Vectorisation, Okapi et calcul de similarité pour le TAL : pour oublier enfin le TF-IDF

Vincent Claveau

Vectorisation, Okapi et calcul de similarité pour le TAL : pour oublier enfin le TF-IDF

Vincent Claveau
IRISA-CNRS
Campus de Beaulieu, 35042 Rennes, France
vincent.claveau@irisa.fr

RÉSUMÉ

Dans cette prise de position, nous nous intéressons au calcul de similarité (ou distances) entre textes, problématique présente dans de nombreuses tâches de TAL. Nous nous efforçons de montrer que ce qui n'est souvent qu'un composant dans des systèmes plus complexes est parfois négligé et des solutions sous-optimales sont employées. Ainsi, le calcul de similarité par TF-IDF/cosinus est souvent présenté comme « état-de-l'art », alors que des alternatives souvent plus performantes sont employées couramment dans le domaine de la Recherche d'Information (RI). Au travers de quelques expériences concernant plusieurs tâches, nous montrons combien ce simple calcul de similarité peut influencer les performances d'un système. Nous considérons plus particulièrement deux alternatives. La première est le schéma de pondération Okapi-BM25, bien connu en RI et directement interchangeable avec le TF-IDF. L'autre, la vectorisation, est une technique de calcul de similarité que nous avons développée et qui offrent d'intéressantes propriétés.

ABSTRACT

Vectorization, Okapi and computing similarity for NLP : say goodbye to TF-IDF

In this position paper, we review a problem very common for many NLP tasks: computing similarity (or distances) between texts. We aim at showing that what is often considered as a small component in a broader complex system is very often overlooked, leading to the use of sub-optimal solutions. Indeed, computing similarity with TF-IDF weighting and cosine is often presented as "state-of-theart", while more effective alternatives are in the Information Retrieval (IR) community. Through some experiments on several tasks, we show how this simple calculation of similarity can influence system performance. We consider two particular alternatives. The first is the weighting scheme Okapi-BM25, well known in IR and directly interchangeable with TF-IDF. The other, called vectorization, is a technique for calculating text similarities that we have developed which offers some interesting properties.

MOTS-CLÉS : Calcul de similarité, modèle vectoriel, TF-IDF, Okapi BM-25, vectorisation.

KEYWORDS: Calculating similarities, vector space model, TF-IDF, Okapi BM-25, vectorization.

1 Introduction

« Okapi, qu'est ce que c'est ? », « Quelle est la différence entre un lemme et un *stem* (racine) ? » Voici deux questions posées à plusieurs reprises dans des conférences de Traitement Automatique des Langues (TAL) pour la première et de Recherche d'Information (RI) pour la seconde. Elles illustrent le relatif cloisonnement des deux communautés concernées, bien que RI et TAL partagent un grand nombre de problématiques, et le langage (écrit ou oral) comme matériau de base. Dans cette prise de position, nous nous intéressons indirectement à la première question et donc à l'une des conséquences de la méconnaissance de certaines techniques de RI dans les applications du TAL. Précisons que le propos n'est pas ici de dresser un état de l'art complet sur les points de rencontre entre TAL et RI (pour un panorama de l'utilisation du TAL en RI, voir par exemple Moreau et Sébillot (2005)).Nous souhaitons simplement mettre en évidence l'importance du calcul de distance (ou de similarité), problématique largement traitée en RI, dans ces applications du TAL[1].

Au travers de quelques travaux, nous montrons dans cet article que la méthode de calcul de similarité, à système identique par ailleurs, influence grandement les résultats. Plus particulièrement, le but de cette communication est de faire valoir quatre points :

1. au sein d'un système de TAL nécessitant des calculs de similarité, le choix de la méthode de calcul doit être soigneusement étudié car il peut changer drastiquement les performances d'un système ;

2. la similarité basée sur le TF-IDF/cosinus n'est pas « état-de-l'art », comme on le lit trop souvent ;

3. des alternatives bien établies et tout aussi simples (par exemple Okapi) donnent de meilleurs résultats en général ;

4. enfin, des propositions récentes, notamment l'approche par vectorisation que nous avons développée, en donnent encore de meilleurs.

Cette position peut paraître d'autant plus facile à prendre qu'elle semble consensuelle. Pour autant, il est frappant de constater combien ces différents points sont loin d'être ancrés dans la communauté : par exemple, dans les actes des quatre dernières éditions de la conférence TALN, la pondération TF-IDF est citée dans 31 articles, contre 4 fois pour Okapi (principalement par les mêmes auteurs de plus).

Cet article est organisé de la manière suivante. La section 2 propose quelques rappels et considérations sur les calculs de distance entre textes, en détaillant notamment les approches TF-IDF et Okapi. Nous présentons ensuite en section 3 une méthode de calcul de distance aux propriétés intéressantes que nous avons développée et utilisée dans plusieurs tâches. Les trois sections suivantes présentent de manière concise trois applications (des références détaillant les approches sont données) dans lesquelles nous rapportons les performances au regard du choix du calcul de similarité.

