Résumé automatique de texte avec un algorithme d'ordonnancement

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'ordonnancement. L'objectif est d'extraire les phrases d'un document qui sont les plus représentatives de son contenu. Pour se faire, chaque phrase d'un document est représentée par un vecteur de scores de pertinence, où chaque score est un score de similarité entre une requête particulière et la phrase considérée. L'algorithme d'ordonnancement effectue alors une combinaison linéaire de ces scores, avec pour but d'affecter aux phrases pertinentes d'un document des scores supérieurs à ceux des phrases non pertinentes du même document. Les algorithmes d'ordonnancement ont montré leur efficacité en particulier dans le domaine de la méta-recherche, et leur utilisation pour le résumé est motivée par une analogie qui peut être faite entre la méta-recherche et le résumé automatique qui consiste, dans notre cas, à considérer les similarités des phrases avec les différentes requêtes comme étant des sorties de différents moteurs de recherche. Nous montrons empiriquement que l'algorithme d'ordonnancement a de meilleures performances qu'une approche utilisant un algorithme de classification sur deux corpus distincts.*

ABSTRACT. *In this paper, we propose a novel approach for automatic text summarization based on machine learning, using a ranking algorithm. The aim of a summarization system is to extract the sentences of a document which are the most representative of its content. In our approach, a given sentence is represented by a vector of relevance scores, where each score is a similarity score between the sentence and a particular query. The ranking algorithm produces a linear combination of these scores which is trained to give higher scores to relevant sentences than to irrelevant ones of the same document. Ranking algorithms have already shown effectiveness in the domain of metasearch, and their use in text summarization is motivated by an analogy between those two domains. Indeed, the similarity scores between the sentences and the different queries can be seen as the outputs of different search engines. We show empirically that the ranking algorithm outperforms a summarization system using a classification algorithm, on two distinct corpora.*

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

KEYWORDS: *text summarization, ranking algorithms, machine learning.*

1. Introduction

Avec l'accroissement actuel des données disponibles en ligne, il est plus que jamais nécessaire d'aider les utilisateurs à avoir un aperçu rapide de l'information. Le résumé automatique de texte donne un aperçu de l'information contenue dans un document, qui permet à l'utilisateur d'avoir plus facilement accès au contenu d'un grand nombre de documents, et facilite la navigation dans un grand corpus. Couplé à un moteur de recherche, cela permettrait aussi aux utilisateurs d'évaluer rapidement la pertinence réelle d'un document par rapport à leur requête.

L'origine de la tâche de résumé automatique de texte remonte aux années 50 (Luhn 1958), les problèmes posés étaient comment un système automatique pourrait être appliqué à de grands corpus, et comment un tel système pourrait générer des résumés synthétiques. Générer de tels résumés nécessite des capacités de compréhension du langage, d'abstraction, ainsi que de génération de langage (Sparck-Jones 1993), rendant les traitements trop complexes pour être appliqués à des grandes bases de documents. Une simplification de la tâche a alors été proposée, qui consiste à extraire d'un document les composants textuels qui sont le plus représentatifs de son contenu. Ces composants sont généralement des phrases, mais peuvent aussi être des paragraphes (Mitra et al. 1997).

Dans cet article, nous présentons une méthode statistique pour le résumé automatique à base d'apprentissage qui extraie les phrases pertinentes d'un document, c'est-à-dire celles qui reflètent le mieux son contenu. Les précédentes approches d'apprentissage pour cette tâche utilisaient le cadre de la classification supervisée, dans lequel la base d'apprentissage est constituée de documents, avec pour chaque document les phrases qui doivent être extraites pour composer le résumé. Un classifieur est alors entraîné pour discriminer les phrases qui doivent être dans le résumé de celles qui ne doivent pas, en effectuant une combinaison de différentes caractéristiques représentant les phrases, et permettant de prédire leur appartenance au résumé (Kupiec et al. 1993, Chuang et Yang 2000, Amini et Gallinari 2002). Après entraînement de ces systèmes, l'extraction des phrases pour le résumé d'un nouveau document est effectuée en ordonnant les phrases selon le score renvoyé par le classifieur, qui est une estimation de la probabilité de la phrase d'appartenir au résumé.

