Prise en Compte de la Structure des Documents pour la Découverte d'Informations Inattendues

François Jacquenet, Christine Largeron

Prise en Compte de la Structure des Documents pour la Découverte d'Informations Inattendues

Using the Structure of Documents to Discover Unexpected Information

F. Jacquenet - C. Largeron
EURISE - Université Jean Monnet
23 rue du Docteur Paul Michelon
42023 Saint-Etienne Cedex 2
Francois.Jacquenet@univ-st-etienne.fr
Christine.Largeron@univ-st-etienne.fr

Résumé

Dans cet article nous nous intéressons à la prise en compte de la structure des documents dans un processus de découverte d'informations inattendues au sein d'un corpus de documents textuels. Faisant suite à un premier travail visant à concevoir et implanter des mesures d'inattendu dans un système baptisé UnexpectedMiner, nous avons cherché à améliorer les performances de celui-ci en prenant en compte la structure des documents analysés. Chaque partie des documents est ainsi pondérée par des coefficients dont les valeurs sont déterminées par un algorithme d'optimisation. Ces coefficients sont alors intégrés dans les mesures d'inattendu utilisées par UnexpectedMiner pour déterminer si un document présente un caractère inattendu ou pas. Les performances de notre nouveau système sont évaluées et mettent en évidence les améliorations de performances induites par la prise en compte de la structure des documents.

Mots Clefs

Fouille de Textes, Information Inattendue, Recherche d'Information, Structure du Document.

Abstract

In this paper we are interested in taking into account the structure of the documents during the discovery of unexpected information in textual databases. Following our first work that aimed at designing and integrating, in the UnexpectedMiner system, some measures for the evaluation of the unexpectedness of documents, we wanted to improve the system by taking into account the structure of the documents processed. Each part of the documents are weigthed by some coefficients whose values are determined by optimization techniques. Those coefficients are then used in the unexpectedness measures used by UnexpectedMiner to determine if a document contains some unexpected information or not. The efficiency of our new system is then evaluated and the experiments put forward the improvements induced by the use of the structure of the documents.

Keywords

Text Mining, Unexpected Information, Information Retrieval, Structure of Documents.

1 Introduction

La découverte automatique d'informations inattendues ou nouvelles dans des documents textuels est une tâche difficile mais particulièrement intéressante compte tenu de ses applications potentielles. Il peut s'agir par exemple de repérer parmi les questions adressées à des listes de diffusion celles qui n'ont pas encore été répertoriées dans les foires aux questions (FAQ), ou bien de trouver parmi des nouvelles publiées dans la presse celles qui traitent d'un nouveau sujet. Dans le domaine de la fouille de données sur le Web, on peut rechercher des pages inattendues sur un site WEB par rapport à un site de référence ou encore, dans le cadre de la veille technologique, on peut identifier des signaux faibles dans des bases d'articles scientifiques et techniques, de brevets, *etc.* Dans ces différentes applications, le but est de repérer des informations inattendues en ce sens qu'elles étaient inconnues auparavant de l'utilisateur. Un certain nombre de recherches ont déjà été consacrées à l'extraction automatique, à partir de textes, de ce qui est appelé, selon les auteurs, des informations inattendues ou nouvelles, des événements rares ou encore des sujets émergents. Parmi les premiers travaux portant sur ce sujet, on peut citer

le programme *TDT* (*Topic Detection and Tracking* [1]) lancé par la *DARPA* dès 1996 dans le but d'identifier de nouveaux événements dans un flux de nouvelles journalistiques [1, 15]. Les principales approches proposées dans *TDT* reposent sur des algorithmes de classification incrémentale, des techniques de plus proche voisin et des modèles probabilistes. Plus récemment, un challenge a été organisé sur le thème de la détection de nouveauté (*Novelty detection*) dans le cadre de la conférence *TREC* [2]. Toutefois, le corpus proposé pour *TREC 2003* est composé de phrases et non de textes [13]. De plus, la liste des documents fournis, qu'il s'agisse des phrases pour *TREC* ou des nouvelles journalistiques pour *TDT*, est triée par ordre chronologique. Ainsi, le problème posé consiste plutôt à chercher des nouveaux documents dans le temps. Dans ces conditions, la plupart des systèmes proposent d'identifier un document pertinent en le comparant à ceux qui le précèdent et à ceux qui le suivent dans le corpus, au moyen d'un critère de similarité. Toutefois, dans de nombreuses applications une telle approche n'est pas adéquate car le corpus n'est pas chronologiquement trié ; ce qui est le cas dans le contexte de la veille technologique où il s'agit d'identifier des textes décrivant des innovations technologiques dans des banques de textes scientifiques intégraux. D'autres travaux ont porté sur la détection de thèmes émergents. Ainsi, Bun *et al.* [2] ont proposé un système qui observe les changements intervenant sur un ensemble de sites Web et, qui repère les thèmes émergents à partir des mots figurant dans les pages modifiées. Cependant, en procédant ainsi, ce système ne permet pas de trouver une information inattendue sur un site Web dès la première visite. Matsumura *et al.* [9] ont également développé un système de recherche de thèmes émergents entre des communautés Web. Après avoir construit par classification des communautés composées de membres ayant les mêmes centres d'intérêt, le système analyse et visualise les co-citations entre des pages Web à l'aide de l'algorithme *KeyGraph* [10]. Les thèmes émergents correspondent alors aux pages Web intéressant plusieurs communautés. La faiblesse de ce système est qu'il suppose que de telles communautés puissent être définies et que les pages considérées puissent leur être attribuées. *WebCompare* développé par Liu *et al.* [8] est probablement le système se rapprochant le plus de nos travaux. Il s'agit d'un système destiné à la veille concurrentielle. Après que l'utilisateur ait indiqué les adresses (*URL*) de pages Web de ses concurrents, *WebCompare* est capable de trouver les pages contenant des informations inattendues par rapport à celles figurant sur son propre site. Le caractère inattendu d'une page Web est évalué à l'aide d'une mesure basée sur le paradigme $TF \times IDF$, tout comme les mesures que nous avons proposées dans le système *UnexpectedMiner* que nous avons développé [5]. Ce système vise à extraire, de corpus documentaires, des documents pertinents pour l'utilisateur en ce sens qu'il contiennent des informations inconnues auparavant de celui-ci et se rapportant au sujet qui l'intéresse. Il est bien adapté pour la recherche de signaux faibles dans le cadre de la veille scientifique et technique