[1]Ce travail a été réalisé dans le cadre du programme Quæro (`http://www.quaero.org`), financé par OSEO, agence nationale de valorisation de la recherche.

2 Calcul de distance et modèle vectoriel

2.1 Représentation vectorielle et sacs de mots

Dés lors que l'on manipule des données (des textes dans notre cas), il est souvent possible de les décrire avec un ensemble (fini et fixe) de valeurs. Ces valeurs, et donc les objets qu'elles représentent, peuvent alors être interprétées comme des vecteurs formant un espace vectoriel. L'avantage de cette représentation est que l'on sait faire certaines opérations assez facilement dans de tels espaces, notamment des calculs de distance/similarité très rapides.

Dans le cas des textes, ces représentations consistent souvent à considérer le document (ou n'importe quelle donnée textuelle) comme un sac-de-mots, c'est-à-dire un ensemble non structuré, sans information sur la séquentialité des mots dans le texte. Usuellement, on calcule pour chaque mot présent dans le document une valeur reflétant son importance comme descripteur du document (cf. schéma de pondération ci-après). Les mots du vocabulaire (ou de la collection de documents traitée) absents du document ont une valeur nulle. Finalement, le texte est donc décrit comme un vecteur d'un espace ayant pour dimensions tous les mots du vocabulaire. Ces espaces sont donc très grands (par exemple $\mathbb{R}^{100\,000}$ pour une collection moyenne monolingue), mais les vecteurs sont aussi très creux, ce qui permet une représentation compacte, mais surtout de rendre certains calculs très rapides (cf. sous-section suivante).

Cette représentation du texte est très utilisée, avec différentes variantes (sac-de-ngrams ou plus généralement sac-de-*features*...). Cela s'explique par sa simplicité de mise en œuvre, ses manipulations efficaces, et bien que pauvre (elle ne nécessite aucun traitement complexe ou de données externes), elle donne souvent de bons résultats en pratique.

2.2 Distances dans les espaces vectoriels

Dans le modèle vectoriel (Salton *et al.*, 1975), les documents et les requêtes sont représentés par des vecteurs. L'appariement entre un document et une requête se fait en calculant la proximité de leur vecteur, le plus souvent en utilisant une distance de type Lp. Pour deux documents d et q (ou un document et une requête), la distance Lp est définie comme suit (on note V le vocabulaire de la collection, c'est-à-dire, l'ensemble des termes d'indexation de tous les documents) :

$$\delta_{Lp}(q,d) = \sqrt[p]{\sum_{t \in V} |q_t - d_t|^p}$$

Les valeurs les plus courantes sont $p = 1$ (distance L1, dite *manhattan* ou *city-block*), $p = 2$ (distance L2 ou euclidienne), ou $p \to \infty$ (distance de Chebyshev) ; p n'est pas nécessairement un entier mais doit être supérieur à 1 pour que soit respectée l'inégalité triangulaire. Rappelons que la distance basée sur le cosinus habituellement utilisée en RI textuelle est équivalente (i.e. produit le même ordonnancement des documents) à la distance L2 lorsque les vecteurs sont normalisés : $\delta_{L2}(q,d) = \sqrt{2 - 2 * \delta_{cos}(q,d)}$

Les distances Lp avec p pair, et donc la distance L2 et le cosinus en particulier, ont l'avantage d'être rapides à calculer quand les vecteurs sont normés (sous Lp) et creux. En effet, seules

les composantes des vecteurs qui sont non nulles à la fois pour la requête et le document interviennent dans le calcul. Dans le modèle vectoriel standard, cela revient à ne s'intéresser qu'aux mots partagés par la requête et le document puisque les termes absents des documents (ou de la requête) ont une pondération nulle. Cela explique les mise en œuvre par fichiers inversés et la grande rapidité de cette étape d'appariement, notamment lorsque l'on manipule des requêtes de quelques mots.

Ce calcul de distance devient beaucoup plus coûteux, dans le cas où les vecteurs sont de grandes dimensions mais ne sont pas creux. Mais il existe des techniques de calcul rapide des distances dans les espaces vectoriels. De manière générale, ces techniques troquent un fort gain en temps de réponse contre une légère perte de précision, les éléments retournés étant simplement similaires et non *les plus* similaires. Une approche est de découper l'espace des données en portions et de n'effectuer des recherches que sur une ou plusieurs portions (Stein, 2007; Datar *et al.*, 2004) et/ou de calculer des approximations de la distance réelle (Lejsek *et al.*, 2008).