Nous pensons que le critère de classification n'est pas bien adapté à la tâche de résumé de texte, bien qu'il puisse être utilisé pour apprendre des fonctions performantes: un classifieur va apprendre une fonction qui tente d'allouer à toutes les phrases pertinentes un score supérieur à une certaine constante c, et à toutes les phrases non pertinentes un score inférieur à c. Une telle contrainte est plus forte que ce que requiert la tâche, qui nécessite plus simplement que les phrases pertinentes d'un document aient des scores supérieurs aux phrases non pertinentes de ce même document.

Le domaine de la méta-recherche fournit cependant des algorithmes d'apprentissage (par ex. Aslam et Montague 2001) qui optimisent des critères, autres que ceux utilisés en classification, plus adaptés à la tâche de résumé automatique. L'objectif de la méta-recherche est de combiner les sorties d'un grand nombre de moteurs de recherche, en supposant que les erreurs de jugement de ces différents moteurs sont indépendantes. L'analogie avec le résumé automatique est que, dans notre cas, nous allons combiner un certain nombre de scores de pertinence disponibles pour chaque phrase, qui chacun peut être vu comme la sortie d'un moteur de recherche. En nous inspirant des algorithmes développés en méta-recherche, nous allons développer un système basé sur un algorithme d'ordonnancement qui combine les caractéristiques des phrases.

Un algorithme d'ordonnancement permet d'apprendre un ordre partiel sur les données. 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'ordonnancement 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. Se basant sur l'analogie avec la méta-recherche, cet article va comparer un algorithme d'ordonnancement avec un algorithme de classification pour la combinaison de scores de pertinence pour le résumé automatique de texte, afin de tester la performance de ces algorithmes pour cette tâche.

Le plan de cet article est le suivant. Dans la section 2, nous décrivons les 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 clusters de mots. Une telle représentation est nouvelle en résumé automatique. Dans la section 3, nous décrivons l'algorithme d'ordonnancement 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 SUMMAC, et un sous-ensemble du corpus WIPO. L'interprétation des résultats est donnée en section 5, et la section 6 conclue l'article.

2. Caractéristiques pour le résumé automatique

Les caractéristiques que nous décrivons dans cette section sont utilisées pour représenter les phrases, et la fonction qui sera apprise par l'algorithme sera une combinaison linéaire de ces caractéristiques. Chaque phrase de la collection est décrite par un vecteur de scores, où la valeur d'une dimension donnée correspond au score de la caractéristique associée à la dimension. Dans le cadre de notre analogie

avec la méta-recherche, chaque caractéristique est considérée comme la sortie d'un moteur de recherche. 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.

Pour le résumé automatique, (Paice et Jones 1993) regroupent les caractéristiques des phrases à prendre en compte en sept catégories: 1) les marqueurs linguistiques (aussi appelés *cue-words*), 2) les acronymes, 3) les mots fréquents d'un document, 4) les mots-clefs 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-clefs 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. Chaque caractéristique est définie par un couple (requête, similarité entre la requête est une phrase). 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.

2.1. *Les requêtes*

Toutes nos requêtes sont issues de deux requêtes types, la première constituée des mots les plus fréquents de la collection de documents considérée, notée **MFT** (*Most Frequent Terms*) dans la suite, et la seconde est constituée des mots du titre du document considérée, notée **title keywords** dans la suite.

A partir de la seconde requête, nous créons trois nouvelles requêtes qui permettent de prendre en compte des enrichissements différents, qui ont montré leur efficacité pour la tâche de résumé automatique (Goldstein et al. 1999). La première, notée **title keywords and LCA,** est obtenue en effectuant un enrichissement de **title keywords** grâce à la technique de l'analyse de contexte local (LCA) (Xu et Croft 1996). La seconde requête, **title keywords and WordNet** est obtenue en ajoutant aux mots du titre leurs synonymes trouvés dans le thesaurus WordNet (FellBaum 1998). La troisième requête, appelée **title keywords and document MFT,** est obtenue en rajoutant aux mots du titre, les mots les plus fréquents du document considéré.