La plupart des travaux consacrés à la recherche d'information nouvelle dans des textes, considèrent uniquement le contenu des documents et rares sont ceux qui exploitent également la structure. Or les études récentes menées en recherche d'information [4, 11] ont montré l'intérêt de prendre en compte ces deux types d'information. Dans le cadre de la recherche de nouveauté, la prise en compte de la structure paraît pourtant justifiée car on peut supposer que toutes les parties d'un document n'ont pas le même poids et qu'il en est de même pour les termes apparaissant dans ces parties. De plus, les documents considérés sont souvent fortement structurés. Ceci est particulièrement vrai dans le cadre de la veille scientifique et technique puisqu'il s'agit alors d'articles scientifiques ou de résumés de thèses composés de parties clairement identifiées comme le titre, les auteurs, les mots clés, le résumé, les sections. Il en va de même pour les fiches descriptives de brevets ou encore pour les pages diffusées sur le Web du fait de l'utilisation de langage de description tels que XML. Dans le prolongement de *TREC2003* visant à rechercher des éléments d'information nouveaux dans des phrases pertinentes, Dkaki et Mothe [3] exploitent cette hypothèse mais en accordant une pondération différente à un terme non pas selon la position qu'il occupe dans les phrases susceptibles de répondre à une requête mais seulement dans la requête elle même. De plus, les pondérations sont choisies de façon expérimentale. Dans cet article, nous proposons d'adapter le système *UnexpectedMiner* pour extraire des informations inattendues à partir de textes en exploitant conjointement leur contenu et leur structure. Ceci nous conduit à revoir le mode de représentation des documents, classiquement utilisé en recherche d'information. Ainsi, au lieu d'être représenté sous forme vectorielle, chaque document est représenté sous une forme matricielle qui sera décrite dans la section suivante. La recherche d'information inattendue peut être ensuite réalisée en utilisant des mesures, définies dans la section 3, et qui sont appliquées à chacune des parties du document. La valeur d'inattendu pour l'ensemble du document est égale à une somme pondérée des valeurs obtenues sur chaque partie. La question délicate du choix des pondérations attribuées aux différentes parties est posée en terme d'optimisa-

1. http://www.nist.gov/speech/tests/tdt

2. Le challenge 'novelty detection" est apparu pour la première fois lors de la conférence TREC 2002. Les communications présentées à cette conférence et aux suivantes sont disponibles à l'adresse http://trec.nist.gov

tion et résolue par apprentissage à l'aide d'une méthode de recuit simulé. Dans la section 4, les résultats expérimentaux obtenus sur une base d'articles scientifiques en tenant compte de la structure sont comparés à ceux trouvés en considérant uniquement le contenu des documents.

2 Prise en compte de la structure des documents

La structure d'un document peut être définie comme l'ensemble des parties distinctes le constituant. Deux parties ne peuvent pas se chevaucher ni être incluses l'une dans l'autre ; elles peuvent seulement être consécutives. Comme parties d'un document, on pourra considérer par exemple le titre, les mots clés, l'introduction, la conclusion, les différentes sections du document découpées selon le niveau de granularité choisi. La prise en compte de cette structure pose deux questions délicates qui seront abordées successivement. La première concerne la représentation du document ; la seconde les modalités de calcul du poids à accorder à chacune des parties pour tenir compte de leur importance respective.

2.1 Représentation d'un document

Un des modèles les plus employés en recherche d'information pour représenter un document est le modèle vectoriel introduit par Salton et McGill. Dans ce modèle [12], un index recense tous les mots $t_1, t_2, ..., t_m$ rencontrés dans le corpus de documents. Chaque document d_j est alors représenté par un vecteur de poids $\vec{d_j} = (w_{1,j}, w_{2,j}, ..., w_{m,j})$ où $w_{i,j}$ représente le poids du mot t_i dans le document d_j. Si le mot t_i n'apparaît pas dans le document d_j alors $w_{i,j} = 0$. Pour évaluer le poids d'un mot dans un document la formule $TF \times IDF$ est généralement utilisée. TF (Term Frequency), la fréquence relative du mot t_i dans un document d_j est définie par :

$$tf_{i,j} = \frac{f_{i,j}}{max_h f_{h,j}}$$

où $f_{i,j}$ désigne la fréquence du mot t_i dans le document d_j. Plus le mot t_i est fréquent dans le document d_j, plus $tf_{i,j}$ est élevé.
IDF (Inverse Document Frequency) est une mesure du pouvoir discriminant du mot t_i définie par :

$$idf_i = \log_2 \frac{C}{n_i} + 1$$

où C est la taille du corpus analysé et n_i le nombre de documents contenant le mot t_i. Plus le mot t_i est rare dans l'ensemble des documents, plus idf_i est élevé. Dans la pratique, la fréquence relative d'un mot t_i est calculée plus simplement par :

$$idf_i = \log \frac{C}{n_i}$$

Le poids $w_{i,j}$ d'un mot t_i dans un document d_j est alors obtenu en combinant les deux critères précédents :

$$w_{i,j} = tf_{i,j} \times idf_i$$

Ce poids est d'autant plus élevé que le mot t_i est fréquent dans le document d_j et rare dans les autres documents.
Notre système intègre ce modèle de représentation mais en l'adaptant pour tenir compte de la structure des documents. Nous proposons en effet de calculer le nombre d'occurrences d'un mot sur chacune des parties d'un document plutôt que sur son intégralité. C'est ensuite en effectuant une somme pondérée de ces nombres d'occurrences que le poids final de ce mot pour le document est déterminé.
Ainsi, si dans l'ensemble des documents analysés par le système on distingue k parties différentes, l'importance de chaque partie l est donnée par un coefficient cs_l, que nous appelons *coefficient de structuration* tel que :

