Pré-segmentation de pages web et sélection de documents pertinents en Questions-Réponses

Nicolas Foucault Sophie Rosset Gilles Adda

LIMSI-CNRS - 508 rue John von Neumann - Plateau du Moulon
Université de Paris-Sud - B.P. 133 - 91403 Orsay Cedex - France
prenom.nom@limsi.fr

RÉSUMÉ

Dans cet article, nous présentons une méthode de segmentation de pages web en blocs de texte pour la sélection de documents pertinents en questions-réponses. La segmentation des documents se fait préalablement à leur indexation en plus du découpage des segments obtenus en passages au moment de l'extraction des réponses. L'extraction du contenu textuel des pages est faite à l'aide d'un extracteur maison. Nous avons testé deux méthodes de segmentation. L'une segmente les textes extraits des pages web uniformément en blocs de taille fixe, l'autre les segmente par TextTiling (Hearst, 1997) en blocs thématiques de taille variable. Les expériences menées sur un corpus de 500K pages web et un jeu de 309 questions factuelles en français, issus du projet Quaero (Quintard *et al.*, 2010), montrent que la méthode employée tend à améliorer la précision globale (top-10) du système RITEL–QR (Rosset *et al.*, 2008) dans sa tâche.

ABSTRACT

Web pages segmentation for document selection in Question Answering

In this paper, we study two different kinds of web pages segmentation for document selection in question answering. The segmentation is applied prior to indexation in addition to the traditionnal passage retrieval step in question answering. In both cases, the segmentation is textual and processed once the web pages textual content has been extracted using our own extraction system. In the first case, a document is tilled homogeneously in text blocs of fixed size while in the second case the segmentation is based on the TextTiling algorithm (Hearst, 1997). Evaluation on 309 factoid questions and a collection of 500K French web pages, coming from the Quaero project (Quintard *et al.*, 2010), showed that such approaches tend to support properly the RITEL–QR system (Rosset *et al.*, 2008) in this task.

MOTS-CLÉS : pages web, TextTiling, sélection de documents, questions-réponses, Quaero, Ritel, segmentation textuelle, segmentation thématique.

KEYWORDS: web pages, TextTiling, document selection, question answering, Quaero, Ritel, textual segmentation, topic segmentation.

1 Introduction

C'est un truisme de nos jours de dire qu'Internet est une mine d'information, de ressources et de savoirs qui peuvent sembler infinis. Ces informations sont utiles à toute personne souhaitant s'informer ou se distraire, mais également aux chercheurs de nombreux domaines (biologie, sciences sociales, informatique, ...) pour qui Internet est devenu un objet de recherches. Cependant, malgré les progrès liés, par exemple par le passage au WEB 2.0, on ne peut que constater que les informations sur le Web ne sont pas fiables, ni même accessibles aisément.

Les systèmes de réponses aux questions sont un moyen efficace de rendre cette information à la fois plus accessible et plus fiable. Plus accessible, car ces systèmes répondent de façon précise, rapide et concise aux questions qui leur sont posées en langue naturelle (à l'instar des moteurs de recherche usuels [1]). Plus fiable, car les réponses sont validées par le système (Peñas *et al.*, 2007).

Une étape primordiale (dans toutes les acceptions du terme) pour les systèmes de questions-réponses (QR) est l'opération qui consiste à extraire le contenu textuel des pages. Pour cela, il est nécessaire (Grau, 2004) de nettoyer, restructurer et filtrer leur contenu (par exemple de corriger les balises HTML et les erreurs d'encodage, d'éliminer les codes javascript résiduels et les spams).

La qualité (au sens de leur adéquation à la tâche QR) des textes obtenus dépend fortement de l'extracteur employé (Baroni *et al.*, 2008) et de la qualité intrinsèque de l'information contenue dans les documents à l'origine. Une tâche cruciale, mais souvent mésestimée, pour un système QR est de pouvoir filtrer les documents dont la qualité intrinsèque est faible, afin d'augmenter la précision de la sélection des meilleurs candidats lors de l'extraction de réponses.

Au cours de travaux précédents (Foucault *et al.*, 2011), nous avons mis en place une stratégie de sélection des documents pertinents pour un système QR en français, en complément de la sélection de documents traditionnelle effectuée par le moteur de recherche du système. Cette sélection repose sur une mesure de la qualité intrinsèque des documents en utilisant un modèle de langue, qui nous fournit *a priori* des mesures objectives sur le degré d'informativité d'un texte. Cette stratégie permet d'écarter de la liste des candidats sélectionnés par le moteur de recherche du système, les documents les plus bruités (c'est-à-dire de faible qualité) pour la tâche QR. Ici, un texte est considéré comme pertinent ou non dans sa globalité.

Il est de coutume en QR (Ligozat, 2006) de découper les documents en passage soit au moment de leur indexation, soit au cours des recherches. L'idée est de réduire la variabilité naturelle des documents en taille et en contenu. En effet, avec des segments textuels plus petits, on peut espérer une variabilité plus faible, et un contenu informationnel (corrélé au contenu linguistique, en particulier lexical et sémantique) plus cohérent, ce qui en retour doit permettre l'extraction de réponses plus pertinentes que celles issues de la globalité du texte. Cette stratégie a fait ses preuves par le passé et des travaux récents autour du découpage de textes en passages (Tiedemann, 2007; Khalid et Verberne, 2008) l'ont consolidée.

A notre connaissance, personne n'a tenté de segmenter les documents préalablement à leur indexation, tout en découpant les segments obtenus en passages au moment des recherches dans le but de réduire plus fortement la variabilité des documents. Dans cet article, nous détaillons plusieurs expériences visant à mesurer l'impact d'une telle pré-segmentation sur la tâche QR.