2.3 Pondérations

Les pondérations sont une des caractéristiques majeures du modèle vectoriel introduit par Salton (1975). Elles permettent de caractériser non seulement la présence ou l'absence de termes dans les documents, mais également leur importance relative pour décrire le contenu du document : un poids w_{ij} est attribué à chaque terme t_i du document d_j ; plus ce poids est important, plus t_i est considéré comme pertinent pour décrire d_j.

Le célèbre TF-IDF décompose la pertinence d'un terme selon deux heuristiques. La première est le TF, issue des travaux de Luhn (1958) : plus le terme est fréquent dans le document, plus il est jugé pertinent. La seconde, l'IDF, est souvent attribuée à Spärck Jones (1972) : plus un terme apparaît dans un grand nombre de document, moins il est pertinent. Sa formulation la plus connue est (tf est le nombre d'occurrence ou la fréquence du terme t dans le document considéré, df sa fréquence documentaire, c'est-à-dire le nombre de documents dans lequel il apparaît, N est le nombre total de documents) :

$$w_{TF-IDF}(t,d) = tf(t,d) * \log(N/df(t))$$

Outre le TF-IDF/cosinus, beaucoup de techniques (pondérations et similarités) pour calculer des similarités dans des espaces algébriques existent (Tirilly, 2010, pour une revue). Parmi celles-ci, la pondération Okapi[2] est devenue une référence grâce aux très bons résultats qu'elle permet d'obtenir sur de nombreuses tâches de RI. Cette pondération a initialement été proposée comme modèle de similarité dans un cadre probabiliste (Robertson *et al.*, 1998) ; ce cadre repose sur le principe de classement probabiliste (PRP, *Probability Ranking Principle* que Robertson énonce ainsi : *If retrieved documents are ordered by decreasing probability of relevance on the data available, then the system's effectiveness is the best to be gotten for the data.*

Ce principe généraliste est en pratique décliné en modèles pouvant s'interpréter comme des pondérations dans un modèle vectoriel. Le modèle Okapi peut ainsi être vu comme un

[2]La formule de cette pondération s'appelle en réalité BM-25, mais est souvent appelée Okapi du nom du premier système l'ayant implémenté.

TF-IDF prenant mieux en compte la longueur des documents. Sa définition est donnée dans l'équation 1 qui indique le poids du terme t dans le document d ($k_1 = 2$ and $b = 0.75$ sont des constantes, dl la longueur du document, dl_{avg} la longueur moyenne des documents).

$$\begin{aligned} w_{BM25}(t,d) &= TF_{BM25}(t,d) * IDF_{BM25}(t) \\ &= \frac{tf(t,d) * (k_1+1)}{tf(t,d) + k_1 * (1 - b + b * dl(d)/dl_{avg})} * \log \frac{N - df(t) + 0.5}{df(t) + 0.5} \quad , \end{aligned} \tag{1}$$

La partie TF_{BM25} est dérivée d'un modèle probabiliste de la fréquence des termes dans les documents, le modèle 2-Poisson de Harter (Spärck Jones *et al.*, 2000). Ce modèle représente la distribution des termes dans les documents comme un mélange de deux distributions de Poisson : l'une représentant la fréquence des termes pertinents pour décrire le document, l'autre celle des termes non-pertinents (Harter, 1975). C'est dans ce TF qu'est intégré une normalisation en fonction de la taille du document.
La partie IDF_{BM25} est une simplification d'une formule dérivée du PRP (Spärck Jones *et al.*, 2000), théoriquement optimale mais nécessitant des données d'apprentissage. L'IDF obtenu est très proche de l'IDF standard et confirme le bien-fondé de la formulation empirique.

Enfin, signalons que de nombreuses autres pondérations très performantes ont été proposées en RI, comme les modèles DFR (Divergence from Randomness, proposés par Amati et Van Rijsbergen (2002)) ou les modèles de langues (Ponte et Croft, 1998). Ces deux approches construisent des mesures de similarités basées sur des modèles probabilistes de fréquence des termes. La encore, ces modèles peuvent s'interpréter comme des pondérations dans un modèle vectoriel.

3 Vectorisation ou distance du second ordre

En plus des techniques de pondération exposées précédemment, nous proposons une autre alternative au TF-IDF que nous avons développée. Il s'agit de la vectorisation, dont nous décrivons dans cette section les principes.

3.1 Principe

La vectorisation est une technique de plongement (*embedding*) permettant de projeter un calcul de similarité quelconque entre deux documents (ou un document et une requête pour la RI) dans un espace vectoriel. Son principe est relativement simple : pour chaque document de la collection considérée, il consiste à calculer avec une mesure de similarité, quelle qu'elle soit, des scores de similarité entre ce document et m documents-pivots. Cette similarité est dite de premier ordre dans la suite de cet article. Les m scores obtenus forment ainsi un vecteur de m dimensions représentant le document (cf. figure 1) et place le document dans un nouvel espace vectoriel. Dès lors, la comparaison de deux documents (ou d'un document et d'une requête) peut donc s'effectuer de manière standard dans ce nouvel espace, par exemple en calculant une distance L2. C'est la similarité de second ordre.