$$\sum_{l=1}^{k} cs_l = 1$$

Le poids d'un mot t_i dans un document d_j est calculé de la façon suivante :

$$w_{i,j} = k \sum_{l=1}^{k} cs_l \,.\, tf_{t_i,l} \,.\, idf_i$$

avec

$$tf_{i,l} = \frac{f^l_{i,j}}{max_h f_{h,j}}$$

où $f^l_{i,j}$ désigne la fréquence du mot t_i dans la partie l du document d_j
On peut noter qu'en attribuant à tous les coefficients la même valeur $\frac{1}{k}$, on retrouve le modèle de Salton. Il reste encore à déterminer les valeurs spécifiques des coefficients cs_l à attribuer à chacune des parties pour prendre effectivement en compte la structure. Ces valeurs doivent traduire l'importance relative de chaque partie. Par exemple, pour des documents découpés en seulement trois parties : le titre, les mots clés et le corps du document on peut décider d'attribuer à la première un poids deux fois supérieur à celui des autres parties ce qui reviendrait à prendre pour vecteur de pondération (0.5, 0.25, 0.25) tandis que pour un autre corpus où les mots clés seraient privilégiés on retiendrait plutôt le vecteur (0.25, 0.5, 0.25). En fait, même avec l'aide d'un expert, le choix des coefficients de structuration reste difficile à faire. C'est la raison pour laquelle, nous proposons de les apprendre automatiquement à partir d'un échantillon du corpus de textes considéré.

2.2 Choix des coefficients de structuration

L'objectif est de rechercher, à l'aide d'un échantillon contenant quelques documents dont on sait a priori qu'ils sont inattendus, le vecteur de pondération permettant le mieux au système de retrouver ces documents. Ce problème peut être posé en termes d'optimisation à condition de définir une fonction objectif, permettant de comparer différents vecteurs de pondération et, qu'il conviendra de maximiser. Pour construire une telle fonction, on peut remarquer que pour chaque configuration (*i.e.* vecteur de pondérations) testée, à l'issue du traitement de l'échantillon par le système *UnexpectedMiner*, on obtient la liste des documents extraits classés par ordre d'inattendu. Toutefois, ce classement, peut contenir des documents qui ne sont pas inattendus. Le système a donc commis une erreur chaque fois qu'un tel document apparaît dans la liste. Cependant, l'erreur commise est d'autant plus grave qu'elle apparaît au début de la liste plutôt qu'à la fin. La fonction objectif, que nous avons définie pour quantifier ces erreurs de classement tient justement compte de l'ordre d'apparition des erreurs. Si le système extrait $|E|$ documents, alors, la fonction objectif pour une configuration C_i est définie par :

$$f_{C_i} = \sum_{j=1}^{|E|} (\frac{1}{r} \times [j])$$

où $[j]$ vaut 1 si le document d_j, extrait en $r^{éme}$ position, n'est pas inattendu et 0 sinon. Ainsi, une erreur au premier rang va coûter 1 alors qu'une erreur au dixième rang coûtera seulement 0.1. Plus la valeur de la fonction sera faible, meilleur sera le classement des documents.
En supposant par exemple, que l'on demande au système d'extraire 10 documents inattendus et qu'il commette trois erreurs, le tableau 1 donne la valeur de la fonction de classement en fonction du rang des erreurs commises sur les dix documents retournés dans 4 cas.

Rangs des erreurs	Fonction de classement
8, 9, 10	0.3361
3, 4, 7	0.7262
1, 6, 10	1.2667
1, 2, 3	1.8333

TAB. 1 – *Exemples de valeurs de la fonction de classement*

Trouver la ou les configurations des coefficients de structuration correspondant au meilleur classement des documents inattendus revient donc à minimiser la fonction de classement. Cela nécessite le calcul de la valeur de la fonction de classement pour toutes les configurations possibles de façon à conserver celle qui conduit à la plus faible valeur de la fonction. Un tel calcul n'est toutefois pas envisageable car les coefficients sont à valeurs réelles et il existe par conséquent une infinité de configurations à tester. De plus, même si on limitait l'ensemble des valeurs possibles pour les coefficients, par exemple en fixant un pas entre deux valeurs possibles pour un coefficient, le nombre de configurations resterait encore très élevée y compris lorsque le nombre de parties composant un document est restreinte.
Pour résoudre ce problème d'optimisation, nous avons choisi de nous orienter vers une approche heuristique en utilisant le recuit simulé [7]. Le principe de cette méthode consiste, à partir d'une configuration initiale, à lui faire subir une perturbation, c'est à dire une modification d'amplitude limitée. Si cette perturbation a pour effet d'améliorer la fonction de classement, cette nouvelle configuration est retenue. Si au contraire, elle provoque une dégradation de la fonction de classement, alors cette nouvelle configuration est retenue avec une probabilité p dépendant d'une combinaison de l'importance de la dégradation et du degré d'avancement dans la recherche de la meilleure solution. De la sorte, au début du processus on accepte plus volontiers une dégradation importante, tandis que plus on est proche de la meilleure solution, plus on sera réticent à accepter une dégradation même minime. Cette probabilité est calculée en s'inspirant de la distribution de Gibbs-Boltzmann. Pour passer d'une solution C_i à une solution C_j, on calcule la variation $\Delta f_{ij} = f_{Cj} - f_{Ci}$ de la fonction objectif et la probabilité $P(i,j)$ d'accepter la transformation est alors définie par:

$$\begin{cases} P(i,j) = 1 & si\ \Delta f_{ij} \leq 0 \\ P(i,j) = \exp^{-\frac{\Delta f_{ij}}{t}} & si\ \Delta f_{ij} > 0 \end{cases}$$

