
Résumé automatique de texte avec un algorithme d'ordonnement

Nicolas Usunier, Massih-Reza Amini et Patrick Gallinari

Laboratoire d'Informatique de Paris 6
8, rue du Capitaine Scott
75015 Paris
{usunier, amini, gallinari}@poleia.lip6.fr

RÉSUMÉ. Dans cet article, nous proposons une nouvelle approche pour le résumé automatique de textes utilisant un algorithme d'apprentissage numérique spécifique à la tâche d'ordonnement. Les précédentes approches d'apprentissage pour le résumé automatique définissaient un ensemble de caractéristiques permettant d'associer à chaque phrase un vecteur de scores, puis d'entraîner un classifieur afin d'obtenir une combinaison de ces scores. L'objectif est d'extraire les phrases d'un document qui sont les plus représentatives de son contenu. Cependant, des résultats théoriques récents suggèrent que le critère de classification peut être sous-optimal pour apprendre des fonctions de score. Ainsi, nous proposons d'utiliser le cadre offert par les algorithmes d'ordonnement, qui permettent d'apprendre des combinaisons des caractéristiques en se concentrant sur les scores relatifs des phrases d'un même document. Les caractéristiques que nous utilisons sont basées sur l'état de l'art, mais aussi sur une nouvelle approche utilisant des groupements de mots qui co-occurrent dans les mêmes documents. Nous montrons empiriquement que les nouvelles caractéristiques, ainsi que la nouvelle approche d'apprentissage, obtiennent des résultats meilleurs que les approches précédentes sur deux corpus distincts.

ABSTRACT. This paper investigates a new approach for automatic text summarization based on a Machine Learning (ML) ranking algorithm. Previous ML approaches defined a set of features which were used to produce a vector of scores for each sentence in a given document and trained a classifier to make a global combination of these scores. The goal is to extract a subset of a document which most reflects its content. However, recent theoretical results suggest that the classification criterion may be suboptimal for learning scoring functions. Therefore, we propose to use ranking algorithms, which also combine the scores of different features but using a criterion which tends to reduce the relative misordering of sentences within a document. Features we use here are either based on the state-of-the-art or built upon word-clusters. These clusters are groups of words which often co-occur with each other, and can serve to expand a query or to enrich the representation of the sentences of the documents. We empirically show that the features used as well as the ranking algorithms outperforms state-of-the-art approaches on two distinct datasets.

MOTS-CLÉS : résumé automatique de texte, algorithmes d'ordonnement, apprentissage automatique.

KEYWORDS: text summarization, ranking algorithms, machine learning.

1. Introduction

Avec l'accroissement actuel du nombre de documents textuels accessibles en ligne, il est souvent difficile de rechercher l'information demandée parmi un ensemble exhaustif de documents existants. Fournir aux utilisateurs un aperçu rapide du contenu de chaque document permet de mieux s'affranchir de cette tâche de navigation. Les moteurs de recherche actuels fournissent ainsi les premières phrases (ou celles contenant les mots clés d'une requête) pour chaque document retourné. Cependant, un tel *résumé* ne traduit pas exactement l'essence d'un document et il est préférable de concevoir un système capable d'extraire pour chaque document un sous-ensemble reflétant au mieux son contenu. Ceci est l'objet de la tâche du résumé automatique introduite par Luhn à la fin des années cinquante (Luhn 1958).

Cette tâche telle qu'elle a initialement été définie spécifie que la méthode doit être suffisamment rapide pour être appliquée à de nombreux documents, et doit générer des résumés synthétiques. Cependant, les traitements nécessaires pour générer de tels résumés sont trop coûteux pour être appliqués à des grands corpus de documents (Sparck-Jones 1993). Une simplification de la tâche a alors été proposée, qui consiste à extraire d'un document les phrases (éventuellement les paragraphes (Mitra et al. 1997)) les plus représentatifs de son contenu. L'extraction de ces phrases est effectuée grâce à des heuristiques spécifiques, dont sept grandes classes ont été identifiées et énumérées dans (Paice et Jones 1993). Par exemple, une classe d'heuristique consiste à comparer les mots d'un passage avec les mots du titre d'un document, une deuxième est de vérifier si le passage contient des marqueurs linguistiques (ou *cue-words*), comme « en conclusion », « en résumé », etc. Ces heuristiques allouent des scores réels aux passages, et les passages obtenant les meilleurs scores constituent le résumé créé automatiquement.

L'état de l'art consiste à combiner ces heuristiques manuellement (Goldstein et al. 1999) ou avec des méthodes génériques (Kupiec et al. 1995). La solution adoptée par ces dernières est d'effectuer une combinaison numérique des heuristiques de façon automatique grâce à l'apprentissage statistique, en utilisant des bases de documents étiquetés : pour chaque document, les phrases possèdent une étiquette *I* ou *-I* correspondant respectivement au fait qu'elles font partie ou non d'un résumé *de référence* du document. Ces étiquettes sont déterminées soit manuellement soit, pour des cas particuliers et à des fins d'expérimentation, par des méthodes automatiques (Marcu 1995) qui utilisent une information supplémentaire non disponible à l'algorithme d'apprentissage. Dans ces systèmes (Kupiec et al. 1995, Chuang et Yang 2000, Amini et Gallinari 2002), un algorithme de classification est alors utilisé pour déterminer la combinaison des scores des heuristiques qui a pour but d'associer à chaque phrase son étiquette *I* ou *-I*. Pour un nouveau document, le résumé est constitué des *k* phrases de ce document qui, d'après la fonction apprise, sont les plus susceptibles d'avoir pour étiquette *I*, où *k* est un taux de compression et est généralement fixé à 10 ou 20% du nombre de phrases du document.

Bien que nécessitant une base étiquetée pour être utilisable, les méthodes à base d'apprentissage pour le résumé ont deux avantages majeurs par rapport aux combinaisons déterminées manuellement. D'une part, la méthode renvoie une combinaison optimale par rapport au critère d'apprentissage, et, d'autre part, elle est générique du corpus de documents : étant donné un nouveau corpus de documents, il suffit de ré-entraîner le modèle d'apprentissage sur ce corpus pour avoir une combinaison optimale pour le nouveau corpus. Cependant, nous pensons que le critère de classification utilisé jusqu'à présent est sous-optimal pour le résumé automatique. En effet, la tâche est d'apprendre une fonction allouant des scores supérieurs aux phrases pertinentes (i.e. ayant l'étiquette I) d'un document aux phrases non-pertinentes (d'étiquette $-I$) de ce même document. Ainsi, c'est la capacité à bien ordonner les phrases qui doit être optimisée. Or (Cortes et Mohri 2003, Caruana et Alexandru 2004) ont montré théoriquement et empiriquement que les performances en terme d'ordonnement d'un classifieur sont peu corrélées à son erreur d'étiquetage, qui est optimisée par le critère de classification. Intuitivement, cela est dû au fait qu'un classifieur va apprendre une fonction qui tente d'allouer à toutes les phrases étiquetées I un score supérieur à une certaine constante c , et à toutes les autres phrases un score inférieur à c . Ainsi, un classifieur compare les scores des phrases avec une constante absolue au lieu de se focaliser sur les scores relatifs entre les phrases pertinentes et non pertinentes d'un même document.