Deux autres requêtes sont obtenues à partir de groupement de mots. Les mots sont dans le même groupement s'ils ont des fortes probabilités de co-occurrence dans les phrases. Les groupements sont obtenus en utilisant la procédure décrite dans (Caillet et al. 2004), qui consiste à représenter chaque mot *w* par un vecteur de dimension *N*, où *N* est le nombre de documents dans la collection, et la valeur la *i*-ième caractéristique du vecteur est le nombre de fois que le mot *w* apparaît dans la

phrase *i*. Le regroupement est ensuite effectué grâce à l'algorithme Naive-Bayes, en maximisant le critère de vraisemblance classifiante (Symons 1983). Le nombre de groupes à trouver est un hyperparamètre de l'algorithme que nous avons arbitrairement fixé à *W/100*. *W* est ici le nombre de mots dans la collection. Une fois ces regroupements obtenus, une première requête est créée à partir de **title keywords** en ajoutant aux mots du titre les mots qui sont dans le même regroupement. Cette requête sera notée **title keywords and word-clusters** dans la suite. Une seconde requête, notée **projected title keywords,** est obtenue en représentant la requête **title keywords** et les phrases dans l'espace des groupes, où la valeur associée à un cluster *c* pour une phrase *p* donnée est le nombre d'occurrences des mots de *c* dans *p*.

2.2. *Mesures de similarité*

Suivant (Kupiec et al. 1995), 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:

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

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.

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).

3. Algorithmes d'apprentissage

Pour combiner les caractéristiques, les approches précédentes étaient effectuées dans 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'ordonnancement 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.

C'est pourquoi nous pensons que le cadre offert par les algorithmes d'ordonnancement sera plus efficace en pratique. Un tel algorithme considère toutes les paires de phrases *(p, p')* d'un même document, telles qu'une des deux phrases *p* et *p'* est pertinente, et l'autre est non pertinente. L'algorithme est un algorithme de classification sur ces paires de phrases, tel que une paire est bien classée si et seulement si le score alloué par la fonction d'ordonnancement *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:

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

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 [[*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. Les algorithmes d'ordonnancement vont donc directement optimiser les rangs des phrases pertinentes, ce qui justifie leur utilisation à la place d'un algorithme de classification pour 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'ordonnancement 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 terme 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'ordonnancement qui nous intéresse ici. Ce type de classifieur est donc particulièrement adapté à la comparaison que nous faisons entre algorithmes de classification et d'ordonnancement pour la tâche de résumé automatique.

L'entrée du classifieur logistique est une représentation des phrases en un vecteur de scores, (s_1, …, 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 $\boldsymbol{s}$ = (s_1, …, s_n) :

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

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

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

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).Les pages possèdent les caractéristiques suivantes :

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

Il existe plusieurs algorithmes d'ordonnancement 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'ordonnancement 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 ($\boldsymbol{s}$, $\boldsymbol{s'}$) est représentée par la différence de leurs vecteurs représentatifs: (s_1-s'_1, …, s_n-s'_n). Nous voulons apprendre une fonction *H*, qui alloue un score à une phrase $\boldsymbol{s}$ 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:

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

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'ordonnancement (dans notre cas la régression logistique), afin d'optimiser le *Ranking Loss*. Adaptée à la tâche d'ordonnancement, 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'***) est bien classée. Les paramètres $\Theta = (\theta_1, ..., \theta_n)$ sont appris en maximisant la log-vraisemblance binomiale correspondante:

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

Cette expression oblige à considérer toutes les paires de phrases pour la calculer, et imposerait 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 ont montré que les paramètres maximisant la log-vraisemblance binomiale sont les mêmes que ceux qui minimisent le coût exponentiel:

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

L'intérêt du coût exponentiel est que dans notre cas, il peut se calculer avec une complexité linéaire par rapport au nombre d'exemple, ce qui est immédiat en le ré-écrivant comme suit:

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

Ainsi, dans nos expériences, nous minimisons le coût exponentiel, ce qui revient aux mêmes paramètres que le modèle d'ordonnancement logistique, et donc cohérent dans la comparaison que nous voulons faire avec la classification logistique pour la

tâche de résumé. La fonction de coût exponentiel est convexe, donc des algorithmes d'optimisation standard permettent d'effectuer l'optimisation. Dans notre cas, nous avons utilisé un algorithme d'*iterative scaling*, qui est une adaptation à l'ordonnancement 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 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, alors que le corpus WIPO, habituellement utilisé en catégorisation automatique, est composé de 75 000 description 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. La difficulté vient du fait qu'une évaluation fiable nécessite un grand nombre de documents, et qu'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 l'*abstract* a été préalablement supprimé, et qui est le plus similaire à cet *abstract*. 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) comme quoi 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'ordonnancement sur des corpus homogènes comme SUMMAC et plus hétérogènes comme WIPO.