1. e.g. Google `http://www.google.fr`

2 Travaux connexes

L'idée de segmenter des documents textuels (article de journaux, livres, ...) en blocs de texte est un axe de recherche activement exploré dans les années 90 en Recherche d'Information (RI) textuelle. Pour réduire la variabilité linguistique d'un texte, une première idée, explorée notamment par Salton (Salton *et al.*, 1996) et Hearst (Hearst, 1997) consiste à opérer une segmentation en blocs thématiques, les frontières de blocs étant les endroits où on détecte un changement de thème. Des calculs de proximité lexicale entre blocs adjacents permettent de réorganiser le texte d'origine en segments plus homogènes (mais toujours de taille variable). Chez Salton, la proximité lexicale est obtenue à l'aide de mesures de distances vectorielles, chaque bloc étant représenté par un vecteur lexical. En fonction de valeurs seuils sur ces distances, des fusions entre paragraphes sont opérées. (Salton *et al.*, 1996) effectue une fusion itérative, chaque itération fusionnant les paragraphes jugés similaires selon cette distance, le document étant exploré du début à la fin, de gauche à droite. L'itération s'arrête lorsque le texte ne contient plus que des blocs thématiquement homogènes. A partir de cette segmentation, Salton dérive un graphe des relations thématiques qu'entretiennent les blocs au sein du document. Dans le même esprit, (Hearst, 1997) fusionne des blocs de textes entre eux, mais de manière plus fine. Elle se fonde sur une analyse plus linguistique du texte que Salton. En effet, l'algorithme de segmentation de Hearst (*TextTiling*) utilise la structure du discours (ici la théorie des chaînes lexicales) et se fonde sur une segmentation en unités lexicales élémentaires (*tokens*). Ces tokens forment les unités de base pour la représentation de chaque bloc textuel. Cet algorithme utilise une mesure de distance fondée sur les statistiques de co-occurence. TextTiling ne fonctionne pas sur les paragraphes d'origine du texte contrairement à l'algorithme de Salton, mais sur des blocs de pseudo-phrases construits sur la base de ces paragraphes.

Plus récemment, des travaux dans le contexte de la RI dans des documents multimédia ont proposé une nouvelle méthode d'indexation de pages web (Faessel, 2008). Celle-ci s'appuie sur l'information des représentations DOM [2] et CSS des pages web pour segmenter ces dernières avant indexation ; (Bruno *et al.*, 2009) démontrent que l'utilisation de ces informations conduit à une augmentation des performances d'un moteur de recherche dans sa tâche. D'autres travaux de segmentation pour la classification automatique de pages web en thème (Qi et Davison, 2009) utilisent la représentation DOM. Dans (Gupta *et al.*, 2003) l'extraction du contenu textuel des pages se fait automatiquement à l'aide de la structure des arbres DOM. Dans (Asirvatham *et al.*, 2001), l'échantillonnage des couleurs des images est utilisé pour catégoriser les pages qui les contiennent. Dans (Kovacevic1 *et al.*, 2004), pour la même tâche, on utilise le rendu visuel. (Guo *et al.*, 2007) utilise des indices visuels (le rendu des pages fourni par le moteur de Mozilla [3]), géométriques (les coordonnées des éléments de l'arbre DOM au sein du rendu des pages) et le style des pages (la répétition d'information) pour définir les blocs d'information pertinents trouvés dans les pages et les annoter sémantiquement. Dans (Feng *et al.*, 2005), les auteurs étudient l'impact d'indices visuels et structurels sur la segmentation en blocs au travers d'une tâche de catégorisation fonctionnelle (blocs de type menu, titre, contenu, etc.). Dans (Vadrevu *et al.*, 2005), les auteurs utilisent des critères de découpage fondés sur l'homogénéité locale du contenu informationnel des pages web (i.e. modèle de segmentation basé sur le concept de *path entropy*) et d'indices visuels dérivés de leur représentation DOM. Certains systèmes comme VIPS (Cai *et al.*, 2003) se fondent essentiellement sur ce type d'indices pour segmenter les pages.

2. Document Object Model (DOM) `www.w3.org/DOM`
3. `www.mozilla.org`

A notre connaissance, aucune tentative d'application de ces techniques comme procédure de segmentation de pages web n'a été faite en QR. Nous avons choisi dans un premier temps d'utiliser un algorithme de première génération : le TextTiling de Hearst. Ce dernier a montré son intérêt pour la sélection de document pertinent en RI et se fonde sur une philosophie sous-jacente commune à notre domaine en TAL (Traitement Automatique des langues). De plus, on en trouve des implémentations en libre accès (contrairement à certains des algorithmes évoqués plus haut). Par ailleurs, TextTiling présente l'avantage de fournir une segmentation en blocs thématiques qui pourrait être mise à contribution par la suite pour renforcer l'analyse sémantique des documents par un système QR. Dans la perspective de travaux futurs en segmentation de pages web autour de leur représentation visuelle en QR, le TextTiling nous permettra de bénéficier d'une segmentation TAL de référence à comparer à des approches de RI non textuelles. C'est dans cette optique que le travail présenté dans cet article se positionne.

Dans la section 3, nous présentons notre méthode de segmentation textuelle développée sur la base du TextTiling de Hearst ; dans la section 4 nous évaluons cette méthode sur la tâche Questions-Réponses. Nous concluons et présentons les perspectives de ce travail dans la section 5.

3 Segmentation textuelle de pages web

Dans cette section, nous présentons la méthode de segmentation de pages web que nous avons mise en place pour la sélection de documents pertinents en QR.

La figure 1 présente les étapes-clés de la chaîne de traitement qui correspond à cette métode : de l'extraction du contenu textuel des pages à l'obtention de blocs de textes normalisés. Chaque étape clé de cette chaîne est décrite successivement dans les sections 3.2, 3.3 et 3.4.