Dans cet article, nous présentons une nouvelle approche d'apprentissage statistique pour le résumé automatique de texte, basée sur un algorithme d'ordonnement (Cohen et al. 1999). Les algorithmes d'ordonnement permettent d'apprendre un ordre partiel sur les données, et ont par ailleurs été utilisés avec succès sur des tâches d'ordonnement comme la *méta-recherche* (Freund et al. 2003) ou pour réordonner des arbres d'analyse syntaxique (Collins 2002). Adapté à la tâche de résumé, le mode d'apprentissage est le suivant: si p_1 et p_2 sont deux phrases d'un même document, p_1 étant pertinente et p_2 ne l'étant pas, un algorithme d'ordonnement va apprendre une fonction telle que le score alloué à p_1 est supérieur à celui de p_2 . Un tel algorithme se concentre donc sur les scores relatifs des phrases d'un même document, et semble donc très adapté à la tâche de résumé automatique.

Le plan de cet article est le suivant. Dans la section 2, nous décrivons les heuristiques de sélection des phrases (appelées aussi *caractéristiques* des phrases) qui seront ensuite combinées par les algorithmes d'apprentissage. Ces caractéristiques sont des scores de similarité entre la phrase considérée et des requêtes génériques, obtenues en effectuant des enrichissements divers par rapport au titre du document considéré. Les similarités sont calculées en représentant les phrases par un modèle standard sac-de-mots, et en utilisant une autre représentation fondée sur des groupes de mots. Une telle représentation est nouvelle en résumé automatique. Dans la section 3, nous décrivons l'algorithme d'ordonnement utilisé, ainsi que l'algorithme de classification qui servira de modèle de référence

dans notre évaluation. Dans la section 4, nous décrivons les expériences et les résultats obtenus sur deux collections de documents: le corpus SUMMAC¹, et un sous-ensemble du corpus WIPO². L'interprétation des résultats est donnée en section 5, et la section 6 conclue l'article.

2. Caractéristiques pour le résumé automatique

Dans cette section, nous décrivons les heuristiques de sélection des phrases. Chacune de ces heuristiques identifie des caractères particuliers des phrases qui tendent à apparaître dans celles qui doivent être extraites pour former un bon résumé, et, pour une phrase donnée, une heuristique renvoie un score réel, qui est d'autant plus grand que le caractère recherché a été identifié. Chaque phrase est alors décrite par un vecteur de scores fournis par les différentes heuristiques, où la valeur à une dimension donnée est le score renvoyé par l'heuristique correspondant à cette dimension. L'algorithme d'apprentissage apprend une combinaison linéaire de ces scores, ou *caractéristiques* des phrases. L'objectif est alors d'avoir des caractéristiques indépendantes, chacune tenant compte d'un critère particulier de pertinence des phrases, puis de les combiner, afin d'obtenir une combinaison plus performante que la meilleure caractéristique.

(Paice et Jones 1993) regroupent les caractéristiques à prendre en compte pour le résumé automatique en sept catégories: 1) les marqueurs linguistiques (aussi appelés *cue-words*), 2) les acronymes, 3) les mots fréquents d'un document, 4) les mots-clés du titre du document, 5) la position de la phrase dans le document, 6) la longueur de la phrase, et 7) les liens sémantiques entre les phrases. Ces caractéristiques ont été utilisées partiellement ou dans leur totalité dans (Kupiec et al. 1995, Goldstein et al. 1999). Nous utiliserons dans notre travail les marqueurs linguistiques, les acronymes et les mots-clés du titre du document ainsi que d'autres caractéristiques présentées dans la section suivante. Ces caractéristiques donnent un score à chaque phrase d'un document. Dans notre approche, chaque caractéristique est définie par un couple (type de requêtes, fonction de similarité entre une requête et une phrase). Un type de requête génère une requête pour un document d donné selon des règles déterminées. Le score associé à chaque phrase p de d pour la caractéristique considérée est la similarité entre la requête précédemment déterminée et la phrase p . Nos différentes caractéristiques correspondent à des enrichissements de requêtes spécifiques, à la représentation des requêtes et des phrases utilisées pour calculer la similarité, ainsi que différentes mesures de similarités utilisées.

¹ http://www.itl.nist.gov/iaui/894.02/related_projects/tipster_summac/cmp_lg.html

² <http://www.wipo.int/ibis/datasets/index.html>

2.1. Les requêtes issues de l'état de l'art

Tous nos types de requêtes sont issus de deux types initiaux. Le premier type initial génère une requête à partir des mots les plus fréquents de la collection de documents considérée, et est noté **MPF** (*Mots les Plus Fréquents*) dans la suite. Le second type initial, noté **mots-clefs du titre** dans la suite, génère une requête constituée des mots du titre du document considéré.

Il a été montré que pour la tâche de résumé automatique, des enrichissements des requêtes du type **mots-clefs du titre** pouvaient améliorer les performances de façon très significative (Goldstein et al. 1999). En effet, le titre du document, ainsi que les phrases, contiennent peu de mots et sont donc sensibles aux variations linguistiques. Autrement dit, il faut pouvoir détecter dans les phrases d'un document les mots sémantiquement proches de ceux de son titre. Il est commun d'utiliser des techniques d'expansion de requêtes (Goldstein et al. 1999), soit en s'appuyant sur le thesaurus WordNet (Fellbaum 1998), soit des techniques d'enrichissement de requêtes à base d'Analyse du Contexte Local (ACL) (Xu et Croft 1996). Ainsi, nous créons deux nouveaux types de requêtes :

- **mots-clefs du titre + WordNet**, qui ajoute aux mots clefs du titre leurs synonymes trouvés dans WordNet,
- **mots-clefs du titre + ACL**, qui applique la technique d'enrichissement à base d'ACL : les mots qui cooccurrent avec les mots du titre dans les phrases obtenant les scores les plus élevés pour la requête issue de **mots-clefs du titre** et la mesure de similarité Sim_1 (cf. section 2.2) sont ajoutés à la requête générée par **mots-clefs du titre**.

Enfin, pour prendre en compte la majorité des caractéristiques de l'état de l'art, nous créons un troisième type de requêtes, appelé **mots-clefs du titre + MPFD**, qui ajoute aux mots-clefs du titre les mots les plus fréquents du document considéré.

2.2. Les requêtes utilisant les groupements de mots

Nous proposons, en plus des types de requêtes issus de l'état de l'art, de nouveaux types de requêtes constitués utilisant de groupements de mots. Ils permettent de prendre en compte les cooccurrences de mots dans le corpus de document tout entier, contrairement aux techniques locales comme l'ACL qui ne considère que des cooccurrences locales au document considéré.