t est un paramètre de contrôle égal à kT où T est la température et k la constante de Boltzmann vaut 6.18E-23. Ce choix de probabilité convient parfaitement aux conditions précédemment évoquées à savoir que si la température T est élevée, un grand nombre de transformations seront acceptées alors que quand elle sera basse, seules les transformations améliorant la solution seront retenues. Pour savoir si la nouvelle solution est acceptée, un nombre est tiré aléatoirement entre 0 et 1 puis comparé à $P(i,j)$. Si $P(i,j)$ est plus petite, la solution est acceptée, sinon elle est rejetée. Le processus est itéré soit avec la nouvelle configuration, si elle est retenue, soit avec l'ancienne dans le cas contraire. Tout au long du recuit simulé, la meilleure solution rencontrée est conservée en mémoire.
La décroissance de la température s'effectue par paliers et pour chacun de ces paliers, un nombre donné de transformations sont essayées. Ainsi, chacune des nouvelles solutions testées appartient au voisinage de la solution actuelle et l'espace des solutions est parcouru sans pour autant toutes les essayer.
Dans cet algorithme, le voisinage d'une solution ainsi

que la valeur de décroissance de la température doivent être fixés par l'utilisateur. Le voisinage est défini en utilisant un *pas* entre des valeurs successives possibles pour un coefficient ; ce qui revient à fixer un intervalle de variation du coefficient de chaque partie. Chacun des coefficients pourra ainsi évoluer dans cet intervalle lors d'une transformation ; la seule condition étant que la somme des coefficients soit toujours égale à 1. Par exemple si la configuration actuelle est 0.05 ; 0.37 ; 0.58 pour un document comportant 3 parties, on autorisera une variation de chacun des poids allant jusqu'à $\pm$ 0.02. Une configuration dans le voisinage pourra être 0.06 ; 0.39 ; 0.55 ou encore 0.03 ; 0.38 ; 0.59.

La température initiale quant à elle, doit être suffisamment importante pour qu'au début du processus d'optimisation un grand nombre de transformations soient acceptées. Kirkpatrick [7] propose de choisir une température t puis d'essayer un certain nombre de transformations et de calculer le taux de transformations acceptées. Si celui-ci est d'au moins 80%, on garde t sinon on double sa valeur et on recommence.

La fonction de décroissance de la température g est souvent une fonction géométrique $g = \mu t$ avec $0 < \mu < 1$. Si on choisi une valeur éloignée de 1, on risque de décroître trop rapidement et de se rapprocher de la trempe. Un choix judicieux proposé par Kirkpatrick se situe entre 0.85 et 0.95.

Le nombre de paliers de température dépend de la fonction de décroissance. Il doit être choisi de telle sorte qu'à la fin de l'algorithme, la température soit assez basse pour que pratiquement aucune transformation ne soit acceptée. La valeur de t doit se situer environ à 5% de sa valeur initiale, d'après Kirkpatrick. Pour le nombre de solutions testées à un palier de température, la manière la plus simple est de choisir un nombre proportionnel au nombre de solutions du problème. Cependant, il convient de veiller à ce que ce nombre multiplié par le nombre de paliers ne conduise pas à tester plus de cas que le nombre total de solutions du problème.

3 Extraction de documents inattendus

3.1 Architecture du système

Le prétraitement visant à mettre les documents sous la forme matricielle, décrite précédemment, est réalisé par le premier module du système *UnexpectedMiner* dont l'architecture est présentée dans la figure 1.

Ce système est articulé autour de trois modules et il requiert deux ensemble de documents. Le premier est un ensemble de documents de référence, noté R, fourni par l'utilisateur et qui permet de cibler le sujet qui l'intéresse. Dans la pratique entre dix et vingt documents doivent suffire. Le second, noté N, désigne l'ensemble des nouveaux documents, issus de différents corpus, susceptibles de contenir des informations inattendues et intéressantes pour l'utilisateur. Les deux ensembles de documents vont subir un prétraitement comportant différentes opérations classiques telles que l'élimination des éléments non pertinents, une analyse morphologique et la suppression des mots vides. A l'issue de ce prétraitement chaque document est représenté sous une forme matricielle. Le but du second module est d'extraire de la base N des nouveaux documents, ceux qui sont les plus similaires aux documents de référence R, à l'aide de la distance du *cosinus*. Enfin, dans cet ensemble S de nouveaux documents jugés similaires, le troisième module, vise à rechercher ceux qui contiennent des informations inattendues par rapport à celles figurant dans les documents de référence et dans les documents de S sélectionnés à l'étape précédente. Le caractère inattendu d'un document est évalué à l'aide de nouvelles mesures que nous avons proposées lors du développement de la première version de *UnexpectedMiner* ne tenant pas compte de la structure des documents [6].

3.2 Recherche des documents similaires

Le but du second module du système *UnexpectedMiner* est d'extraire de la base N de nouveaux documents, ceux qui sont le plus similaires aux documents de référence R fournis par le veilleur. La similarité s_{jk} entre un nouveau document $d_j \in N$ et un document de référence $d_k \in R$ est égale à la distance du *cosinus*, couramment employée dans les systèmes de recherche d'information. Elle est égale au cosinus de l'angle formé par les vecteurs représentant ces documents :

$$s_{jk} = \frac{\vec{d_j} \bullet \vec{d_k}}{|\vec{j}| \times |\vec{k}|}$$

où

$$\vec{d_j} \bullet \vec{d_k} = \sum_i w_{i,j} \times w_{i,k}$$

$$|\vec{j}| = \sqrt{\sum_{i=1,m} w_{i,j}^2}$$

La similarité moyenne s_j du nouveau document $d_j \in N$ avec l'ensemble des documents de référence R est égale à :

$$s_j = \frac{1}{|R|} \sum_{k=1}^{|R|} s_{jk}$$

où $|R|$ désigne le nombre de documents de référence.

Après avoir classé par ordre décroissant de similarité moyenne les nouveaux documents, un sous ensemble S est extrait de N. Il est composé des nouveaux documents les plus proches de ceux fournis comme référence par le veilleur.