3.1 Présentation générale

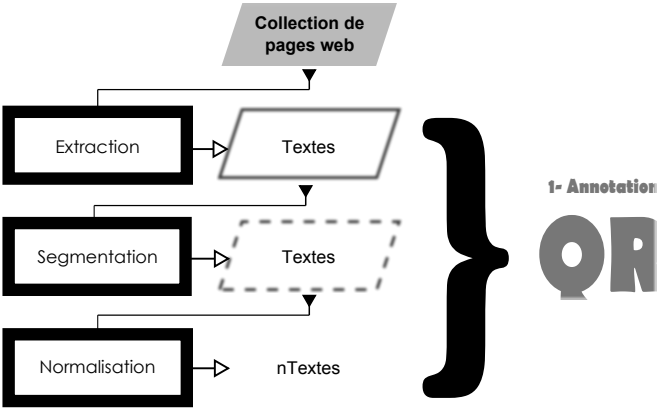

FIGURE 1 – Notre procédure de segmentation de pages web en lien avec les (pré-)traitements QR.

En théorie, nous aurions dû inverser les étapes de normalisation et de segmentation afin que la segmentation bénéficie des traitements de normalisation (voir section 3.4). Cependant, une telle inversion nécessite certaines modifications de notre chaîne de normalisation : en effet, cette dernière supprime l'indentation des textes utile à l'algorithme de TextTiling (voir section 3.3.1). Nous n'avons malheureusement pas eu le temps de mettre en place les modifications adéquates.

3.2 Extraction

Notre procédure d'extraction se déroule en deux temps : **pré-traitement** (section 3.2.1) puis **représentation** et **extraction** du contenu textuel des pages web (section 3.2.2). La phase de pré-traitement des pages web est prise en charge par *Kitten* (Falco *et al.*, 2012), un outil de traitement de documents web développé au LIMSI. La représentation et l'extraction du contenu textuel des pages web se fait sur les versions des pages pré-traitées par Kitten à l'aide du navigateur textuel de pages web *Lynx* [4].

3.2.1 Pré-traitement des pages web

Le pré-traitement des documents web est réalisé à l'aide de Kitten. Ce choix est motivé par les performances état de l'art que ce dernier a obtenu en qualité d'extracteur textuel (Falco *et al.*, 2012) dans le cadre d'évaluations QR sur le système *Fidji* (Moriceau et Tannier, 2010).

Kitten est un outil développé au LIMSI, dédié aux traitements et à la normalisation de données Html. Les pages web fournies en entrée sont traitées et de nouvelles pages web au format Xhtml valide W3C (encodées en UTF8) sont produites en sortie. Ces pages sont bien formées (correction de leur squelette Html via *jTidy* [5]), sans erreurs d'encodage (correction de leur encodage via *jChardet* [6] et conversion des caractères Html spéciaux dans une base Unicode via *HTMLCleaner* [7]).

Kitten produit des pages web exploitables en Extraction d'Information (EI) (Baroni *et al.*, 2008) sans appliquer d'heuristiques de nettoyage prédéfinies contrairement à des outils classiques de nettoyage de contenu comme *Boilerpipe* (Kohlschütter *et al.*, 2010) ou *Ncleaner* (Evert, 2008) ; par exemple Ncleaner préserve pour l'essentiel le texte des balises `<title>`, `<h1>`, `<h2>`, `<h3>`, `<div>` et `<p>` contenus dans le corps des pages, toute autre balise étant jugée non pertinente pour l'extraction. Par ailleurs, Kitten dispose de nombreuses fonctions et filtres configurables qui le rendent flexible. Ainsi, il est possible de conserver le contenu des attributs `<title>` associé à un lien tout en supprimant le lien ou au contraire conserver ce lien tout en supprimant les attributs `<title>` qui lui sont associés. Kitten se rapproche donc plutôt d'outils de développement populaires dans le domaine de l'EI web comme la librairie Python *Beautiful Soup* [8] et le framework de crawling web *Scrapy* [9].

Kitten dispose de son propre module d'extraction de pages web et d'un système d'extraction back-off basé sur Lynx. Celui-ci sert d'extracteur principal dans notre système de segmentation.

4. `lynx.browser.org`
5. `http://jtidy.sourceforge.net`
6. portage Java de la détection automatique d'encodage d'une page du moteur de Mozilla
7. `http://htmlcleaner.sourceforge.net`
8. `http://www.crummy.com/software/BeautifulSoup`
9. `scrapy.org`

3.2.2 Représentation des pages web et extraction textuelle

La représentation des pages utilisée par notre moteur d'extraction se fait grâce à Lynx. Ce dernier est un navigateur d'informations distribuées à portée générale pour Internet. Il permet de naviguer sur le Web depuis une console, en mode textuel uniquement. C'est un outil libre intégré automatiquement dans la plupart des distributions Linux grand public comme Ubuntu, qui intègre de nombreuses fonctionnalités web, dont l'extraction du contenu textuel de pages web.

Nous avons retenu Lynx pour deux raisons. La première raison est qu'il fournit une extraction textuelle de pages web qui reflète leur rendu visuel. La seconde raison est que Lynx peut fournir une décomposition linéaire du contenu des pages web en blocs de texte, adaptée à la plupart des traitements d'analyses de documents en QR.