La première étape à la constitution de ces types de requêtes est la création de groupements de mots, sur le principe suivant : deux mots sont dans le même groupement s'ils ont une forte probabilité de cooccurrence dans les documents. Pour former les groupes, nous reprenons la procédure décrite dans (Caillet et al. 2004), qui consiste à représenter chaque mot w par un vecteur $\langle n(w,d) \rangle_{d \in D}$ où D est la collection de documents et $n(w,d)$ le nombre d'occurrences de w dans le document

d. Le regroupement des mots est ensuite effectué, sur la base de cette représentation, grâce à l'algorithme Naive-Bayes, en maximisant le critère de vraisemblance classifiante (Symons 1983). Ce procédé permet donc de trouver des groupements de mots qui tendent à apparaître dans les mêmes documents. Le nombre de groupes à trouver est un hyperparamètre de l'algorithme. Nous l'avons fixé à la valeur $W/100$, où W est le nombre de mots du vocabulaire, qui permet de bonnes performances empiriques. Il a été montré dans (Cailliet et al. 2004) que les mots appartenant à un même groupe présentent généralement des similarités thématiques, et peuvent donc aider à trouver des phrases pertinentes traitant directement du sujet du titre sans employer exactement les mêmes mots. Les mots des groupes sont aussi différents que ceux trouvés par les techniques utilisant les cooccurrences locales comme le LCA, et fournissent donc une information supplémentaire et indépendante de celle que nous avons déjà.

A partir de ces groupements de mots et de la requête appelée **mots-clefs du titre**, nous créons deux nouveaux types de requêtes qui correspondent à deux utilisations possibles de ces groupements :

- **mots-clefs du titre + groupements** est obtenue en effectuant une extension de la requête obtenue grâce à **mots-clefs du titre**. Pour chaque mot du titre, les mots de son groupe sont ajoutés à la requête.

- **mots-clefs du titre projetés**, génère la même requête que **mots-clefs du titre**, mais représente cette requête et les phrases dans l'espace des groupements de mots, où la valeur associée à un groupement c pour une phrase p donnée est le nombre d'occurrences des mots de c dans p .

2.3. Mesures de similarité

Nous utilisons la représentation $tf.idf$ des phrases et des requêtes pour obtenir une première mesure de similarité entre une requête q et une phrase p selon la formule suivante:

$$Sim_1(q, p) = \frac{\sum_{w \in q \cap p} tf(w, q)tf(w, p)idf^2(w)}{\|q\| \|p\|}$$

Où $tf(w, p)$ est le nombre d'occurrences du mot w dans la phrase p , $idf(w)$ est

l'*inverse document frequency* du mot w , et $\|x\| = \sqrt{\sum_{w \in x} (tf(w, x)idf(w))^2}$.

Enfin, nous avons modifié les mesures de similarité pour allouer un poids plus fort aux acronymes et accroître le score des phrases contenant les marqueurs linguistiques. Pour prendre les acronymes en compte, nous utilisons la même mesure que Sim_1 avec $tf(w, p)$ doublé dans le cas où w est un acronyme.

Résumé automatique avec un algorithme d'ordonnement 7

Les marqueurs linguistiques sont pris en compte par le biais de la mesure $Sim_3(q, p)$, qui vaut $2 * Sim_1(q, p)$ si p contient un marqueur linguistique, et $Sim_1(q, p)$ sinon. Enfin, d'autres mesures de similarités ont été utilisées: $Sim_4(q, p) = \sum_{w \in q \cap p} 1$, qui compte le nombre de mots communs entre la requête et la phrase, $Sim_5(q, p) = \sum_{w \in q \cap p} idf(w)$ et $Sim_6(q, p) = q.p$, le produit scalaire de q et de p dans l'espace de représentation considéré (espace des mots ou espace des groupements, selon les cas).

Les caractéristiques que nous avons dérivées des requêtes et des mesures de similarité précédentes sont résumées dans le tableau 1.

Caractéristique	(type de requête, mesure de similarité)
Titre	(mots-clefs du titre , Sim_1)
Titre + ACL	(mots-clefs du titre + ACL , Sim_1)
Titre + WN	(mots-clefs du titre + WordNet , Sim_1)
Titre + MFTD	(mots-clefs du titre + MPFD , Sim_1)
Titre + Groupements	(mots-clefs du titre + groupements , Sim_1)
Titre + Acronymes	(mots-clefs du titre , Sim_2)
Titre + Marqueurs	(mots-clefs du titre , Sim_3)
MotsCommuns	(mots-clefs du titre , Sim_4)
SommeIdfs	(mots-clefs du titre , Sim_5)
Titre projeté	(mots-clefs du titre projetés , Sim_6)
MPFgénérique	(MPF , Sim_1)

Tableau 1. Récapitulatif des caractéristiques.

3. Algorithmes d'apprentissage

Les approches existantes pour combiner les caractéristiques utilisent le cadre de la classification, soit avec le modèle Naive Bayes (Kupiec et al. 1995), soit avec la régression logistique (Amini et Gallinari 2002). La justification d'une approche de classification pour le résumé automatique est qu'une erreur de classification nulle implique que les scores alloués aux phrases pertinentes (resp. non pertinentes) par le classifieur sont toutes supérieures (resp. inférieures) à une constante c . L'ordonnement des phrases est alors correct.

Cependant, en pratique, l'erreur de classification n'est jamais nulle. Dans ce cas, si nous considérons, par exemple, une phrase non pertinente mal classée, l'information que nous avons est que son score s est supérieur à la constante c

précédente, mais il est impossible de savoir s'il existe une phrase pertinente du même document avec un score, lui aussi supérieur à c , mais inférieur à s . De plus, nous ne savons pas combien de phrases pertinentes du même document sont dans ce cas. Ainsi, l'erreur de classification ne fournit pas l'information suffisante pour prédire l'ordre induit par le score du classifieur des phrases d'un même document. Optimiser l'erreur de classification n'optimise donc pas les rangs des phrases pertinentes par rapport aux phrases non pertinentes d'un même document.

Le critère de classification ne semble donc pas exactement adapté à la tâche de résumé automatique. Afin d'apprendre des combinaisons de caractéristiques plus performantes, nous proposons alors d'utiliser le cadre offert par les algorithmes d'ordonnement, qui nous semble plus pertinent pour cette tâche. Un tel algorithme considère toutes les paires de phrases (p, p') d'un même document, telles qu'une des deux phrases p ou p' est pertinente, et l'autre est non pertinente. Le modèle est un algorithme de classification sur ces paires de phrases, tel qu'une paire est bien classée si et seulement si le score alloué par la fonction d'ordonnement H à la phrase pertinente est supérieur à celui alloué à la phrase non pertinente. L'erreur de classification sur les paires induit par H est appelé le *Ranking Loss* (noté ***RLoss*** dans la suite) et est égal à:

$$RLoss(D, H) = \frac{1}{|D|} \sum_{d \in D} \frac{1}{|S_d^1| |S_d^{-1}|} \sum_{s \in S_d^1} \sum_{s' \in S_d^{-1}} \mathbb{I}[H(s') \geq H(s)]$$

Où D est la collection de documents pour l'apprentissage, S_d^1 est l'ensemble des phrases pertinentes du document d , S_d^{-1} l'ensemble des phrases non pertinentes de ce document, et $\mathbb{I}[pr]$ vaut 1 si le prédicat pr est vrai, et 0 sinon.