Il est important de noter que la vectorisation change l'espace de représentation. Il ne s'agit donc pas seulement d'une réduction de l'espace ou d'une approximation de la distance

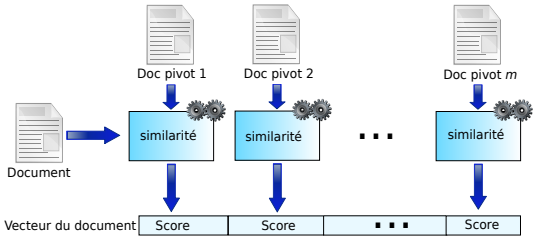

FIGURE 1 – Schématisation du principe de vectorisation d'un document

originelle comme proposée par exemple dans les travaux de (Bourgain, 1985). Il ne s'agit pas non plus d'une orthogonalisation, comme pour les approches LSI/LSA. C'est ce changement d'espace qui est à l'origine de deux propriétés intéressantes.
D'une part, cet *embedding* permet de réduire la complexité quand le calcul de similarité de premier ordre est trop coûteux pour être utilisé en-ligne (Claveau *et al.*, 2010).
D'autre part, la vectorisation permet que deux documents soient considérés comme proches s'ils sont proches des mêmes documents-pivots. Cette comparaison indirecte, ou affinité du second ordre, permet par exemple de mettre en relation des documents textuels qui ne contiennent pourtant aucun mot en commun.

3.2 Constitution des documents-pivots

De nombreuses alternatives sont possibles pour constituer les m documents-pivots, en fonction du problème abordé. Dans le principe, ces pivots ne sont pas nécessairement issus de la collection traitée, même s'il semble plus raisonnable que ce soit le cas ; on a ainsi une assurance plus grande d'une bonne adéquation, notamment du vocabulaire, de ces pivots avec les documents à traiter. Dans les expériences présentées ci-après, nous indiquons pour chaque application comment ces pivots ont été constitués.

3.3 Calcul de distances

Notre processus de vectorisation permet de ramener n'importe quel modèle de similarité à une représentation vectorielle. La comparaison de deux documents se fait donc par un calcul de distance entre vecteurs, comme dans le modèle vectoriel classique. Les vecteurs sont normalisés (en cohérence avec la distance utilisée) ; dans les expériences présentées dans cet article, nous utilisons une distance L2 (les vecteurs sont donc normalisés en L2). À la différence du modèle vectoriel classique dans lequel les vecteurs représentant les documents sont généralement extrêmement creux, les vecteurs obtenus par vectorisation n'ont pas forcément beaucoup de composantes à 0. Comme nous l'avons expliqué précédemment, cela rend le calcul de distance beaucoup plus coûteux s'il est fait de manière exhaustive et exacte. Dans les expériences reportées ci-dessous, les quantités de données manipulées permettent un tel calcul exact ; nous avons donc pris le parti d'évaluer les résultats sans le biais d'une recherche approximative. Les calculs de distance se font donc de manière classique avec la

distance L2.

4 Utilisation en recherche d'information

Dans les systèmes de recherche d'information, les modèles vectoriels sont très largement utilisés pour calculer les distances entre requêtes et textes. Il existe de nombreux articles ont comparé les mesures/pondérations standard, comme le TF-IDF et Okapi (Savoy, 2005, *inter alia*). Dans cette section, nous rapportons quelques expériences (Claveau *et al.*, 2010, pour les détails) les comparant également à la vectorisation.

4.1 Description de la tâche

Pour ces évaluations, nous utilisons deux collections de RI aux caractéristiques très différentes, en français et provenant de la campagne d'évaluation Amaryllis. La première est la collection ELDA, petite collection de 3500 documents issus de questions/réponses de la commission européenne, accompagnée de 19 requêtes. La seconde est la collection INIST composée de 160 000 documents (résumés d'articles de diverses disciplines scientifiques) et de 30 requêtes. Pour ces deux collections, les requêtes sont composées de plusieurs champs : titre, corps, description et concepts associés. Dans les expériences reportées ci-dessous, seuls le titre et le corps sont utilisés. À chaque requête est associée la liste des documents pertinents qui sont attendus en réponse et donc utilisés pour évaluer les performances des systèmes de RI. Dans les expériences rapportées ci-dessous, ces performances sont mesurées en utilisant les mesures classiques de la RI, à savoir la MAP (*Mean Average Precision*) et les précisions obtenues pour divers seuils de documents (5 premiers documents retournés, 10 premiers...) moyennées sur l'ensemble des requêtes.