Si Lynx produit des extractions textuelles fidèles au rendu visuel des pages web, l'agencement des blocs d'extraction diffère de celui observé dans un navigateur web classique du type *Firefox* [10]. En effet, Lynx effectue une traversée gauche-droite descendante des pages web. En conséquence, les blocs d'information textuelle rencontrés le long du parcours sont mis bout à bout dans le fichier d'extraction résultant. Ainsi, on trouve souvent dans les extractions textuelles de Lynx la suite de blocs suivant (donnés ici selon leur contenu visuel) : bandeau, menus, colonne gauche, bloc de contenu principal, colonne droite, puis pied de page. On peut aussi trouver des agencements moins stéréotypiques selon le design des pages et trouver des séries de blocs de contenu principal qui s'enchaînent. L'étape d'extraction textuelle est réalisée par Lynx via le système d'extraction back-off de Kitten.

3.3 Stratégie de segmentation

Nous avons utilisé deux stratégies de segmentation en blocs de texte. La première stratégie consiste à segmenter les textes extraits par Lynx en blocs thématiques de taille variable par l'algorithme de TextTiling de Hearst (Hearst, 1997). La seconde stratégie vise à contrôler la précédente et segmente les textes extraits par Lynx de façon uniforme en blocs de taille identique.

3.3.1 Segmentation par TextTiling

L'algorithme de TextTiling de Hearst (Hearst, 1997) segmente un texte en unités appelées *multi-paragraphes* en fonction des thématiques abordées dans le texte. Traditionnellement, il est utilisé pour détecter les thématiques dans des textes fortement structurés (e.g. articles de journaux, textes issus de livres ...) et de grande taille (i.e. de plusieurs pages). Une des questions sous-jacente aux expériences que nous avons menées était de savoir si cet algorithme pourrait être utile pour la segmentation de pages web.

L'algorithme, présenté en détail dans (Hearst, 1997) s'articule autour des 3 étapes suivantes :

– **tokenisation** ;
– **calcul de scores lexicaux** ;
– **identification de frontières**.

10. `http://www.mozilla.org`

La procédure de segmentation démarre par une étape de **tokenization** du texte qui lui est fourni en entrée. Les mots qui sont des *stopwords* ne sont pas tokenisés et sont écartés. Les autres subissent une étape de stemming basée sur une fonction d'*analyse morphologique*. Le texte est tokenizé en pseudo-phrases de longueur prédéfinie censée représenter la longueur moyenne d'un paragraphe (20 pseudo-phrases par défaut). Les paragraphes d'origine du texte servent de point d'ancrage pour la tokenization, qui elle-même dépend de l'indentation dans le texte.

L'algorithme évalue ensuite la **proximité lexicale** qui existe entre toutes les paires de blocs adjacents possibles, et fournit un score fondé sur des co-occurences lexicales de tokens qui mesure l'écart entre deux blocs. Les blocs sont constitués des pseudo-phrases obtenues lors de la phase de tokenization. La détermination des scores lexicaux varie selon la stratégie utilisée. TextTiling dispose de 2 stratégies de comparaison de blocs différentes. La première (*block comparison*), compare deux blocs adjacents de texte et calcule leur écart sur la base du nombre de tokens qu'ils ont en commun. La seconde méthode (*vocabulary introduction*) évalue ce même écart sur la base des tokens issus des pseudo-phrases qui bordent la frontière entre deux blocs.

Enfin, l'algorithme procède au **marquage des frontières** de blocs pertinentes sur la base des écarts mesurés à l'étape précédente. Ceci est fait à l'aide d'une fenêtre glissante sur les blocs. Les frontières de blocs présentant les plus forts écarts sont sélectionnées comme frontières thématiques.

L'implémentation que nous avons utilisée du TextTiling est fournie par le package Python NLTK sans la fonction d'*analyse morphologique*. Le calcul des scores lexicaux se fait par *block comparison*.

3.3.2 Segmentation uniforme

Cette segmentation représente la condition contrôle dans nos expériences. Elle se contente de segmenter chaque fichier texte qui lui est présenté en 8 blocs, c'est-à-dire la moyenne du nombre de blocs de segmentation obtenus par TextTiling sur notre corpus d'expérimentation au cours de tests préliminaires ; ceci revient à fixer la taille moyenne des blocs en nombre de lignes (voir la section 4.3).

La segmentation se fait selon un parcours linéaire du texte d'entrée, du début jusqu'à la fin, les points de coupe sont déterminés à l'avance selon le nombre total de lignes dans le texte et le nombre maximum de blocs fixé en sortie (8). Les textes trop petits (ceux de moins de 8 lignes) ne sont pas segmentés et sont considérés comme des blocs uniques.

3.4 Normalisation

La normalisation est une étape durant laquelle un texte *brut* est traité afin qu'une unité lexicale soit explicitement définie. Au cours de la normalisation, le texte est transformé dans une forme où les mots et les nombres sont clairement délimités, la ponctuation est séparée des mots, et des phrases ou pseudo-phrases sont clairement formées.

Notre normalisation passe par plusieurs étapes : séparation des mots et nombres de la ponctuation, reconstruction de la casse sur les mots, ajout de la ponctuation le cas échéant et séparation en phrases ou pseudo-phrases du texte d'entrée. Elle s'appuie sur des lexiques, des dictionnaires de règles et des modèles de langue (Déchelotte *et al.*, 2007).

4 Evaluation Questions-Réponses

4.1 Hypothèses de travail et conditions expérimentales

Les expériences présentées ont pour but d'examiner l'hypothèse selon laquelle une fenêtre d'analyse plus réduite pour traiter les documents web permettrait une sélection du système QR plus précise (c'est-à-dire obtenir des réponses plus pertinentes et en plus grand nombre). À cette fin, nous réalisons une segmentation avant l'indexation des documents en plus du découpage habituel en passages réalisé lors de l'extraction des réponses. La segmentation des documents est effectuée par TextTiling ou uniformément par notre algorithme de segmentation contrôle.