Cette expression montre que minimiser *RLoss* (i.e. l'erreur de classification sur les paires de phrases) revient à minimiser, dans chaque document, la somme du nombre de phrases non pertinentes qui ont un score supérieur à une phrase pertinente. Intuitivement, les algorithmes d'ordonnement vont donc directement optimiser les rangs des phrases pertinentes. Plus formellement, si l'on considère un document d de la collection D fixé, la quantité

$$\frac{1}{|S_d^1| |S_d^{-1}|} \sum_{s \in S_d^1} \sum_{s' \in S_d^{-1}} \mathbb{I}[H(s') \geq H(s)]$$

est égale à la probabilité que $H(s') \geq H(s)$ pour deux phrases $s \in S_d^1$ et $s' \in S_d^{-1}$ tirées aléatoirement (Cortes et Mohri 2003). Ainsi, l'expression $RLoss(D, H)$ est la moyenne sur tous les documents de la collection de la probabilité qu'une phrase pertinente ait un score inférieur à une phrase non pertinente du même document. Comme l'objectif est précisément d'apprendre une fonction qui alloue aux phrases pertinentes des scores supérieurs aux phrases non pertinentes, l'utilisation d'algorithmes permettant de trouver une fonction H qui minimise le *RLoss* (i.e. d'algorithmes d'ordonnement) semble particulièrement adaptée à la tâche de résumé automatique.

Dans la suite de cette section, nous présentons l'algorithme de classification qui sera notre modèle de référence dans nos expériences, puis l'algorithme d'ordonnement utilisé.

3.1. Algorithme de classification: régression logistique

La régression logistique a déjà été utilisée avec succès en résumé automatique (Amini et Gallinari 2002) et a montré de bonnes performances en termes de rappel et de précision en tant qu'algorithme de combinaison de caractéristiques. Son choix comme modèle de référence est donc naturel. Par ailleurs, nous verrons plus tard qu'il est facile d'adapter la régression logistique à la tâche d'ordonnement qui nous intéresse ici. Ce type de classifieur est donc particulièrement adapté à la comparaison que nous faisons entre algorithmes de classification et d'ordonnement pour la tâche de résumé automatique.

L'entrée du classifieur logistique est une représentation des phrases en un vecteur de scores, (s_1, \dots, s_n) , où le score s_i est renvoyé par la caractéristique i de la section 2.2. Le classifieur logistique fait alors l'hypothèse suivante, étant donné une phrase $\mathbf{s} = (s_1, \dots, s_n)$:

$$P(\text{pertinente} / \mathbf{s}) = \frac{1}{1 + e^{-2 \sum_{i=1}^n \lambda_i s_i}}.$$

Les paramètres $\Lambda = (\lambda_1, \dots, \lambda_n)$ sont appris en maximisant la log-vraisemblance binomiale (Friedman et al. 1998), qui s'écrit:

$$L(D; \Lambda) = -\frac{1}{2} \sum_{y=-1,1} \frac{1}{|S^y|} \sum_{s \in S^y} \log(1 + e^{-2y \sum_{i=1}^n \lambda_i s_i})$$

Où D est l'ensemble des documents d'apprentissage, S^{-1} et S^1 sont respectivement l'ensemble des phrases non pertinentes et pertinentes de l'ensemble d'apprentissage, et y est la classe de la phrase ($y = 1$ pour une phrase pertinente, $y = -1$ sinon).

3.2. Adaptation de l'algorithme d'ordonnement pour le résumé automatique

Il existe plusieurs algorithmes d'ordonnement dans la littérature d'apprentissage automatique, qui sont des adaptations du perceptron (Collins 2002, Shen et Joshi 2004), ou de l'algorithme *AdaBoost*, appelé *RankBoost* (Freund et al. 2003). Pour la tâche de résumé, le nombre total de phrases dans la collection peut être très élevé, il est donc nécessaire d'avoir un algorithme simple et rapide. Les algorithmes basés sur le perceptron auraient dans notre cas une complexité quadratique par rapport au nombre d'exemples, alors que *RankBoost* dans sa configuration habituelle ne produit pas de combinaison linéaire des caractéristiques

d'entrée. Afin que la présentation reste simple, nous comparons dans cet article un algorithme de classification linéaire avec un algorithme d'ordonnement linéaire, appelé LinearRank dans la suite, qui combine la rapidité (complexité linéaire par rapport au nombre d'exemples) et la simplicité.

Une paire de phrases (s, s') est représentée par la différence de leurs vecteurs représentatifs: $(s_1-s'_1, \dots, s_n-s'_n)$. Nous voulons apprendre une fonction H , qui alloue un score à une phrase s en faisant une combinaison linéaire de ces caractéristiques $H(s) = \sum_{i=1}^n \theta_i s_i$. Le *Ranking Loss* décrit au début de la section s'écrit alors:

$$RLoss(D, H) = \frac{1}{|D|} \sum_{d \in D} \frac{1}{|S_d^1| |S_d^{-1}|} \sum_{s \in S_d^1} \sum_{s' \in S_d^{-1}} \left[\sum_{i=1}^n \theta_i (s_i - s'_i) \geq 0 \right]$$

Cette expression est une erreur de classification habituelle d'un classifieur linéaire sur les paires de phrases représentées par la différence de leurs vecteurs de scores. Il est alors possible d'adapter n'importe quel algorithme de classification linéaire à la tâche d'ordonnement (dans notre cas la régression logistique), afin d'optimiser le *Ranking Loss*. Adaptée à la tâche d'ordonnement, l'hypothèse logistique s'écrit:

$$P(1 / s, s') = \frac{1}{1 + e^{-2 \sum_{i=1}^n \theta_i (s_i - s'_i)}}$$

Où s est une phrase pertinente pour un document donné, et s' est une phrase non pertinente pour ce même document. $P(1 / s, s')$ est la probabilité a posteriori que la paire (s, s') soit bien classée. Les paramètres $\Theta = (\theta_1, \dots, \theta_n)$ sont appris en maximisant la log-vraisemblance binomiale correspondante:

$$L(D, \Theta) = -\frac{1}{|D|} \sum_{d \in D} \frac{1}{|S_d^1| |S_d^{-1}|} \sum_{s \in S_d^1} \sum_{s' \in S_d^{-1}} \log(1 + e^{-2 \sum_{i=1}^n \theta_i (s_i - s'_i)})$$

Cette expression oblige à considérer toutes les paires de phrases pour la calculer, et impose donc une complexité quadratique par rapport au nombre d'exemples. Pour résoudre ce problème, nous nous basons sur un résultat de (Friedman et al. 1998), qui montre que les paramètres maximisant la log-vraisemblance binomiale sont les mêmes que ceux qui minimisent le coût exponentiel:

$$ELoss(D, \Theta) = \frac{1}{|D|} \sum_{d \in D} \frac{1}{|S_d^1| |S_d^{-1}|} \sum_{s \in S_d^1} \sum_{s' \in S_d^{-1}} e^{-\sum_{i=1}^n \theta_i (s_i - s'_i)}$$