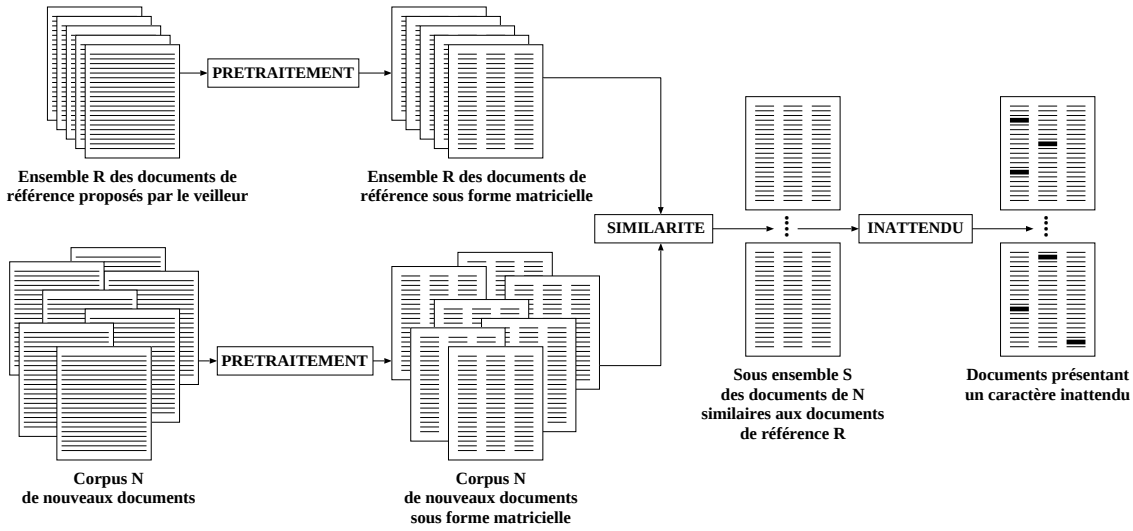

FIG. 1 – *Architecture du système UnexpectedMiner*

3.3 Recherche de documents inattendus

L'objectif du troisième module du système *UnexpectedMiner* est de rechercher les documents de S contenant des informations inattendues par rapport à celles contenues non seulement dans les documents de référence (R) mais aussi dans les documents de S sélectionnés à l'étape précédente. En effet, un document sera très inattendu si les thèmes qu'il aborde ne sont présents ni dans un autre document de S ni dans un document de R. Pour évaluer le caractère inattendu d'un document d_j par rapport aux autres documents appartenant à $R \cup S - \{d_j\}$, nous avons comparé la mesure introduite par Liu dans le système WebCompare [8] avec deux mesures performantes que nous avions proposées dans [6].

Mesure tenant compte de la fréquence des mots. La mesure développée par Liu, Ma et Yu [8] pour repérer des pages inattendues dans un site WEB est définie par :

$$M1(d_j) = \frac{\sum_{i=1}^{m} U^1_{i,j,c}}{m}$$

avec:

$$U^1_{i,j,c} = \begin{cases} 1 - \frac{tf_{i,c}}{tf_{i,j}} & si\ tf_{i,c}/tf_{i,j} \leq 1 \\ 0 & sinon \end{cases}$$

où d_j désigne un document de S et D_c le document obtenu en combinant tous les documents de référence de R avec les documents sélectionnés sauf d_j : $R \cup S - \{d_j\}$.

Mesure tenant compte du pouvoir discriminant des mots. La seconde mesure, que nous avons proposée, fait intervenir directement le pouvoir discriminant idf_i d'un mot t_i puisqu'elle évalue le caractère inattendu d'un document d_j par la somme des poids $w_{i,j}$ des mots t_i qui le représentent :

$$M2(d_j) = \sum_{i=1}^{m} w_{i,j}$$

3.4 Mesure tenant compte du poids maximum

La troisième mesure, quant à elle, attribue comme valeur d'inattendu à un document d_j le poids le plus élevé apparu dans son vecteur de représentation :

$$M3(d_j) = max_l w_{l,j}$$

L'intérêt de ces deux dernières mesures est qu'elles intègrent les coefficients de structuration déterminés par le processus d'optimisation.

Des tests ont été réalisés pour évaluer le système *UnexpectedMiner* en tenant compte de la structure des documents et comparer ces différentes mesures. Ils sont présentés dans la section suivante.

4 Expérimentations

Nous avons réalisé dans un premier temps uniquement l'évaluation du module d'extraction d'information inattendue puis dans un second temps celle du système global pour dissocier les erreurs commises dans la recherche des documents similaires de celles réalisées lors de la recherche des documents inattendus. En effet, si le module d'extraction de documents similaires était parfait, il fournirait un ensemble S comportant uniquement des documents intéressant le veilleur et susceptibles de contenir des informations inattendues. Toutefois, ce module n'est jamais totalement parfait et

les erreurs qu'il peut commettre constituent du bruit rendant plus difficile la tâche d'extraction d'informations inattendues. Ces documents, jugés à tort comme similaires par le module 2, auront de fortes chances d'être considérés ensuite par le module 3 comme étant inattendus, alors qu'en fait ils sont hors sujet et sans intérêt pour le veilleur.
Finalement, nous comparons les performances du système *UnexpectedMiner* dans sa version ne prenant pas en compte la structure et dans celle la prenant en compte.

4.1 Corpus et critères d'évaluation utilisés

Dans ces expérimentations, l'ensemble de référence R est composé de 18 articles scientifiques en anglais consacrés à l'apprentissage automatique, tandis que la base N contient 223 nouveaux documents dont 40 seulement se rapportent à ce sujet et 18 parmi ces 40 sont inattendus d'après le veilleur. Les 183 autres documents, en dehors du sujet, sont des articles de biologie, de chimie ou encore d'informatique mais ne traitant pas d'apprentissage automatique.
Les critères d'évaluation du système *UnexpectedMiner* que nous avons utilisés sont ceux communément employés en recherche d'information : les taux de précision et de rappel définis par J.A. Swets [14]. La *précision* indique le pourcentage de documents extraits par le système et qui ont réellement un caractère inattendu. Le *rappel* mesure quant à lui le pourcentage de documents ayant un caractère inattendu retrouvés dans le corpus de nouveaux documents N par le système. De plus, la précision et le rappel sont calculés d'abord en demandant au système d'extraire le document le plus inattendu, puis en lui demandant d'en extraire deux puis trois et ainsi de suite. De la sorte, il est possible de tracer des courbes de précision et de rappel où figurent en abscisse le nombre de documents demandés au système et en ordonnée la précision et le rappel correspondant. De plus, nous avons également réutilisé la fonction objectif employée dans le processus d'optimisation.