4.2 Approches proposées

Pour ces expériences, un système de RI vectoriel a été implémenté dont seul le calcul des distances varie. Basé sur ces distances, les documents sont proposés par similarité décroissante avec les requêtes. Pour le calcul par vectorisation, les m documents-pivots sont choisis comme des concaténations aléatoires de documents de la collection formant une partition de l'ensemble des documents. Différents nombres de pivots (et donc la dimension de l'espace résultant) sont testés, et les similarités de premier ordre (servant à construire les vecteurs) sont calculées avec Okapi.

4.3 Résultats

Les tableaux 1 et 2 présentent les résultats de ce système, selon les différentes méthodes de calcul de distances. En prenant Okapi comme base de comparaison, on indique les améliorations statistiquement significatives (t-test avec $p = 0.05$) en gras et les dégradations significatives en italiques. Plusieurs éléments en ressortent. D'une part, le système basé sur

BM-25 domine très largement et dans tous les cas celui s'appuyant sur TF-IDF. Ce résultat est conforme à l'ensemble des expériences de ce type dans la littérature. Les résultats de la vectorisation sont moins tranchés en terme de performances globales (MAP), puisque les résultats sont très fortement améliorés pour la petite collection, mais comparable pour la collection INIST. En revanche, il apparaît clairement et dans tous les cas la propriété de la vectorisation de trouver plus de documents pertinents (précision améliorée pour des seuils hauts). C'est cette propriété, reposant sur la capacité de juger de la similarité de deux documents mêmes s'ils ne partagent pas de termes communs, qui est particulièrement utile pour certaines tâches de TAL.

	TF-IDF	Okapi	Vectorisation ($m = 1\,750$)	Vectorisation ($m = 3\,500$)
MAP	*28.36 (-21.7 %)*	36.22	39.01 (+7.7 %)	**43.46 (+20 %)**
P@10	*36.18 (-24.1 %)*	47.67	**51.67 (+8.4 %)**	**54.67 (+14.7 %)**
P@50	*26.37 (-12.1 %)*	30.00	29.07 (-3.1 %)	**33.53 (+11.8 %)**
P@100	*19.36 (-6.6 %)*	20.73	19.87 (-4.2 %)	21.50 (+3.7 %)
P@500	6.04 (-0.9 %)	5.99	5.71 (-4.8 %)	6.17 (+2.9 %)
P@1000	3.16 (-0.4 %)	3.15	3.15 (0 %)	**3.27 (+3.9 %)**
P@3000	1.08 (+0.4 %)	1.07	**1.17 (+9.0 %)**	**1.18 (+10 %)**

TABLE 1 – Performances des systèmes sur la collection ELDA

	TF-IDF	Okapi	Vectorisation ($m = 10\,000$)
MAP	*10.62 (-26.9 %)*	14.52	14.26 (-1.8 %)
P@10	*24.00 (-29.4 %)*	34.00	*29.00 (-14.7 %)*
P@50	*14.40 (-22.0 %)*	18.47	18.00 (-2.5 %)
P@100	*10.93 (-8.9 %)*	12.00	**13.47 (+12.2 %)**
P@500	4.16 (-2.6 %)	4.27	**4.93 (+15.3 %)**
P@1000	2.40 (-2.4 %)	2.46	**2.89 (+17.3 %)**
P@3000	1.00 (-1 %)	1.01	**1.12 (+10.9 %)**

TABLE 2 – Performances des systèmes sur la collection INIST

5 Utilisation pour la fouille de texte

Comme nous le soulignions dès l'introduction, calculer des distances entre textes n'est pas utile que pour la RI, mais aussi pour beaucoup de tâches du TAL. Dans cette section, nous illustrons l'importance du choix de la similarité dans un contexte de fouille de textes.

5.1 Description de la tâche

Cette tâche était l'une de celles proposées dans le cadre du Défi Fouille de Texte (DeFT) 2011 (Grouin et Forest, 2011). Elle consistait à retrouver l'année de parution d'extraits d'articles de journaux OCRisés publiés entre 1801 et 1944. Les participants disposaient de

données d'apprentissage (extraits d'articles avec leur date de parution), et de données de test (extraits d'articles pour lesquels il faut fournir la date de parution). Deux sous-tâches étaient proposées : l'une avec des extraits de 300 mots et l'autre avec des extraits de 500 mots.

La difficulté de ce défi tenait d'une part à la qualité très dégradée des textes OCRisés et au grand nombre de classes possibles (144 années). La mesure d'évaluation mise en place par les organisateurs permettait de pondérer les erreurs de datation en fonction de leur distance à l'année réelle. Cette mesure va de 0 pour un document éloigné de plus de 15 ans, à 1 quand la prédiction de date est exacte.