L'intérêt du coût exponentiel est que dans notre cas, il peut se calculer avec une complexité linéaire par rapport au nombre d'exemples, car $ELoss$ peut être réécrit comme suit :

$$ELoss(D, \Theta) = \frac{1}{|D|} \sum_{d \in D} \frac{1}{|S_d^1| |S_d^{-1}|} \left(\sum_{s \in S_d^1} e^{-\sum_{i=1}^n \theta_i s_i} \right) \left(\sum_{s' \in S_d^{-1}} e^{\sum_{i=1}^n \theta_i s'_i} \right)$$

Pour comparer l'approche d'ordonnement à l'approche habituelle de classification, il est nécessaire d'utiliser des modèles comparables. Ainsi, nous utilisons dans nos expériences un classifieur logistique et son équivalent dans le cas de l'ordonnement, pour lequel la fonction à optimiser $L(D, \Theta)$ est décrite au paragraphe précédent. Cependant, nous minimisons le coût exponentiel $ELoss(D, \Theta)$ dans le cas de l'ordonnement, car les paramètres obtenus sont les mêmes qu'en optimisant $L(D, \Theta)$, mais la complexité est réduite. Par ailleurs, La fonction de coût exponentiel est convexe, donc des algorithmes d'optimisation standards permettent d'effectuer l'optimisation. Dans notre cas, nous avons utilisé un algorithme d'*iterative scaling*, qui est une adaptation à l'ordonnement d'un algorithme décrit dans (Lebanon et Lafferty 2001) développé pour la classification.

4. Expériences

Un système de résumé automatique doit trouver l'information pertinente que l'utilisateur recherche, tout en éliminant l'information non pertinente. Il est alors crucial d'évaluer de tels systèmes sur leur capacité à extraire les parties informatives d'un document par rapport à l'attente de l'utilisateur. Dans ce but, nous avons utilisé deux ensembles de données, le premier venant de l'évaluation SUMMAC cmp_lg (http://www.itl.nist.gov/iaui/894.02/related_projects/tipster_summac/cmp_lg.html), et le second étant le corpus WIPO (<http://www.wipo.int/ibis/datasets/index.html>). Le corpus SUMMAC est composé de 183 articles scientifiques, extraits de conférences sponsorisées par l'association ACL (*Association for Computational Linguistics*), alors que le corpus WIPO, habituellement utilisé en catégorisation automatique, est composé de 75 000 descriptions de brevets en anglais. Dans nos expériences, nous n'avons considéré qu'un sous-ensemble de la collection, en sélectionnant 1000 documents au hasard.

L'évaluation d'un algorithme de résumé s'effectue en comparant les phrases sélectionnées par l'algorithme à un résumé de référence. Idéalement, ces résumés de référence sont constitués de phrases du document sélectionnées par des humains. Cependant, une évaluation fiable doit être réalisée sur un grand nombre de documents. Ainsi, il est difficile, pour des raisons de temps, de créer manuellement des résumés de référence pour les 1000 documents de la collection WIPO utilisée dans nos évaluations. Les résumés de référence, pour les deux collections utilisées, ont donc été créés de manière automatique, grâce à la technique d'alignement décrite dans (Marcu 1999). Cette méthode est un algorithme glouton, dont le but est de sélectionner le sous-ensemble des phrases d'un document, auquel le résumé fourni par les auteurs a été préalablement supprimé, et qui est le plus similaire à ce résumé. Le principe est de partir du document tout entier, puis d'enlever à chaque étape une

phrase telle que le sous-ensemble restant maximise la similarité avec l'*abstract*. L'algorithme s'arrête lorsqu'il n'est plus possible de supprimer une phrase sans faire diminuer la similarité du sous-ensemble déjà sélectionné. Cette technique d'évaluation des systèmes de résumé automatique a déjà été utilisée dans (Goldstein et al. 1999 et Amini et Gallinari 2002), et est justifiée par une étude décrite dans (Marcu 1999) qui montre que les résumés produits par cette technique sont proches des résumés effectués par des humains.

Le corpus WIPO contient des documents beaucoup plus hétérogènes que le corpus SUMMAC. Ainsi, en terme de comparaison d'algorithmes, il est intéressant d'expérimenter les différences de comportement d'un classifieur et d'un algorithme d'ordonnement sur des corpus homogènes comme SUMMAC et plus hétérogènes comme WIPO.

Dans nos expériences, des prétraitements ont été effectués sur les deux collections. Les balises XML ont été enlevées, ainsi que les méta-données (auteurs, sections, etc.). Les mots faisant partie d'un *anti-dictionnaire* ont été enlevés, et les frontières entre les phrases ont été trouvées en appliquant l'analyseur morphosyntaxique Tree-Tagger¹. Dans chaque collection, les mots apparaissant dans moins de deux documents n'ont pas été considérés, les documents pour lesquels la méthode d'alignement de (Marcu 1999) trouvaient une seule phrase ont été supprimés, ainsi que les documents dont le titre contenait moins de deux mots. Au total, 10 documents de la collection SUMMAC, et 146 du corpus WIPO ont été écartés. Le tableau 2 résume les caractéristiques des deux collections.

	SUMMAC	WIPO
# documents	173 (183)	854 (1000)
# moyen de phrases par doc.	156,57	179,63
# maximum de phrases par doc.	928	1507
# minimum de phrases par doc.	15	21
# docs. dans les bases apprentissage-test	73-100	360-494
# moyen de mots par phrase	11,73	14,22
Taille du vocabulaire	15 621	56 856
# phrases moyen du résumé de référence	11,44	6,07
# max de phrases dans le résumé de réf.	27	19
# min de phrases dans le résumé de réf.	3	2

Tableau 2. *Caractéristiques des corpus d'évaluation.*

¹ www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/DecisionTreeTagger.html

5. Résultats

L'évaluation des systèmes de résumé a fait l'objet d'un certain nombre de propositions, principalement dans le cadre du programme Tipster et des compétitions SUMMAC. Pour la tâche d'extraction des phrases pertinentes, nous avons utilisé la méthode d'évaluation des compétitions SUMMAC, qui mesure la performance d'un système en terme de rappel/précision à un taux de compression du document de 10%.

Cette section est organisée comme suit : la partie 5.1 présente les groupements de mots obtenus grâce à la méthode de la section 2.2. Les performances des différentes heuristiques, ainsi que des algorithmes d'apprentissage sont données en section 5.2 et 5.3, et la section 5.4 donne un exemple de résumé obtenu.

5.1. Groupements de mots

Nous donnons ici certains exemples de mots appartenant au même groupe, où les groupes sont trouvés par la technique de groupement décrite au paragraphe 2.2.

- *groupe 1* :

Transduction, language, grammar, set, word, information, model, number, words, rules, rule, lexical

- *groupe 2*:

Tag, processing, speech, recognition, morphological, Korean, morpheme

D'une façon générale, les groupements basés sur les cooccurrences globales dans les documents tendent à faire apparaître des mots reliés thématiquement. Ce résultat est cohérent avec les observations faites dans (Caillet et al. 2004), et est illustré par les deux exemples précédents. Utilisant ces groupements de mots, un article dont le titre contient le mot « language » pourra reconnaître que « grammar » est un mot thématiquement relié, ce qui n'est pas simple à trouver dans des thesaurus comme WordNet. De plus, pour le résumé automatique, ces cooccurrences globales permettent de trouver des mots reliés aux mots du titre, même s'ils ne cooccurrent avec ces derniers dans aucune phrase du document considéré. La condition est que les mots tendent à apparaître dans les mêmes documents en général. Ils permettent donc de trouver des mots pertinents que l'ACL serait incapable de trouver. Ils fournissent donc une nouvelle information, qui était indisponible aux techniques de l'état de l'art.