Les évaluations de l'impact de ces algorithmes de segmentation sur le système RITEL–QR (voir section 4.2) sont présentées section 4.5 selon 3 conditions expérimentales :

– **condition 1** : condition sans segmentation ou baseline (**bsln**) ;
– **condition 2** : condition en segmentation par TextTiling (**TT**) ;
– **condition 3** : condition en segmentation contrôle (**ctrl**).

4.2 Système d'expérimentation : RITEL–QR

Le système RITEL–QR que nous utilisons dans les expériences est complètement décrit dans (Bernard *et al.*, 2009) et (Galibert, 2009). Il s'agit d'un système qui a été conçu à l'origine comme un système de dialogue (Toney *et al.*, 2008). D'un point de vue général, on peut dire que le système s'appuie sur une analyse multi-niveaux, appliquée sur les questions et sur les documents. Les documents sont totalement analysés et indexés d'après les résultats d'analyse. La recherche est effectuée dans l'index complet des documents. L'analyse permet de repérer et typer des éléments pertinents d'information qui peuvent prendre la forme d'entités nommées, complexes et structurées, de chunks morpho-syntaxiques, d'actes de dialogue et de marqueurs thématiques.

La première étape de RITEL–QR consiste à créer un *descripteur de recherche* (DDR) qui contient toutes les informations utiles pour la recherche de documents, l'extraction de passages pertinents et l'extraction de réponses. Ces informations sont les éléments de la question, leurs transformations possibles (dérivations morphologiques, synonymes etc. et les poids associés), et les types attendus de la réponse (avec les poids associés). Ces types sont le plus souvent des types d'entités nommées (*personne*, *lieu* ...) et reflètent la taxonomie d'entités utilisée au moment de l'analyse.

La sélection des documents consiste à fournir, à partir de l'index, les n documents les plus pertinents, c'est-à-dire ceux contenant le plus d'informations présentes dans le DDR. En fonction de ces informations et de leur densité, les documents obtiennent un score. Ensuite, des passages sont extraits de chaque document. Ces passages sont de tailles variables (une fenêtre d'analyse différente est appliquée selon la catégorie de la question) et sont scorés selon le même principe que les documents. L'extraction et l'évaluation des candidats réponses s'appuient sur la redondance de ces derniers dans les documents et les passages. On considère que les éléments de passages qui correspondent à un type possible de réponse du DDR et qui ne sont ni des élements ni des sous-éléments définis dans le DDR, sont des candidats réponses potentiels. A chacun d'eux est finalement attribué à un score de pertinence (Bernard *et al.*, 2009).

4.3 Corpus

Le corpus de pages web utilisé dans nos expérimentations (ci-après *Q07fr*) est composé de 499 734 pages web (5Gbytes) tout venant (i.e. journal, Wikipédia, blog, site de vente, forum, etc.) et en français. Il nous est fourni par le projet Quaero [11] et sert de corpus standard dans le cadre des évaluations QR au sein du projet (Quintard *et al.*, 2010). Les questions de test et d'entraînement utilisées (309 et 722 questions) proviennent du même projet. Elles ont été créées à partir de logs utilisateurs (Quintard *et al.*, 2010) du moteur de recherche français Exalead [12] et sont composées de questions *factuelles* (e.g. *Qui est Gandhi ?*, *Combien pèse la tour Eiffel ?*, *Où se situe Pondichéry ?* et *Que signifie CSDPTT ?*).

(a)

Etape/Cond	**bsln**	**ctrl**	**TT**
extraction	497 228	497 228	497 228
segmentation	-	3 686 749	3 857 585
normalisation	485 037	3 660 264	3 686 875
annotation	485 037	3 660 264	3 686 875
indexation	484 060	3 658 988	3 686 857
durée totale	1,1j (26,5h)	2,38j (57,3h)	5,3j (127,5h)

(b)

Stat/Index	**nbB**	**nbL**	**nbB**	**nbL**	**nbB**	**nbL**
Min	1	1	1	1	1	1
Max	1	461 075	8	1 262	186	9 277
Sd	0	7 646,5	0,01	18,7	8,4	32,61
Mean	1	295,4	8	19,4	7,3	20,9

TABLE 1 – (a) Nombre de blocs par condition expérimentale (**Cond**) selon leur type : segmenté (**ctrl** et **TT**) ou non (**bsln**), selon les étapes nécessaires à les créer (**Etape**) et la durée totale de traitement correspondant. (b) Statistiques (**Stat**) du nombre de blocs (**nbB**) et de lignes (**nbL**) moyens (**Mean**), minimum (**Min**), maximum (**Max**) et déviation standard (**Std**) des index (**Index**) relatifs à chaque condition expérimentale.

Le tableau 1 (a) présente les résultats des traitements (en terme de nombre de fichiers traités) de chacune des étapes de notre chaîne de segmentation, ainsi que des étapes de pré-traitements QR (annotation et indexation), pour chacune des conditions d'expérimentations (**Cond**) à partir de *Q07fr*. Ces résultats suivent le schéma de la figure 1. Chaque sortie d'une étape dépend du résultat qui précède pour une condition donnée. On peut noter que le nombre de fichiers issus de la segmentation dans les conditions contrôle (**ctrl**) et TextTiling (**TT**) sont proches (environ 20K blocs de différence). Les blocs indexés dans ces 2 conditions correspondent aux 484 060 textes indexés en condition baseline (**bsln**). La durée des traitements (parallèles/mêmes serveurs) dans ces conditions est respectivement de 2 à 5 fois plus longue qu'en condition sans segmentation.