5.2 Approches proposées

L'approche que nous avons proposée lors de notre participation repose sur un apprentissage paresseux (*lazy-learning*), à savoir les k-plus proches voisins (k-ppv), qui se veut souple et adapté à la tâche. Dans cette approche, une instance inconnue est classée en trouvant les k instances connues les plus similaires et en lui assignant la classe majoritaire de ces instances. Il n'y a donc pas à proprement parler d'apprentissage, d'où le nom de *lazy-learning*, mais l'induction repose sur la calcul de similarité, qui permet de trouver les plus proches voisins, et la mise en œuvre du vote (Beyer *et al.*, 1999, pour les autres paramètres pouvant intervenir).

Ce calcul de similarité est donc central. Lors du défi, nous avons utilisé la pondération Okapi, et cette simple approche nous a permis d'être classés premiers. Dans la sous-section suivante, nous avons repris ces expériences et nous présentons en plus les résultats obtenus avec un TF-IDF et par vectorisation. Pour ces expériences, les documents-pivots sont simplement des concaténations aléatoires des articles de l'ensemble d'entraînement. La seule contrainte est que les documents concaténés doivent avoir la même année de parution. Chaque dimension de notre nouvel espace vectoriel correspond donc à une année (et éventuellement, plusieurs dimensions peuvent porter sur la même année). Comme précédemment, chaque document de l'ensemble d'entraînement est décrit à l'aide de ces pivots : sa distance (similarité de premier ordre) à chacun des pivots est calculée en utilisant Okapi, ce qui forme l'ensemble de ses coordonnées dans le nouvel espace. Il est fait de même pour les documents test. Finalement, les plus proches voisins d'un document test sont donnés par une distance $L2$.

5.3 Résultats

Le tableau 3 recense les résultats de l'approche k-ppv avec une distance de type Okapi tels que publiés, ainsi que la même approche utilisant cette fois la distance par vectorisation. À des fins de comparaison, nous indiquons également les résultats du même système qui utiliserait cette fois un TF-IDF standard avec une similarité cosinus, ainsi que ceux obtenus par le LIMSI, deuxième système le plus performant.

Il apparaît encore une fois l'intérêt de l'utilisation d'une pondération BM-25 comparée au TF-IDF standard. Celle-ci a permis d'obtenir les meilleurs résultats avec un algorithme simple de vote lors du challenge, alors que le TF-IDF aurait fait apparaître le système comme moins adapté que la proposition du LIMSI. La vectorisation permet en plus de dépasser ces résultats ; grâce au choix des documents-pivots, les documents de même année de parution peuvent avoir des représentations vectorielles proches, même s'ils ne partagent pas de mots communs.

	extraits de 300 mots	extraits de 500 mots
système LIMSI	0.378	0.452
k-ppv TF-IDF	0.364	0.398
k-ppv Okapi	0.430	0.472
k-ppv Vectorisation	**0.466**	**0.505**

TABLE 3 – Résultats sur la tâche de datation de DeFT 2011 d'un système k-ppv avec différents calculs de similarité et d'une *baseline* (système du LIMSI)

6 Utilisation pour la segmentation thématique

Outre la fouille de textes, le calcul de similarité est aussi utile dans certains systèmes dédiés à d'autres applications classiques du TAL, comme par exemple la segmentation thématique. Comme dans les sections précédentes, nous illustrons l'influence du choix du calcul de similarité dans un tel système de segmentation reposant sur ce calcul (Claveau et Lefèvre, 2011).

6.1 Description de la tâche

La segmentation thématique est une tâche classique du TAL consistant à diviser un texte ou un flux textuel en parties thématiquement cohérentes. De nombreuses méthodes ont été proposées dans la littérature, que l'on peut diviser en deux familles. Il y a d'une part des approches s'appuyant sur des propriétés de formatage des documents, ou sur la détection de marqueurs discursifs (Christensen *et al.*, 2005). L'autre grande famille d'approches s'appuie sur le contenu des documents pour détecter les changements de thème. C'est dans cette famille que s'inscrit notre approche et beaucoup des systèmes existants, tels que SEGMENTER (Kan *et al.*, 1998), l'approche Utiyama et Isahara (Utiyama et Isahara, 2001; Guinaudeau *et al.*, 2010), DOTPLOTTING (Reynar, 2000), C99 (Choi, 2000), TEXT-TILING (Hearst, 1997).

Ces différentes approches de l'état de l'art ont été comparées, que ce soit sur l'anglais (Choi, 2000) ou le français (Sitbon et Bellot, 2004). À des fins de comparaison, nous réutilisons ces données[3] sur le français pour évaluer l'importance du calcul de similarité dans ce contexte. Celles-ci ont été constituées artificiellement en segmentant et mélangeant des articles du journal *Le Monde* de plusieurs catégories (Sports, Arts...) et des extraits de la bible. Pour répondre à la critique d'artificialité de ces données, nous utilisons également deux autres jeux, composés respectivement de transcriptions de journaux TV et d'émissions de reportage développés par Guinaudeau *et al.* (2010). La segmentation de référence a été effectuée indépendamment en considérant qu'un changement de thème a lieu à chaque changement de reportage. Cette définition de la rupture thématique a l'avantage de correspondre à un besoin applicatif réel et bien défini. Les bandes-son de ces deux corpus ont été transcrites automatiquement par le système de reconnaissance de la parole IRENE (Huet *et al.*, 2010).