5.2. Performance des heuristiques

Pour chacun des corpus, nous avons évalué en terme de rappel et de précision chacune des caractéristiques décrites dans la section 2, afin de comparer les effets des différentes méthodes d'enrichissement de requête et de la représentation des phrases utilisant les groupements de mots. Les courbes de rappel/précision pour les deux corpus sont présentées figure 1. La comparaison des performances de la requête **Titre** avec **Titre+ACL**, **Titre+WN**, **Titre+MPF**, **Titre+Groupements** permet de comparer les effets des différents enrichissements, alors que la comparaison de **Titre Projeté** et de **Titre** permet de comparer la puissance des différentes représentations des phrases. Les résultats montrent que les trois meilleures caractéristiques sont **Titre+ACL**, **Titre projeté**, et **Titre+Groupements**. Ces résultats montrent que des enrichissements en considérant des co-occurrences locales (i.e. utilisant ACL), et globales (i.e. les groupements de mots) améliorent la performance de la requête initiale **Titre**, ce qui est cohérent avec les observations faites en recherche d'information *ad-hoc* (Xu et Croft 1996).

Cependant, nous pouvons remarquer que la requête **Titre+ACL** a une performance variable sur les deux corpus considérés: sur SUMMAC, cette requête a une précision de 70% lorsque le rappel est de 50% alors que la requête **Titre** obtient une précision de 40% à ce niveau de rappel. Par contre, sur WIPO, la différence de précision entre les deux requêtes est de 6% à ce même niveau de rappel. Cette différence pourrait être due au fait qu'il y a en moyenne moins de phrases dans les résumés du corpus WIPO (voir tableau 2). Il est alors possible que plus de phrases non pertinentes soient considérées dans le calcul des co-occurrences locales. La différence entre les deux caractéristiques **Titre+Groupements** et **Titre projeté** est que la première ne prend pas en compte tous les mots des groupements considérés, alors que la seconde considère les phrases représentées dans l'espace des clusters. Cette différence conduit à des calculs d'*idf* différents pour la seconde caractéristique, qui sont très influencées par le nombre de groupements de mots choisis. Dans nos expériences, la caractéristique **Titre+Groupements** a de meilleures performances sur les deux corpus. Cela peut être dû au fait que les groupements de mots contiennent trop de mots non pertinents, ce qui amène la caractéristique **Titre projeté** à allouer des scores élevés à des phrases non pertinentes. Ainsi, un travail ultérieur est nécessaire pour examiner précisément les deux points suivants. Premièrement, l'influence du nombre de groupements sur les performances relatives entre les deux caractéristiques est encore peu connue. Deuxièmement, il reste à étudier les avantages et les inconvénients de la représentation sur des espaces de groupements de mots par rapport à l'ajout des mots d'un groupement à une requête.

5.3. Performance des algorithmes d'apprentissage

Les performances des deux algorithmes de combinaison des caractéristiques précédentes sont tracées dans la figure 2. Sur les deux corpus, l'algorithme de classification (LogisticClassifier), notre modèle de référence, obtient des performances meilleures que chacune des caractéristiques seules. Cela confirme les résultats obtenus dans l'état de l'art (Kupiec et al. 1995, Amini et Gallinari 2002), qui montre que le cadre de la classification permet d'apprendre des combinaisons d'heuristiques performantes pour le résumé automatique. De plus, l'algorithme d'ordonnement (LinearRank) proposé obtient de meilleures performances que l'algorithme de classification sur les deux bases expérimentales. Ces résultats positifs confirment l'hypothèse émise dans cet article comme quoi le critère d'ordonnement, plus adapté au résumé automatique, permet d'apprendre des combinaisons plus performantes que celles obtenues dans le cadre de la classification.

Sur le corpus WIPO, la différence de performance entre LogisticClassifier et LinearRank varie entre 5 et 9%. D'autre part, la différence entre la meilleure caractéristique et l'algorithme de classification varie de 3 à 9% sur le corpus SUMMAC, et de 0 à 5% sur le corpus WIPO. Cela montre que la combinaison des caractéristiques trouvée par l'algorithme de classification, comparée à la meilleure caractéristique, varie beaucoup selon les corpus. Au contraire, l'algorithme d'ordonnement trouve une combinaison nettement supérieure à la meilleure caractéristique sur les deux corpus.

Une analyse des poids trouvés par les deux algorithmes d'apprentissage montre que les caractéristiques les plus importantes dans les combinaisons sont **Titre**, **Titre+ACL**, **Titre+Groupements**, **Titre projeté** et **MPFGénérique**. Cela montre que les algorithmes d'apprentissage prennent fortement en compte des caractéristiques selon deux critères: d'abord, leur capacité à allouer des scores élevés aux phrases pertinentes, et deuxièmement l'indépendance relative des caractéristiques entre elles. Ainsi, la caractéristique **MPFGénérique**, qui pourtant est la moins performante dans nos expériences, a un poids plus important dans la combinaison produite par l'algorithme d'ordonnement que les requêtes **Titre+WN** ou **Titre+Marqueurs**, parce que ces dernières sont plus corrélées à la requête **Titre**. Cette remarque est cohérente avec les études faites dans le domaine de la méta-recherche (Aslam et Montague 2001), qui montrent que pour que les combinaisons soient performantes, les caractéristiques doivent être indépendantes. De plus, cela confirme l'intérêt des clusters de mots pour le résumé automatique de texte, car les requêtes basées sur les clusters de mots sont performantes, tout en fournissant une information de pertinence indépendante des autres requêtes performantes comme **Titre+ACL**.