Le tableau 1 (b) présente le nombre moyen de blocs (**nbB**) et de lignes par bloc (**nbL**) obtenus en conditions contrôle et TextTiling. Ces informations sont aussi données pour la condition baseline à titre indicatif (un bloc par fichier). On constate que l'algorithme de TextTiling et le contrôle se comportent de façon très similaire : en moyenne, le nombre de blocs produits (**ctrl** : 8 et **TT** : 7,3) ainsi que leur taille (**ctrl** : 19,4 et **TT** : 20,9) sont semblables. Le TextTiling produit une légère sur-segmentation : la déviation standard est 2 fois plus élevée que celle du contrôle (**ctrl**) en nombre de lignes, le maximum de blocs pour un même document étant également plus grand.

11. http://www.quaero.org
12. http://www.exalead.com

4.4 Métriques d'évaluation

Dans ce travail, nous employons les métriques habituellement utilisées en QR :

– la *précision* définie équation (1), est le ratio entre le nombre de réponses correctes et le nombre total de questions. Si le système est capable de fournir plusieurs réponses par question, on ne considère que la première. CR_i est le rang de la première réponse correcte pour la question i. CR_i prend pour valeur $+\infty$ si aucune réponse correcte n'a été trouvée.
– Le *top-n* défini équation (2), mesure la *précision* selon les réponses correctes de rang 1 à n.
– Le *Mean Reciprocal Rank* (Moyenne des Réciproques des Rangs ou MRR) défini équation (3), permet de mesurer la qualité du classement des réponses (10 par question) effectué par le système. La réponse correcte la mieux classée est pondérée par l'inverse de son rang initial. Une absence de réponse correcte entraîne une contribution nulle. Le score final correspond à la moyenne des contributions.

$$\text{précision} = \frac{\#\text{CR}_i = 1}{\#\text{questions}} \quad (1) \qquad \text{top-}n = \frac{\#\text{CR}_i \leq n}{\#\text{questions}} \quad (2) \qquad \text{MRR} = \frac{\sum \frac{1}{\text{CR}_i}}{\#\text{questions}} \quad (3)$$

4.5 Résultats

Les résultats sont présentés dans les parties (a) et (b) du tableau 2. On a utilisé le test de McNemar (McNemar, 1947; Agresti, 1990) de R [13] pour juger de la significativité des résultats présentés tableau 2 (a). Les résultats du test sont donnés tableau 3 [14].

On constate, tableau 2 (a), que les deux conditions de segmentation testées sont proches de la condition baseline suggérant ainsi que la segmentation n'apporte pas de réels bénéfices à notre système QR. Les performances du système sont très proches en terme de **précision** (0,6 point de différence au plus entre **bsln** et **TT**). Mais le **MRR** présente un écart plus important entre les conditions (2 points entre les conditions **bsln** et **ctrl**, et 1 point entre les conditions **bsln** et **TT**). La segmentation **ctrl** semble donc permettre au système de trouver de meilleures réponses (i.e. des réponses plus précises) qu'en condition baseline ou TextTiling. Toutefois, d'après les tests statistiques des performances QR présentés tableau 3, ceci n'est qu'une tendance.

Le test de McNemar (McNemar, 1947), que nous avons utilisé dans nos expériences, établit la significativité des résultats observés entre 2 conditions A et B et une mesure M donnée, selon des variations observées entre A et B, synthétisées dans une table de contingence 2x2. De là, le test (bilatéral) estime une valeur Q (i.e. khi² de McNemar) pour un degré de liberté df donné et dérive une valeur p. Si p est inférieure (ou égale) au seuil critique α, l'hypothèse nulle H_0 est rejetée et la différence observée entre A et B est jugée significative. Dans notre cas, une table de contingence comptabilise le total de questions (#q) pour lesquelles RITEL-QR trouve une réponse de même exactitude en conditions A et B. Il y a 4 types de compte, nombre total de questions avec une réponse : correcte (r) selon A et selon B (#rr), fausse (w, xs ou xl) selon A et selon B (#WW), correcte selon A et fausse selon B (#rW) et inversement (#Wr).

13. `http://www.r-project.org`
14. Ces derniers sont les mêmes sur le top-10 et le MRR, puisque ce test ne distingue pas les réponses selon leur rang.

Ainsi, on peut constater que l'hypothèse H_0 selon laquelle la différence observée entre les conditions **bsln** et **ctrl** n'est pas significative pour les performences QR en top-10, est à peine rejetée : la valeur de p obtenue dans ces conditions n'étant pas inférieure mais tout juste alignée sur le seuil critique de significativité α [15].

L'étude du nombre total de bonnes réponses fournies par le système selon leur position au sein du **top-10** tableau 2 (b) (**bsln** : 178, **TT** : 183 et **ctrl** : 190) confirme cette tendance. On voit aussi dans ce tableau que les réponses apportés par le système en condition **ctrl** (jusqu'à 12 réponses suppplémentaires, soit 3,9% de réponses en plus) se trouvent dans le top-3, là où la segmentation par TextTiling a tendance à apporter de nouvelles réponses à des rangs inférieurs.

Nous avons pu constaté que la segmentation des documents accélérait les (pré-)traitements QR.