[3]Nous remercions L. Sitbon pour la mise à disposition de ces données et des systèmes de l'état-de-l'art.

6.2 Approche proposée

Notre technique de segmentation thématique cherche à détecter les ruptures thématiques en comparant (i.e. en calculant une distance) le contenu avant et après chaque segment. Plus la distance est importante, plus la rupture est probable. D'un point de vue technique, elle adapte un principe utilisé en segmentation d'image : la ligne de partage des eaux (*watershed*). Celle-ci consiste à représenter l'image à segmenter comme un relief (ou surface topographique) en calculant un gradient de l'image pour faire ressortir les zones de fortes variations (par exemple de luminance d'un pixel). Une inondation progressive du relief par ses minima est alors simulée et des digues sont (virtuellement) construites pour séparer les différents bassins associés à chaque minimum. À l'issue du processus, ces digues représentent les lignes de partage des eaux, ou autrement dit les frontières des régions.

Dans le cas de texte, il n'y a qu'une dimension à considérer, et le gradient est vu comme une mesure de distance textuelle. Un gradient est donc calculé entre chaque phrase (ou groupe de souffle dans le cas de transcriptions) : on calcule la similarité entre les phrases précédentes et suivantes de chaque frontière potentielle (Claveau et Lefèvre, 2011, pour une présentation plus détaillée). Il faut noter qu'on ne compare pas seulement le groupe de souffle précédent au groupe de souffle suivant, mais on considère les n précédents et les n suivants. Là encore, ce calcul de similarité peut se faire en utilisant les outils standard de RI comme le TF-IDF, ou bien Okapi, ou encore par vectorisation.

6.3 Résultats

Pour évaluer la qualité des segmentation sur nos jeux de données, nous indiquons la mesure WindowDiff (WD), habituellement utilisée pour l'évaluation de ce type de tâche, et qui peut être vu comme un taux d'erreurs (Pevzner et Hearst, 2002, pour une présentation détaillée). À des fins de comparaison avec des systèmes n'utilisant pas le WindowDiff, nous utilisons aussi la F-mesure (F1).

	News		*Reports*	
Méthodes	F1 (%)	WD	F1 (%)	WD
Utiyama (Utiyama et Isahara, 2001)	59.44	-	51.09	-
Guinaudeau (Guinaudeau *et al.*, 2010)	61.41	-	62.92	-
DOTPLOT (Reynar, 2000)	36.42	0.4472	49.49	0.2125
C99 (Choi, 2000)	50.25	0.3646	57.42	0.1893
TEXTTILING (Hearst, 1997)	38.73	0.313	23.38	0.3456
TF-IDF + Watershed	43.04	0.3833	60.12	0.1844
Okapi + Watershed	60.04	0.2571	69.22	0.1428
Vectorisation + Watershed	**64.4**	**0.2269**	**73.31**	**0.1181**

TABLE 4 – Performances des systèmes de segmentation sur la collection *News* et *Reports*

Sur cette application, les résultats sont encore une fois très homogènes, quel que soit le corpus d'évaluation. À système équivalent, le choix de la mesure a une très grande influence, et on constate comme précédemment que le TF-IDF est surpassé par Okapi, lui-même légèrement dépassé par l'approche par vectorisation. On note encore qu'à architecture similaire, le

Méthodes	Sous-corpus de test				
	Sciences	Éco	Sports	Arts	Bible
Meilleur système	0.2132	0.2243	0.2839	0.2811	0.3139
(d'après Sitbon et Bellot (2004))	C99	C99	C99	C99	DOTPLOT
TF-IDF + Watershed	0.2964	0.2996	0.3205	0.3560	0.3702
Okapi + Watershed	0.2135	0.2177	0.2654	0.2729	0.2981
Vectorisation + Watershed	**0.1967**	**0.1836**	**0.2582**	**0.2587**	**0.2931**

TABLE 5 – Performances (WindowDiff) des systèmes de segmentation sur la collection de Sitbon et Bellot (2004)

système avec TF-IDF aurait été écarté, ses performances étant largement inférieures à l'état-de-l'art.

7 Conclusion

Le calcul de similarité entre textes est une problématique importante pour le TAL. Pour autant, ce qui n'est souvent qu'un composant dans des systèmes plus complexes est parfois négligé et des solutions sous-optimales sont employées. Dans les expériences précédentes relevant du cas très fréquent de la représentation vectorielle, on a montré combien un simple changement de pondération (de TF-IDF vers Okapi) pouvait modifier les performances finales d'un tel système. Ainsi, il convient de s'interroger si l'emploi de ces calculs de similarité sous-optimaux n'a pas condamné des systèmes qui auraient été par ailleurs jugés performants à condition d'utiliser un module de similarité plus état-de-l'art.