4.2 Evaluation des mesures d'inattendu

Nous avons d'abord évalué le système sans faire intervenir le module de recherche de documents similaires ; ce qui revient à prendre comme ensemble S les 40 documents se rapportant à l'apprentissage automatique. Les valeurs de la fonction de classement obtenues pour chaque mesure figurent dans le tableau 2. La valeur de la fonction de classement étant d'autant plus élevée que les erreurs sont commises rapidement, il semble que ce soit la mesure $M3$ tenant compte du poids maximum qui fournisse en priorité des documents inattendus tandis que la mesure $M2$ tenant compte du pouvoir discriminant, et plus encore la mesure $M1$ proposée par Liu semblent commettre des erreurs dès les premières réponses. Ceci est confirmé par les résultats obtenus en terme de précision et de rappel, et présentés dans les figures 2, 3 et 4 où l'axe des abscisses indique le nombre de documents inattendus demandés au système. On remarque en effet, que si l'utilisateur demande au système un document inattendu, seule la mesure $M1$ de Liu ne parvient pas à en extraire un puisque la précision vaut 0% (figure 2) alors qu'elle atteint 100% pour les deux autres mesures (figure 3 et 4). Il faut au moins exiger cinq documents pour que

Mesure	Valeur de la fonction de classement
1	2.07
2	0.92
3	0.59

TAB. 2 – *Valeurs de la fonction de classement pour le module d'inattendu*

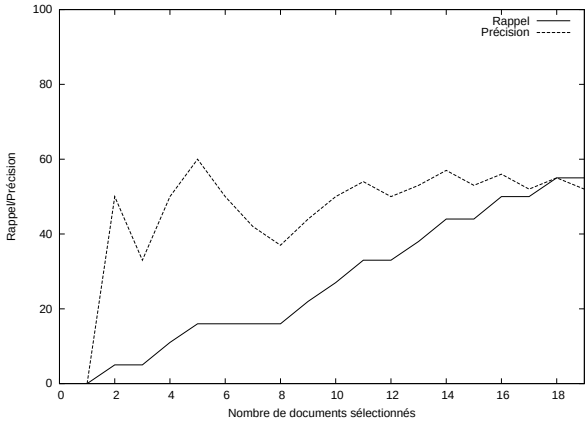

FIG. 2 – *Evaluation du module d'inattendu - Précision et rappel pour la mesure 1.*

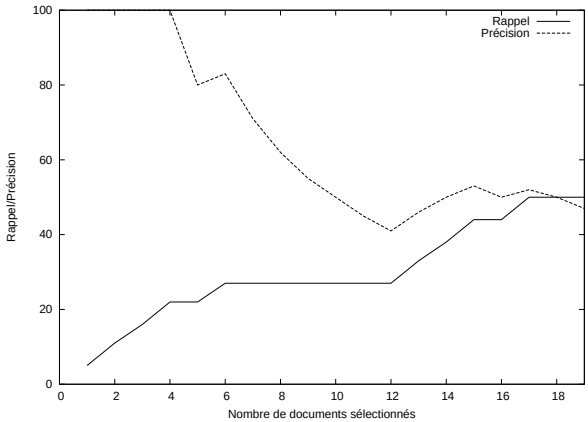

FIG. 3 – *Evaluation du module d'inattendu - Précision et rappel pour la mesure 2.*

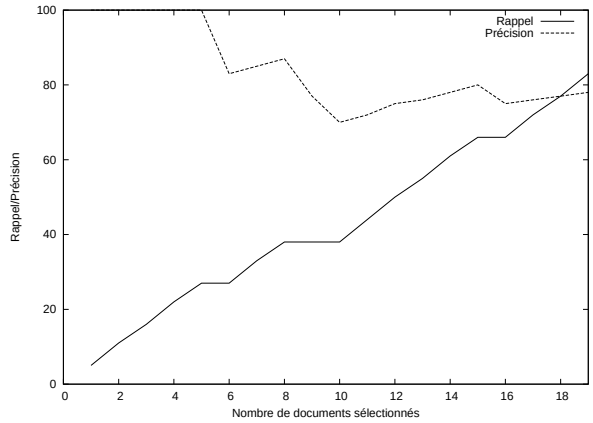

FIG. 4 – *Evaluation du module d'inattendu - Précision et rappel pour la mesure 3.*

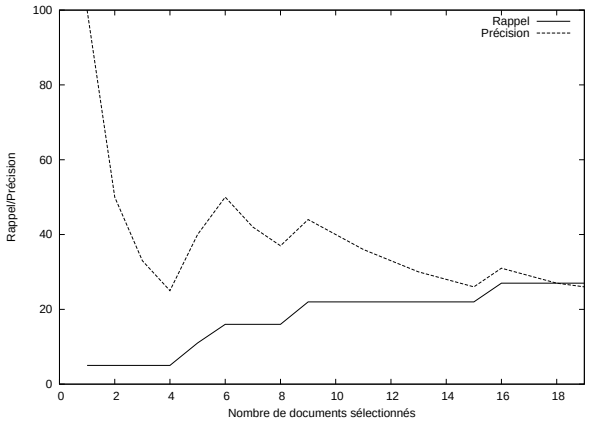

FIG. 6 – *Evaluation du système global - Précision et rappel pour la mesure 2*

la mesure $M2$ tenant compte du pouvoir discriminant commette une première erreur et six documents pour la mesure $M3$ tenant compte du poids maximum. C'est d'ailleurs, cette dernière mesure qui extrait le mieux les documents inattendus, quel que soit le nombre demandé par l'utilisateur.