Cond	**P**	**MRR**	**top-10**	**#q**
bsln	31.4	39.6	57.6	309
ctrl	31.7	41.6	61.5	309
TT	32.0	40.5	59.2	309
Cond	**#r**	**#xs**	**#xl**	**#w**
bsln	97	6	8	198
ctrl	98	11	5	195
TT	99	11	2	197

(a)

	Cond		
rang	**bsln**	**ctrl**	**TT**
1	97	98	99
2	26	32	26
3	17	26	19
4	9	7	7
5	8	7	11
6	6	4	6
7	7	5	3
8	6	5	4
9	2	2	8
10	0	4	0
Total	178	190	183

(b)

TABLE 2 – (a) Résultats QR globaux par condition expérimentale (**Cond**). **P** : précision ; **MRR** : rang moyen réciproque ; **top-10** : précision sur 10 rangs. **#q** : nombre total de questions évaluées. #r, #xs, #xl et #w : nombre total de réponses *justes*, *trop courtes*, *trop longues* et *fausses*, selon le top-1. (b) Focus sur les résultats du top-10 présentés en (a), selon chaque position (**rang**) dans le classement (réponses justes uniquement).

df=1,α=.05		**ctrl** (A) / **bsln** (B)					**bsln** (A) / **TT** (B)					**ctrl** (A) / **TT** (B)				
mesure	#q	**r**	**W**	Q	p	H_0	**r**	**W**	Q	p	H_0	**r**	**W**	Q	p	H_0
P	**r**	77	21	0	1	×	80	17	.02	.86	×	80	18	0	1	×
	W	20	191				19	193				19	192			
top-10	**r**	167	23	3.55	.05	✓	164	14	.48	.48	×	174	16	1.44	.23	×
(MRR)	**W**	11	108				19	112				9	110			

TABLE 3 – Résultats de significativité du test (bilatéral) de McNeamr pour les résultats QR du tableau 2 (a). Q : khi² de McNemar. df : degré de liberté. p : p-valeur. α : seuil critique. H_0 : hypothèse nulle (rejet : ✓). #q : nombre total de questions pour lesquelles on a une réponse de même nature ou non entre 2 conditions A et B. **r** : réponse *juste*. **W** : réponse *fausse*. Les rectangles **rW/rW** représentent des tables de contingence.

15. p=0,059 si on augmente la précision du test de McNemar fournie tableau 3

5 Conclusion et perspectives

Au cours de travaux précédents (Foucault *et al.*, 2011) nous avons mis en place une stratégie de sélection de documents pertinents pour un système QR sur le français. Elle s'appuie sur un modèle de langue qui fournit *a priori* une mesure objective du degré d'informativité d'un texte. Cette mesure de la qualité intrinsèque des documents sert à filtrer les documents non pertinents pour la tâche QR. L'effet d'un tel filtrage appliqué à l'échelle globale des documents, c'est avéré assez limité. La variabilité naturelle des pages web en taille et en contenu (comme leur caractère multi-thématique) pénalise vraisemblablement le système dans sa tâche. Nous avons donc cherché à développer un système de segmentation qui permette d'appliquer ce filtrage à une échelle non plus globale mais locale, sur des sous-parties de document. Le travail présenté dans cet article avait pour objectif de mettre un tel système de segmentation en place.

La question à laquelle nous avons voulu répondre dans cette article est la suivante : segmenter les documents avant l'indexation, en plus du découpage habituel des documents en passages lors de l'extraction des réponses, améliore-t-il les performances d'un système de questions-réponses ?

Pour répondre à cette question, nous avons testé deux types de pré-segmentation supportée par une extraction de contenu textuel de pages web maison. L'une segmente les textes extraits via un algorithme de texttiling classique (TextTiling) en blocs thématiques de taille variable. L'autre les segmente uniformément en blocs de taille fixe, sans découpage thématique.

Les résultats obtenus ne nous permettent pas de trancher nettement en faveur de l'une ou l'autre de ces approches de segmentation. Cependant, les tendances observées suggèrent qu'une pré-segmentation des pages web comme nous l'avons définie peut servir un système QR ; segmenter les documents avant l'indexation afin de renforcer l'effet du découpage de ces derniers en passages lors de l'extraction des réponses, améliore la précision du système en terme de top-10 sans pour autant diminuer cette dernière en terme de top-1. Cette tendance est plus marquée pour la segmentation uniforme de pages web que pour une segmentation plus « intelligente » à l'aide de l'algorithme de TextTiling (sans *analyse morphologique*, le calcul des scores lexicaux se faisant par *block comparison*). Ce constat est contradictoire avec d'autres travaux, mais confirme certaines conclusions apportées par Hearst dans ses travaux de segmentation thématique de textes en Recherche d'Information (Hearst, 1997). Il serait intéressant de déterminer les raisons amenant à ce constat. Si la nature des documents (page web versus texte), est l'une des raisons qui pourrait l'expliquer, qu'en est-il par exemple de la longueur des documents et de la version du TextTiling que nous avons utilisé dans nos expériences ?

En perspective des travaux présentés dans cet article, nous projetons d'abord d'étudier l'impact d'une pré-segmentation uniforme des pages web sur notre stratégie de sélection de documents pertinents développée dans (Foucault *et al.*, 2011). Concernant nos travaux de segmentation de pages web en QR à partir de la représentation visuelle des pages, nous comptons évaluer la pertinence de la procédure d'extraction mise en place au sein du système de segmentation de pages web présenté dans cet article.

Remerciements

Ce travail a été financé partiellement par l'OSEO, dans le contexte du programme Quaero.

Références

AGRESTI, A. (1990). *Categorical data analysis*. New York : Wiley, London.

ASIRVATHAM, A. P., RAVI, K. K., PRAKASH, A., KRANTHI, A. et RAVI, K. (2001). Web page classification based on document structure.

BARONI, M., CHANTREE, F., KILGARRIFF, A. et SHAROFF, S. (2008). Cleaneval : a competition for cleaning web pages. *In LREC*. European Language Resources Association.

BERNARD, G., ROSSET, S., GALIBERT, O., BILINSKI, E. et ADDA, G. (2009). The limsi participation in the qast 2009 track : experimentating on answer scoring. *In CLEF'09*, Corfu, Grece.

BRUNO, E., FAESSEL, N., GLOTIN, H., MAÎTRE, J. L. et MICHEL., S. (2009). Ir web search based on presentation and multimedia content. *In Actes des 25emes Journees Bases de Donnees Avancees (BDA 2009)*, pages 408–407.

