,2178
9-11	0,1669
9-11	0,16722
9-11	0,1678

Catégorie	Window Diff
Général	0,1670
Général	0,1681
Général	0,1686
Art	0,2327
Art	0,2335
Art	0,2338
Bible	0,2558
Bible	0,2565
Bible	0,2567
Eco.	0,1672
Eco.	0,1679
Eco.	0,1686
Sciences	0,1812
Sciences	0,1813
Sciences	0,1814
Sport	0,2114
Sport	0,2118
Sport	0,2121

TAB. *3: meilleurs résultats toutes implémentations confondues pour chaque catégorie et chaque taille de segments.*

Catégorie	Méthode	Test no	Window Diff
Art	C99	37	0,2811
Art	C99	38	0,2814
Art	C99	36	0,2845
Bible	DotPlot	13	0,3139
Bible	DotPlot	14	0,3140
Bible	C99	25	0,3161
Eco.	C99	33	0,2243
Eco.	DotPlot	13	0,2262
Eco.	DotPlot	12	0,2264
Sciences	C99	25	0,2132
Sciences	C99	35	0,2132
Sciences	C99	34	0,2179
Sport	C99	32	0,2839
Sport	C99	38	0,2850
Sport	C99	37	0,2854

TAB. *4: les 3 meilleures implémentations, pour chaque catégorie de document, pour des tailles de segments entre 3 et 11 phrases.*

4.5 Paramétrage de Dotplotting ou C99

C99 et Dotplotting sont les outils les plus efficaces (*cf.* sections 4.2 et 4.3), même dans le cas d'une valeur approchée (*cf.* section 4.4) pour la prédiction du nombre de segments à trouver. Mais le tableau 4 qui présente les 3 meilleures implémentations pour chaque catégorie, et pour une taille des segments donnée (entre 3 et 11 phrases), montre que le meilleur paramétrage est fonction du thème des documents, ainsi que du type de document à segmenter (littéraire ou journalistique). Ceci nous amène à suggérer l'évaluation sur un plus grand nombre de catégories, afin de déterminer statistiquement la meilleure méthode à utiliser suivant le contenu des documents. Il s'agirait alors d'une méthode supervisée s'appuyant sur des outils non supervisés.

5 Conclusion

Les expériences que nous avons conduites sur des documents en français montrent que dans les conditions où le texte à segmenter est une suite d'articles bien distincts, et où la qualité des outils est évaluée automatiquement en fonction de la distance entre les frontières trouvées et celles à trouver, l'outil C99 est le plus efficace. Les expériences ont confirmé le fait que le type de document que l'on segmente, son thème, la taille et la variation de taille des segments

à repérer, sont autant de caractéristiques influençant le travail des segmenteurs. D'autre part, ces méthodes de segmentation pourraient être améliorées en y intégrant des critères de choix de limites supplémentaires lorsque des marques de mise en forme sont présentes dans le texte (titre, changement de paragraphe, ...). Enfin, nous avons montré que moyennant de petites adaptations, les méthodes classiques de segmentation non supervisés de documents en anglais sont également efficaces sur des textes en français.

La prochaine étape de notre travail consistera à tester des méthodes supervisées et semi-supervisées pour finalement proposer une approche hybride (symbolique et probabiliste) capable de s'adapter au mieux à de nouveaux contextes et de profiter de la mise à disposition de corpus d'apprentissage en français. Cela sera effectué dans le projet Technolangue AGILE-OURAL.

Références

BAEZA-YATES R. & RIBEIRO-NETO B. (1999). *Modern Information Retrieval*. Addison-Wesley.

BEEFERMAN D., BERGER A. & LAFFERTY J. (1997). Text segmentation using exponential models. In *Proceedings of the 2nd conf. on Empirical Methods in Natural Language Processing*, USA.

CHOI F. Y. Y. (2000). Advances in domain independent linear text segmentation. In *Proceedings of the 1st Meeting of the North American Chapter of the Association for Computational Linguistics*, USA.