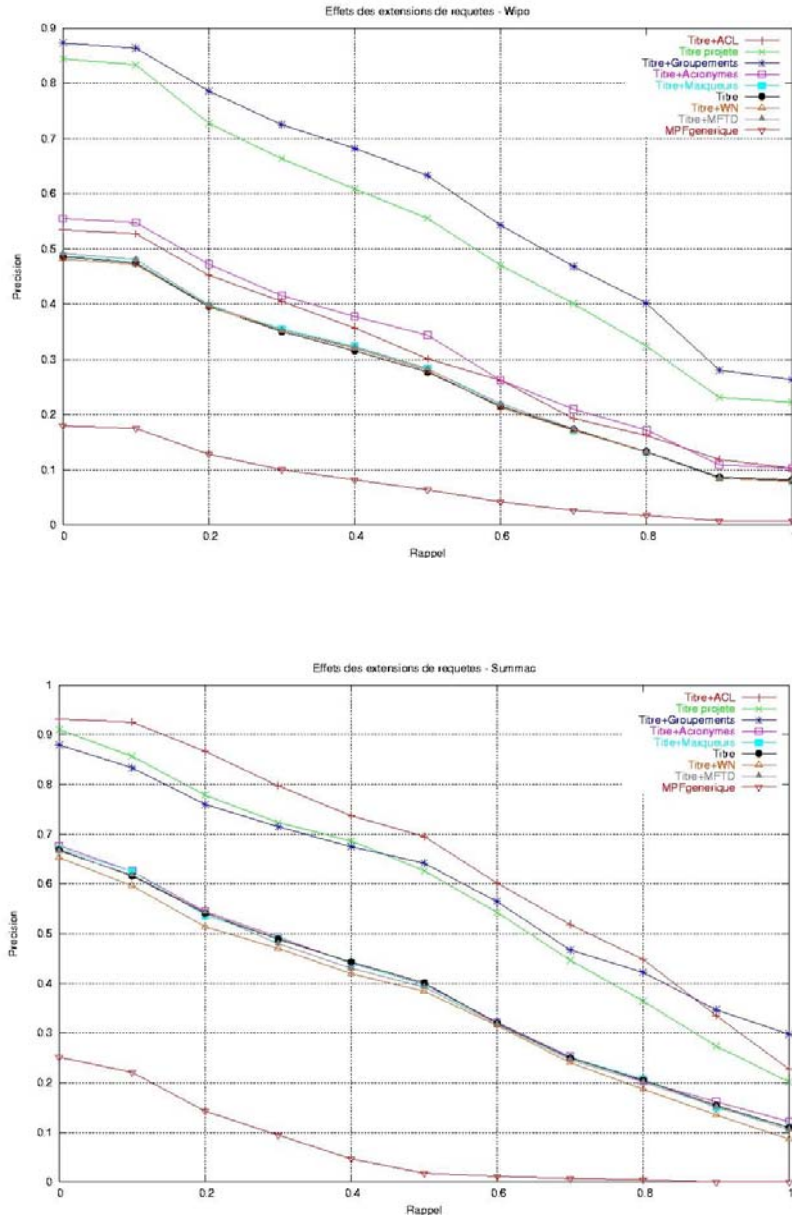


Figure 1. Effet des extensions de requêtes sur les corpus WIPO (haut) et SUMMAC (bas).

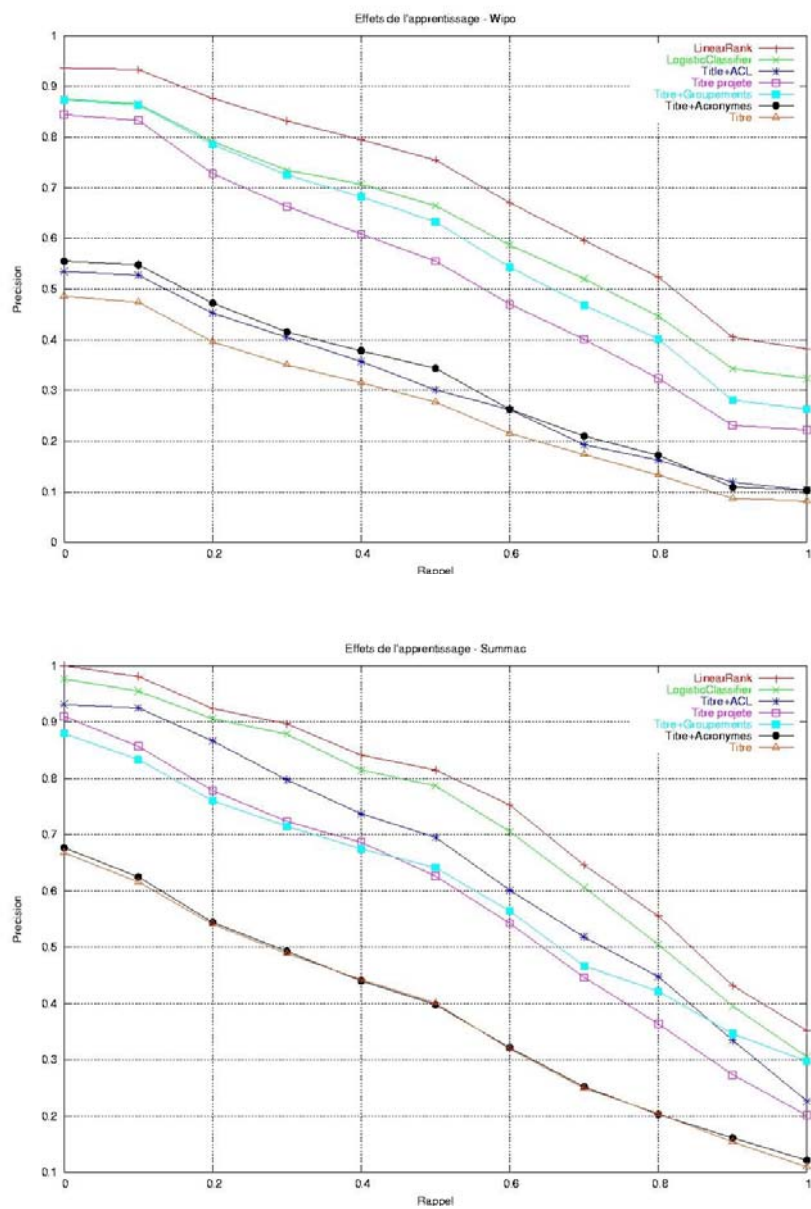


Figure 2. Effet de l'apprentissage sur les corpus WIPO (haut) et SUMMAC (bas).

5.4. Exemple de résumé obtenu

Nous présentons dans cette section un exemple de résumé obtenu pour l'article « Lexical Acquisition via Constraint Solving », de Ted Pedersen et Weidong Chen paru dans « AAAI 1995 Spring Symposium Series », qui fait partie de la base SUMMAC. Le résumé par extraction est présenté sans post-traitement : les phrases présentées sont données dans l'ordre des scores décroissants, et en particulier peuvent ne pas suivre l'ordre chronologique de l'article. La fonction utilisée pour ordonner les phrases est la fonction apprise grâce à l'algorithme d'ordonnement. Le résumé par extraction est celui obtenu pour un taux de compression de 10%, autrement dit, il contient un nombre de phrases égal à 10% le nombre de phrases total de l'article. Pour vérifier la pertinence du résumé par extraction, le résumé de l'article fourni par les auteurs est recopié. Les deux résumés sont présentés tableau 3.

Article : Ted Pedersen et Weidong Chen, 1995, « Lexical Acquisition via Constraint Solving », AAAI 1995 Spring Symposium Series.	
Résumé fourni par les auteurs	Résumé par extraction de phrases
<p>This paper describes a method to automatically acquire the syntactic and semantic classifications of unknown words. Our method reduces the search space of the lexical acquisition problem by utilizing both the left and the right context of the unknown word. Link Grammar provides a convenient framework in which to implement our method.</p>	<p>This paper explores lexical acquisition in a framework without phrase structure rules. We selected link grammar due to its explicit use of right and left context and the availability of an implementation that includes a word lexicon. Link grammar is a context-free linguistic framework that is lexically based. Parsing a sentence in link grammar consists of choosing one disjunct for each word such that it can be connected to the surrounding words as specified in that disjunct. An implementation is under way to extend the parser of link grammar to automatically acquire the syntax and semantics of unknown words. An interesting topic is to integrate feature structures and unification with link grammar to allow more expressive handling of semantic information.</p>

Tableau 3. Exemple de résumé par extraction obtenu.