CAI, D., YU, S., WEN, J.-R. et MA, W.-Y. (2003). VIPS : a vision-based page segmentation algorithm. Rapport technique, Microsoft (MSR-TR-2003-79).

DÉCHELOTTE, D., SCHWENK, H., ADDA, G. et GAUVAIN, J.-L. (2007). Improved machine translation of speech-to-text outputs. *In Interspeech'07*, Antwerp. Belgium.

EVERT, S. (2008). A lightweight and efficient tool for cleaning web pages. *In LREC*. European Language Resources Association.

FAESSEL, N. (2008). Indexation de blocs extraits de pages web en utilisant le rendu visuel. *In CORIA*, pages 393–400. Université de Renne 1.

FALCO, M.-H., MORICEAU, V. et VILNAT, A. (2012). Kitten : a tool for normalizing html and extracting its textual content. *In* CHAIR), N. C. C., CHOUKRI, K., DECLERCK, T., DOĞAN, M. U., MAEGAARD, B., MARIANI, J., ODIJK, J. et PIPERIDIS, S., éditeurs : *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).

FENG, J., HAFFNER, P. et GILBERT, M. (2005). A learning approach to discovering web page semantic structures. *In Proceedings of the Eighth International Conference on Document Analysis and Recognition*, ICDAR'05, pages 1055–1059, Washington, DC, USA. IEEE Computer Society.

FOUCAULT, N., ADDA, G. et ROSSET, S. (2011). Language modeling for document selection in question answering. *In RANLP'11*, pages 716–720, Hissar, Bulgaria.

GALIBERT, O. (2009). *Approches et méthodologies pour la réponse automatique à des questions adapatées à un cadre intéractif en domaine ouvert*. Thèse de doctorat, Paris-Sud11, LIMSI/CNRS.

GRAU, B. (2004). Méthodes Avancées pour les Systèmes de Recherche d'Informations. *In Visualisation d'Information et Interaction*, chapitre 10 : Systèmes de question-réponse, pages 189–218. Hermès. Dir. M. Ihadjadene.

GUO, H., MAHMUD, J., BORODIN, Y., STENT, A. et RAMAKRISHNAN, I. (2007). A general approach for partitioning web page content based on geometric and style information. *In Proceedings of the Ninth International Conference on Document Analysis and Recognition - Volume 02*, ICDAR '07, pages 929–933, Washington, DC, USA. IEEE Computer Society.

GUPTA, S., KAISER, G., NEISTADT, D. et GRIMM, P. (2003). Dom-based content extraction of html documents. *In Proceedings of the 12th international conference on World Wide Web*, WWW '03, pages 207–214, New York, NY, USA. ACM.

HEARST, M. A. (1997). Texttiling : Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23:33–64.

KHALID, M. A. et VERBERNE, S. (2008). Passage retrieval for question answering using sliding windows. *In In Proceedings of COLING 2008, Workshop IR4QA*.

KOHLSCHÜTTER, C., FANKHAUSER, P. et NEJDL, W. (2010). Boilerplate detection using shallow text features. *In Proc. of 3rd ACM International Conference on Web Search and Data Mining New York City, NY USA (WSDM 2010)*.

KOVACEVIC1, M., DILIGENTI, M., GORI, M. et MILUTINOVIC1, V. (2004). Visual adjacency multigraphs . a novel approach for a web page classification. *In Proceedings of the Workshop on Statistical Approaches to Web Mining (SAWM)*, pages 38–49.

LIGOZAT, A. L. (2006). *Exploitation et fusion de connaissances locales pour la recherche d'informations precises*. Thèse de doctorat, Paris-Sud11, LIMSI/CNRS.

MCNEMAR, Q. (1947). Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157.

MORICEAU, V. et TANNIER, X. (2010). Fidji : using syntax for validating answers in multiple documents. *Information Retrieval*, 13(5):507–533.

PEÑAS, A., RODRIGO, Á. et VERDEJO, F. (2007). Overview of the answer validation exercise 2007. *In CLEF*, pages 237–248.

QI, X. et DAVISON, B. D. (2009). Web page classification : Features and algorithms. *ACM Computing Surveys*, 41(2):12 :1–12 :31.

QUINTARD, L., GALIBERT, O., ADDA, G., GRAU, B., LAURENT, D., MORICEAU, V., ROSSET, S., TANNIER, X. et VILNAT, A. (2010). Question answering on web data : The QA evaluation in Quæro. *In LREC'10*, Valletta, Malta.

ROSSET, S., GALIBERT, O., BERNARD, G., BILINSKI, E. et ADDA, G. (2008). The limsi participation to the qast track. *In Working Notes of CLEF 2008 Workshop*, Aarhus, Denmark.

SALTON, G., SINGHAL, A., BUCKLEY, C. et MITRA, M. (1996). Automatic text decomposition using text segments and text themes. *In Proceedings of the the seventh ACM conference on Hypertext*, HYPERTEXT '96, pages 53–65, New York, NY, USA. ACM.

TIEDEMANN, J. (2007). Comparing document segmentation strategies for passage retrieval in question answering. *In Proceedings of the Conference on Recent Advances in Natural Language Processing (RANLP'07)*, Borovets, Bulgaria.

TONEY, D., ROSSET, S., MAX, A., GALIBERT, O. et BILINSKI, E. (2008). An Evaluation of Spoken and Textual Interaction in the RITEL Interactive Question Answering System. *In* (ELRA), E. L. R. A., éditeur : *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco.

VADREVU, S., GELGI, F. et DAVULCU, H. (2005). Semantic partitioning of web pages. *In Proceedings of the 6th international conference on Web Information Systems Engineering*, WISE'05, pages 107–118, Berlin, Heidelberg. Springer-Verlag.