C. FELLBAUM, Ed. (1998). *WordNet, an electronic lexical database*. The MIT Press.

HALLIDAY M. & HASAN R. (1976). *Cohesion in English*. Longman.

HEARST M. A. (1997). Text-tiling : segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, p. 59–66.

HELFMAN J. I. (1994). Similarity patterns in language. In *IEEE Visual Languages*, p. 173–175.

J.CALLAN (1994). Passage-level evidence in document retrieval. In *Proccedings of the ACM/SIGIR Conference of Research and Development in Information Retrieval*, p. 302–310.

KAN M.-Y., KLAVANS J. L. & MCKEOWN K. R. (1998). Linear segmentation and segment significance. In *Proceedings of the 6th International Workshop of Very Large Corpora (WVLC-6)*.

MCDONALD D. & CHEN H. (2002). Using sentence selection heuristics to rank text segments in txtractor. In *Proceedings of the 2nd ACM/IEEE Joint Conference on Digital Libraries*, p. 28–35.

PEVZNER L. & HEARST M. A. (2002). A critique and improvement of an evaluation metric for text segmentation. *Computational Linguistics*, p. 19–36.

PORTER M. (1980). An algorithm for suffix stripping. *Program*, p. 130–137.

PORTER M. F. (2001). Snowball : A language for stemming algorithms. http ://snowball.tartarus.org.

REYNAR J. C. (2000). *Topic segmentation : Algorithms and applications*. PhD thesis, University of Pennsylvania, Seattle, WA.

RIJSBERGEN C. J. V. (1979). *Information Retrieval*. Buttersworth.

SCHMID H. (1994). Probabilistic part-of-speech tagging using decision trees. In *International Conference on New Methods in Language Processing*, p. 44–49, Manchester, UK.

VERONIS J. (2004). stoplist. http ://www.up.univ-mrs.fr/ veronis/index.html.

Annexe

Méthode	Numéro de test	Paramètres
JTextTile (implémentation de TextTiling par Choi)		
	5	avec les paramétres par défaut
Implémentation de DotPlot par Choi, maximisation, cosine sim, stem, mots vides retirés		
	6	Normal
	7	Lissage
	8	Réduction du bruit
	9	Seuil maximal
	10	Masque 3x3
	11	Masque 5x5
	12	Masque 7x7
	13	Masque 9x9
	14	Masque 11x11(*i.e.* C99)
	15	Masque 13x13
	16	Masque 11x11 avec avec activation de la diffusion
	17	Masque 11x11 , rang utilisé en tant que poids dans la matrice de similarité
DotPlot basé sur les blocs, maximisation, densité des points comme similarité		
	18	Steming activé mais pas de retrait des mots vides
	19	Steming activé mais et retrait des mots vides
	20	Activation de la diffusion
Implémentation exacte de DotPlot		
	21	Maximisation
	22	Minimisation
JSegmenter (implémentation de Segmenter par Choi)		
	23	Modèle de distance fixe
	24	Modèle de distance adaptatif
C99		
	25	Nombre de frontières connu
	26	Nombre de frontières inconnu
TextTiling		
	27	Paramètres par défaut
	28	Paramètres suggérés par Hearst, k=6, w=20
Simulation		
	29	Le document est découpé en 10 segments réguliers.
C99 avec des masques variables		
	30	Masque 1x1, *i.e.* tous les rangs sont à 0
	31	Masque 3x3
	32	Masque 5x5
	33	Masque 7x7
	34	Masque 9x9
	35	Masque 11x11
	36	Masque 13x13
	37	Masque 15x15
	38	Masque 17x17
Segmenter		
	39	Segmenter 1.6 avec les paramètres par défaut
Autres tests avec C99		
	40	Prise en compte de l'entropie calculée avec la fréquence des termes
	41	Prise en compte de l'idf calculé avec la fréquence des termes
	42	C99 avec un Masque 3x3, et une terminaison automatique

TAB. *5: description des implémentations avec les paramètres utilisés*