Dans nos expériences, des prétraitements ont été effectués. Les balises XML ont été enlevées, ainsi que les méta-recherche (auteurs, sections, etc.). Les mots faisant partie d'une *anté-dictionnaire* ont été enlevés, et les frontières entre les phrases ont été trouvées en appliquant l'analyseur morphosyntaxique Tree-Tagger (www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/DecisionTreeTagger.html). 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 phrases ont été supprimés, ainsi que les documents dont titre contenait moins de deux mots. Au total, 10 documents de la collection SUMMAC, et 146 du corpus WIPO ont été mis à l'écart. Le tableau 1 résume les caractéristiques des deux collections.

	SUMMAC	WIPO
Nombre de 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 1. *Comparaison des corpus d'évaluation.*

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%.

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, ainsi que de la représentation des phrases utilisant les groupements de mots. Les courbes de rappel/précision pour les deux corpus sont données figure 1. La comparaison des performances de la requête **Title** avec **Title+LCA**, **Title+WN**, **Title+MFT**, **Title+Term-clusters**

permet de comparer les effets des différents enrichissements, alors que la comparaison de **Projected Title** et de **Title** permet de comparer la puissance des différentes représentations des phrases. Les résultats montrent que les trois meilleures caractéristiques sont **Title+LCA**, **Projected Title**, et **Title+Term-clusters**. Ces résultats montrent que des enrichissements en considérant des co-occurrences locales (i.e. utilisant LCA), et globales (i.e. les groupements de mots) améliorent la performance de la requête initiale **Title**, ce qui est cohérent avec les observations faites en recherche d'information *ad-hoc*.

Cependant, nous pouvons remarquer que la requête **Title+LCA** 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 **Title** 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 1). 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 **Title+Term-clusters** et **Projected Title** est que la première ne prend pas en compte tous les mots des groupements considérés, alors que la seconde considère les phrase représentées dans l'espace des clusters. Cette différence conduit à des calculs d'*idf* qui varient pour la seconde caractéristique, qui sont très influencées par le nombre de groupements de mots choisis. Dans nos expériences, la requête **Title+Term-clusters** 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 requête **Projected Title** à allouer des scores élevés à des phrases non pertinentes. Ainsi, un travail ultérieur est nécessaire pour étudier l'influence du nombre de groupements sur les performances relatives entre les deux requêtes, et pour comprendre entièrement l'effet de la représentation sur des espaces de groupements de mots sur le résumé automatique par rapport à ajouter les mots à une requête.

Les performances des deux algorithmes de combinaison des caractéristiques précédentes sont tracées dans la figure 2. Sur les deux corpus, les algorithmes de classification (LogisticClassifier) et d'ordonnancement (LinearRank) obtiennent de meilleurs résultats que les caractéristiques seules. Cela signifie que chacun des deux cadres d'apprentissage est adapté à combiner des caractéristiques pour le résumé automatique, mais que pour le résumé automatique, le cadre des algorithmes d'ordonnancement est plus adapté.

Sur le corpus WIPO, elle 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'ordonnancement trouve une combinaison nettement supérieure à la meilleure caractéristique sur les deux corpus.