4.3 Evaluation du système global

Le système complet a ensuite été évalué en procédant non seulement à la recherche des documents inattendus mais aussi, au préalable, à la détermination de l'ensemble S des documents similaires. L'ensemble N contient alors 223 nouveaux documents dont seulement 40 se rapportent au sujet de la veille et devraient se retrouver dans l'ensemble S si le module de recherche des documents similaires ne faisait aucune erreur. Or, à l'issue de ce module, parmi les 40 docu-

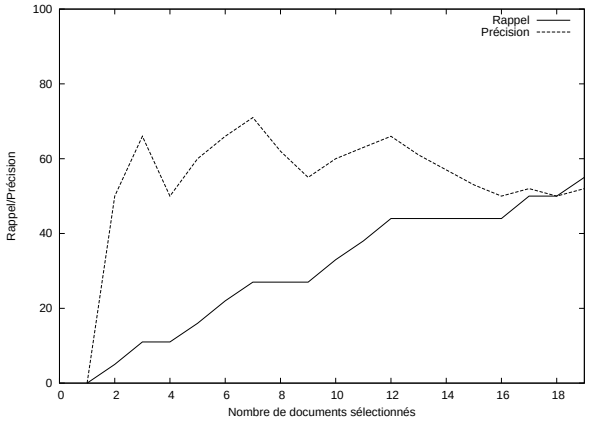

FIG. 7 – *Evaluation du système global - Précision et rappel pour la mesure 3*

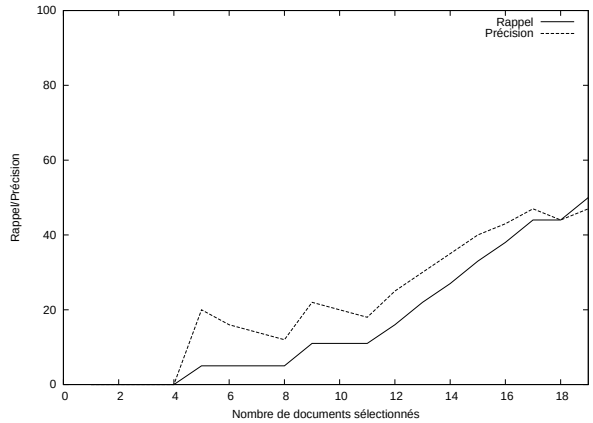

FIG. 5 – *Evaluation du système global - Précision et rappel pour la mesure 1*

ments jugés similaires aux documents de référence par le système, seuls 35 le sont réellement. De plus, parmi ces 35 documents similaires il n'y en a plus que 12 qui sont inattendus. La mise en oeuvre du module de similarité entraîne donc une dégradation importante des performances puisque 6 documents inattendus ont été jugés à tort hors sujet et écartés avant même que soit mis en oeuvre le module d'extraction d'information inattendue. Cette dégradation apparaît dans le tableau 3 où les valeurs de la fonction score pour les trois mesures sont supérieures aux valeurs obtenues sans mettre en oeuvre le module de similarité. Cette dégradation est confirmée, pour les trois mesures, par les résultats obtenus en termes de précision et de rappel. De plus, dans cette nouvelle expérimentation, c'est

Mesure	Valeur de la fonction de classement
1	2.71
2	1.90
3	1.76

TAB. 3 – *Valeurs de la fonction de classement pour le système complet*

encore la mesure $M1$ de Liu qui détecte le moins bien les documents inattendus puisqu'elle restitue à tort les quatre premiers documents (figure 5), tandis que sur ces quatre premiers documents, la mesure $M2$ tenant compte du pouvoir discriminant commet trois erreurs (figure 6) et la mesure $M3$ du poids maximum deux erreurs (figure 7). Globalement c'est encore cette dernière mesure qui fournit les meilleurs résultats.

4.4 Evaluation de l'apport de la Structure

Dans cette dernière expérimentation nous évaluons l'apport de la prise en compte de la structure des documents dans la qualité des résultats du système *UnexpectedMiner*. Comme pour la première expérimentation, nous ne considérons ici que le module d'inattendu et donc une nouvelle fois l'ensemble S est constitué des 40 documents se rapportant à l'apprentissage automatique. Les courbes 8 et 9 permettent de comparer, pour la mesure $M2$, les résultats du système ne prenant pas en compte la structure des documents avec ceux du système la prenant en compte. On peut constater que, sans tenir compte de la structure, le système produit une erreur dès le deuxième document retourné alors qu'en tenant compte de la structure, le système ne produit une erreur qu'à partir du cinquième document retourné.

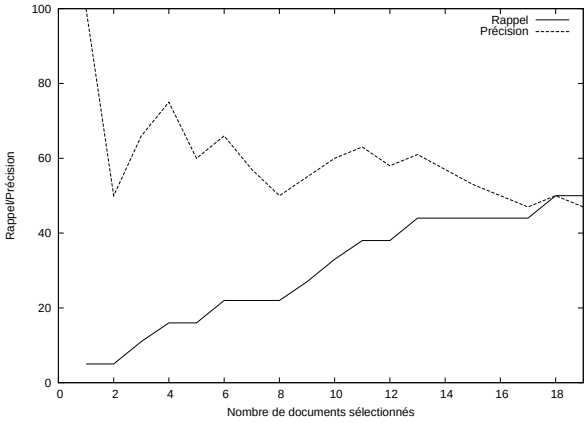

FIG. 8 – *Précision et rappel pour la mesure 2 sans prise en compte de la structure*

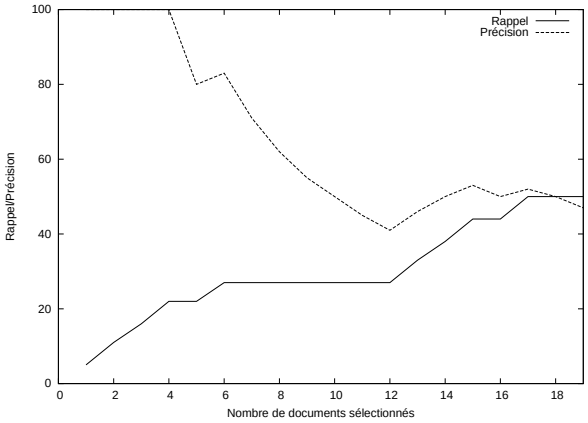

FIG. 9 – *Précision et rappel pour la mesure 2 avec prise en compte de la structure*

En ce qui concerne la mesure $M3$, comme on peut le constater sur les figures 10 et 11, les résultats sont similaires.