Cet exemple montre que l'approche de résumé par extraction est suffisante pour donner un aperçu du contenu, et que cet aperçu est bien plus informatif que le titre uniquement de l'article. L'abstract de l'article fourni par les auteurs précise, par rapport au titre, qu'ils essayent de déterminer de façon automatique les catégories syntaxiques et sémantiques des mots, ce qui est présent dans le résumé par extraction à l'avant dernière phrase. L'utilisation des contextes à gauche et à droite du mot, ainsi que l'utilisation de « link grammar » sont présents dans le résumé par extraction à la deuxième et troisième phrase. Ainsi, bien que l'ordre des informations ne soit pas l'ordre chronologique exact de l'abstract fourni par les auteurs, elles sont suffisantes pour qu'un utilisateur comprenne le contenu de l'article. De plus, les phrases extraites contiennent les mots « words », « grammar », qui sont dans le même groupe que le mot « lexicon » (cf. paragraphe 5.2), bien qu'elles ne co-occurrent qu'avec peu de mots du titre dans les phrases du document. Cela montre l'intérêt des clusters de mots qui recherchent des cooccurrences globales sur l'ensemble de la collection, ainsi que leur intérêt pour le résumé automatique.

6. Conclusion

Dans cet article, nous avons présenté de nouvelles caractéristiques pour le résumé automatique de texte, et proposé l'utilisation d'un algorithme d'ordonnement pour la combinaison de caractéristiques.

Les caractéristiques introduites utilisent des groupements de mots, qui contiennent les mots apparaissant dans les mêmes documents. Ces groupements peuvent être utilisés pour trouver des mots dans le cas d'un enrichissement de requête, ou pour donner une représentation des phrases, qui est une alternative à la représentation classique sous forme de sac-de-mots. Dans les deux cas, les performances observées en terme de précision et de rappel sont prometteuses, avec des gains de précision par rapport avec la meilleure caractéristique de l'état de l'art compris entre 10% et 20% selon le niveau de rappel sur le corpus WIPO, et au niveau de la meilleure caractéristique de l'état de l'art sur le corpus SUMMAC. Les caractéristiques utilisant les groupements de mots apportent de plus une information supplémentaire et indépendante par rapport aux caractéristiques habituelles utilisées en résumé automatique. Elles sont donc d'un grand intérêt dans le cas où l'on souhaite construire une combinaison de caractéristiques performante.

L'utilisation d'un algorithme d'apprentissage utilisant le cadre de l'ordonnement est nouvelle pour le résumé automatique. De plus, les résultats théoriques de (Cortes et Mohri 2003) suggèrent que le cadre de la classification est sous-optimal pour apprendre une fonction d'ordonnement. Or, la tâche de résumé automatique est intrinsèquement une tâche d'ordonnement, et les résultats expérimentaux que nous avons obtenus confirment l'hypothèse que le cadre offert par les algorithmes d'ordonnement est plus adapté à la tâche.

Références

- Amini M.-R., Gallinari P., « The Use of unlabeled data to improve supervised learning for text summarization ». *Proceedings of the 25th ACM SIGIR Conference*, 2002, p. 105-112.
- Aslam, J.A., Montague, M., « Models for metasearch », *Proceedings of the 24th ACM SIGIR Conference*, 2001, p. 276 - 284.
- Caillet M., Pessiot J.-F., Amini M.-R., Gallinari P. « Unsupervised Learning with Term Clustering for Thematic Segmentation of Texts », *Proceedings of RIAO*, 2004, p. 648-660.
- Caruana R., Alexandru N.-M. « Data mining in metric space: an empirical analysis of supervised learning performance criteria », *Proceedings of the 10th ACM SIGKDD Conference*, 2004, p. 69-78.
- Collins, M., « Ranking algorithms for named-entity extraction: Boosting and the voted perceptron », *Proceedings of the 40th Annual Meeting of the ACL*, 2002, p. 489-496.
- Cortes, C., Mohri M. « AUC optimization vs error rate minimization », *Proceedings of NIPS*, 2003.
- Chuang W.T., Yang J., « Extracting sentence segments for text summarization: a machine learning approach ». *Proceedings of the 23rd ACM SIGIR Conference*, 2000, p. 152-159.
- Fellbaum C., « WordNet, an Electronic Lexical Database », MIT Press, Cambridge MA, 1998, ISBN 0-262-06197-X.
- Freund, Y., Iyer, R., Schapire, R.E., Singer, Y., « An efficient boosting algorithm for combining preferences ». *Journal of Machine Learning Research*, n° 4, 2003, p. 933-969.
- Friedman J., Hastie T., Tibshirani R., « Additive Logistic Regression: a Statistical View of Boosting ». Technical Report, 1998, Stanford University.
- Goldstein J., Kantrowitz M., Mittal V., Carbonell J., « Summarizing Text Documents: Sentence Selection and Evaluation Metrics », *Proceedings of the 22nd ACM SIGIR conference*, 1999, p. 121-127.
- Lebanon, G., Lafferty J., « Boosting and maximum likelihood for exponential models », Technical Report CMU-CS-01-144, 2001, School of Computer Science, CMU.
- Kupiec J., Pederson J., Chen F.A., « Trainable Document Summarizer », *Proceedings of the 18th ACM SIGIR Conference*, 1995, p. 68-73.
- Luhn P.H., « Automatic creation of literature abstracts », *IBM Journal of Research and Development*, 1958, p. 155-164.
- Marcu D., « The Automatic Construction of Large-Scale corpora for Summarization Research ». *Proceedings of the 22nd ACM SIGIR Conference*, 1999, p.137-144.
- Mitra M., Singhal A., Buckley C., « Automatic Text Summarization by Paragraph Extraction ». *Proceedings of the ACL'97/EACL'97 Workshop on Intelligent Scalable Text Summarization*, 1997, p. 31-36.
- Paice C.D., Jones P.A., « The identification of important concepts in highly structured technical papers ». *Proceedings of the 16th ACM SIGIR Conference*, 1993, p. 69-78.

Résumé automatique avec un algorithme d'ordonnement 21

- Sparck-Jones K., « Discourse modeling for automatic summarizing ». Technical Report 29D, 1993, Computer Laboratory, university of Cambridge.
- Shen, L., Joshi, A.K., « Ranking and Reranking with Perceptron ». *Machine Learning, Vol. 60*, 2005, p. 73-96.
- Symons M.J., « Clustering Criteria and Multivariate Normal Mixture ». *Biometrics*, Vol. 37, 1981, p. 35-43.
- Xu, J., Croft, W.B., « Query expansion using local and global document analysis », *Proceedings of the 19th ACM SIGIR Conference*, 1996, p. 4-11.