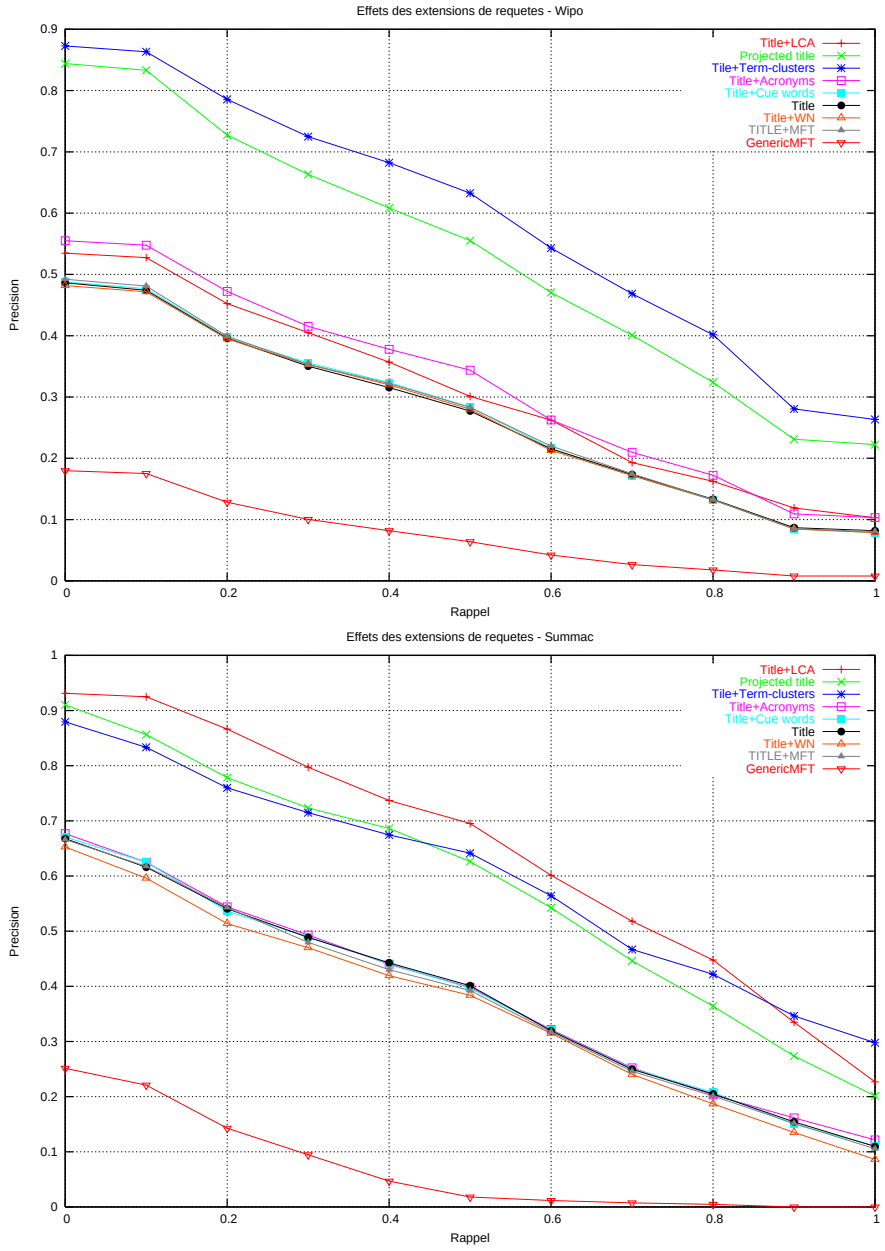

Figure 1. *Effet des extensions de requêtes sur les corpus SUMMAC (haut) et WIPO (bas).*