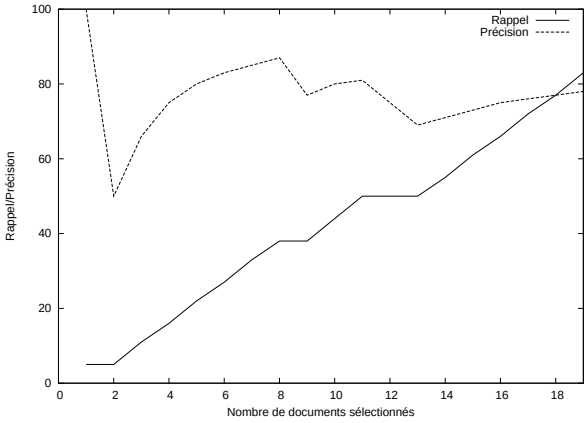

FIG. 10 – *Précision et rappel pour la mesure 3 sans prise en compte de la structure*

Sans prendre en compte la structure le système fait une erreur dès le deuxième document retourné alors qu'en tenant compte de la structure, le système n'effectue une erreur qu'à partir du septième document retourné ce qui constitue une amélioration considérable.

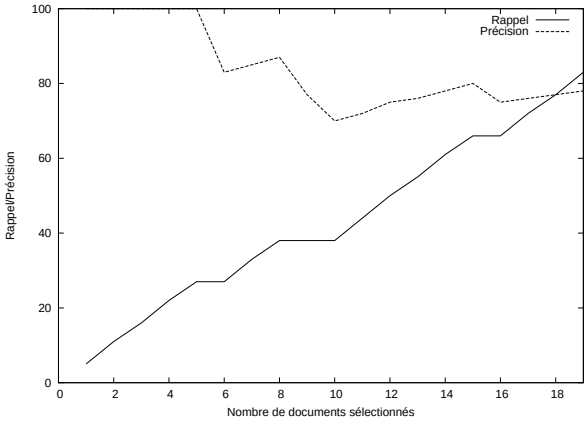

FIG. 11 – *Précision et rappel pour la mesure 3 avec prise en compte de la structure*

5 Conclusion et perspectives

Dans cet article nous avons montré comment il est possible de prendre en compte la structure des documents au sein d'un système de découverte d'informations inattendues. Nous avons étendu le système *UnexpectedMiner* en ce sens en affectant des coefficients de structuration à chacune des parties des documents. Ces coefficients sont ensuite utilisés pour pondérer l'importance que l'on souhaite donner à ces diverses parties dans les mesures d'inattendu utilisées par le système. Nous avons utilisé un algorithme de recuit simulé pour déterminer au mieux les valeurs de chacun de ces coefficients. Les expérimentations nous ont permis de montrer que la prise en compte de la structure des documents permettait d'améliorer significativement les performances du système *UnexpectedMiner*.

Comme nous avons pu le constater, il sera nécessaire dans le futur, pour améliorer l'efficacité du système global, de revoir le module de recherche de documents similaires. Actuellement celui-ci repose sur l'utilisation classique des formules $TF \times IDF$, mais la mise en place de techniques plus evoluées, c'est-à-dire plus en phase avec l'état de l'art actuel du domaine de la recherche d'information, devra être envisagée.

Toutefois, notre intérêt premier devra se porter sur le module de recherche d'informations inattendues, qui constitue plus le coeur du système. Dans ce cadre, la détermination automatique des coefficients de structuration est une voie de recherche intéressante en vue d'améliorer son efficacité.

Références

[1] J. Allan, J. Carbonell, G. Doddington, J. Yamron, and Y. Yang. Topic detection and tracking pilot study: Final report. In *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, pages 194–218, 1998.

[2] K. K. Bun and M. Ishizuka. Emerging topic tracking system. In *Proceedings of the 1st International Conference on Web Intelligence*, LNCS 2198, pages 125–130, 2001.

[3] T. Dkaki and J. Mothe. Recherche de la pertinence et de la nouveauté dans les textes. In *Actes de la Conférence en Recherche d'Information et Applications (CORIA)*, pages 229–245, 2004.

[4] F. Franck. *Modélisation, indexation et recherche de documents structurés.* PhD thesis, Université Joseph Fourier Grenoble I, France, 1998.

[5] F. Jacquenet and C. Largeron. Discovering Unexpected Information for Technology Watch. In *Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, LNCS 3202, pages 219–230, 2004.

[6] F. Jacquenet and C. Largeron. Extraction automatique d'information inattendue à partir de textes - accepté pour publication. *Revue des Nouvelles Technologies de l'information numéro spécial Fouille de données complexes*, 2005.

[7] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. In *Sciences*, pages 671–680, 1983.

[8] B. Liu, Y. Ma, and P. S. Yu. Discovering unexpected information from your competitors' web sites. In *Proceedings of the 7th ACM SIGKDD international conference on Knowledge Discovery and Data mining*, pages 144–153, 2001.

[9] N. Matsumura, Y. Ohsawa, and M. Ishizuka. Discovery of emerging topics between communities on WWW. In *Proceedings of the 1st International Conference on Web Intelligence*, LNCS 2198, pages 473–482, 2001.

[10] Y. Ohsawa, N. E. Benson, and M. Yachida. Keygraph: Automatic indexing by co-occurrence graph based on building construction metaphor. In *Proceedings of the Advances in Digital Libraries Conference*, pages 12–18, 1998.

[11] B. Piwowarski. *Techniques d'apprentissage pour le traitement d'informations structurées : application à la recherche d'information.* PhD thesis, Université Pierre et Marie Curie, France, 2003.

[12] G. Salton and M. J. McGill. Introduction to modern information retrieval. In *McGraw-Hill*, 1983.

[13] I. Soboroff and D. Harman. Overview of the trec 2003 novelty track. In *NIST Special Publication:SP 500-255*, pages 38–53. The Twelfth Text Retrieval Conference (TREC 2003), 2003.

[14] J.A. Swets. Information retrieval systems. *Science*, 141:245–250, 1963.

[15] C. Wayne. Topic detection and tracking (tdt) overview and perspective. http://www.nist.gov/speech/publications/darpa98/html/tdt10/tdt10.htm, 1998.