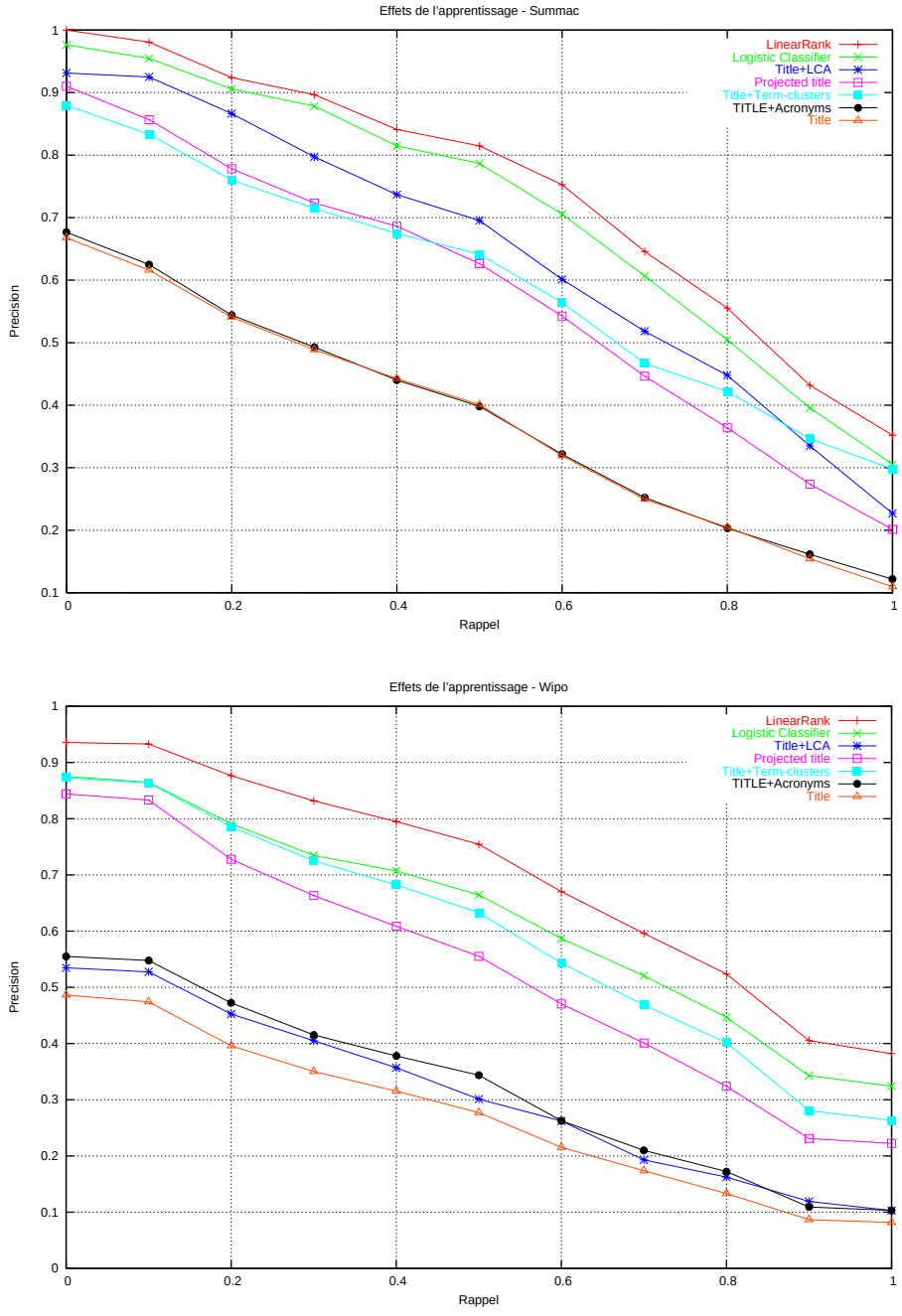

Figure 2. *Effet de l'apprentissage sur les corpus SUMMAC (haut) et WIPO (bas).*

Une analyse des poids trouvés par les deux algorithmes d'apprentissage montre que les caractéristiques les plus importantes dans les combinaisons sont **Title**, **Title+LCA**, **Title+Term-clusters**, **Projected Title** et **genericMFT**. 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 requête **genericMFT**, qui pourtant est la moins performante dans nos expériences, a un poids plus important dans la combinaison produite par l'algorithme d'ordonnancement que les requêtes **Title+WN** ou **Title+Cue Words**, parce que ces dernières sont plus corrélées à la requête **Title**. Cette remarque est cohérente avec les études faites dans le domaine de la méta-recherche, 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 **Title+LCA**.

6. Conclusion

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

Les caractéristiques introduites utilisent des groupements de mots, qui contiennent les mots occurrant dans les mêmes phrases. 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. De plus, ils apportent 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'ordonnancement est nouvelle pour le résumé automatique. Elle est motivée par une analogie entre cette tâche et le problème de méta-recherche. Nous avons montré empiriquement que ce type d'algorithme obtient de meilleures performances qu'un algorithme de classification. Ceci est probablement dû au fait que le critère des algorithmes d'ordonnancement est plus adapté à la tâche de résumé que celui des algorithmes de classification.

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 SIGIR 2001.

Caillet M., Pessiot J.-F., Amini M.-R., Gallinari P. « Unsupervised Learning with Term Clustering for Thematic Segmentation of Texts », *Proceedings of RIAO*, 2004

Collins, M., « Ranking algorithms for named-entity extraction: Boosting and the voted perceptron ». *Proceedings of ACL-2002*.

Chuang W.T., Yang J., « Extracting sentence segments for text summarization: a machine learning approach ». *Proceedings of the 23th ACM SIGIR conference*, 2000, p. 152-159.

Fellbaum C., « WordNet, an Electronic Lexical Database », MIT Press, Cambridge MA, 1998.

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 SIGIR*, 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, 1995, p. 68-73.

Luhn P.H., « Automatic creation of litterature abstracts », IBM Journal, 1958, p. 159-165.

Marcu D., « The Automatic Construction of Large-Scale corpora for Summarization Research ». Proceedings of the 22nd ACM SIGIR conference, 1999.

Mitra M., Singhal A., Buckley C., « Automatic Text Summarization by Paragraph Extraction ». Proceedings of the ACL'97/EACL'97 Workshop, 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, 1993, p. 69-78.

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, Special Issue on Learning in Speech and Language Technologies, 2004.

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 SIGIR 1996.