Outre l'utilisation de pondérations plus actuelles comme Okapi, nous avons montré que la vectorisation offre d'excellentes performances et des propriétés intéressantes. D'un point de vue technique, elle conserve le cadre vectoriel qui permet au besoin l'emploi d'outils de calcul de distances efficaces. Une autre propriété intéressante pour le TAL est la possibilité de juger de la similarité de textes même lorsqu'ils ne partagent pas de mots communs.

Quelques remarques s'imposent en complément de ces résultats. Tout d'abord, la représentation sac-de-mots n'est pas la représentation la plus adaptée à tous les problèmes de similarité. Les approches par modèles de langue, outils communs du TAL, sont par exemple plus adaptées dès lors que l'aspect séquentiel du texte est important (Ebadat *et al.*, 2011, pour un exemple). Il en va de même pour les similarités entre arbres syntaxiques ou graphes sémantiques.

D'un point de vue plus général, la relative séparation des communautés RI et TAL explique sans doute le relatif manque d'importance accordée aux calculs de similarité dans certaines tâches du TAL. Cela milite pour des rapprochements thématiques ponctuels entre ces communautés, par exemple par le biais de tutoriels, numéros spéciaux de journaux, organisation de conférences jointes... Enfin, cela peut passer aussi par des enseignements dédiés ; on remarque en effet que les descriptifs des principales formations TAL, lorsque disponibles, font rarement état de modules abordant l'indexation ou la recherche d'information.

Références

AMATI, G. et VAN RIJSBERGEN, G. J. (2002). Probabilistic models of information retrieval based on measuring the divergence from randomness. *ACM Transactions on Information Systems*.

BEYER, K., GOLDSTEIN, J., RAMAKRISHNAN, R. et SHAFT, U. (1999). When is "nearest neighbor" meaningful ? *In Proceedings of the Int. Conf. on Database Theory*, pages 217–235.

BOURGAIN, J. (1985). On Lipschitz embedding of finite metric spaces in hilbert space. *Israel Journal of Mathematics*, 52(1).

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

CHRISTENSEN, H., KOLLURU, B., GOTOH, Y. et RENALS, S. (2005). Maximum entropy segmentation of broadcast news. *In Proceedings of the 30th IEEE ICASSP*.

CLAVEAU, V. et LEFÈVRE, S. (2011). Topic Segmentation of TV-streams by mathematical morphology and vectorization. *In Proceedings of InterSpeech*, pages 1105–1108, Italie.

CLAVEAU, V., TAVENARD, R. et AMSALEG, L. (2010). Vectorisation des processus d'appariement document-requête. *In 7e conférence en recherche d'informations et applications, CORIA'10*, pages 313–324, Sousse, Tunisie.

DATAR, M., IMMORLICA, N., INDYK, P. et MIRROKNI, V. (2004). Locality-sensitive hashing scheme based on p-stable distributions. *In Proc. of the 20th ACM Symposium on Computational Geometry*, Brooklyn, New York, USA.

EBADAT, A. R., CLAVEAU, V. et SÉBILLOT, P. (2011). Using shallow linguistic features for relation extraction in bio-medical texts. *In Actes de la 18e conférence sur le Traitement Automatique des Langues Naturelles, TALN'11*, volume 2, pages 125–130, Montpellier, France.

GROUIN, C. et FOREST, D., éditeurs (2011). *Actes de l'atelier Défi Fouille de Textes (DeFT'11)*, Montpellier, France.

GUINAUDEAU, C., GRAVIER, G. et SÉBILLOT, P. (2010). Improving ASR-based topic segmentation of TV programs with confidence measures and semantic relations. *In Proc. Annual Intl. Speech Communication Association Conference (Interspeech)*.

HARTER, S. (1975). A probabilistic approach to automatic keyword indexing. *Journal of the american society for information science*, 26(6):197–206.

HEARST, M. (1997). Text-tiling : segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23(1):33–64.

HUET, S., GRAVIER, G. et SÉBILLOT, P. (2010). Morpho-syntactic post-processing with n-best lists for improved French automatic speech recognition. *Computer Speech and Language*, 24(4):663–684.

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

LEJSEK, H., ASMUNDSSON, F., JÓNSSON, B. et AMSALEG, L. (2008). Nv-tree : An efficient disk-based index for approximate search in very large high-dimensional collections. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 99(1).

LUHN, H. P. (1958). The automatic creation of literature abstracts. *IBM Journal on Research and Development*, 2(2).

MOREAU, F. et SÉBILLOT, P. (2005). Contributions des techniques du traitement automatique des langues à la recherche d'information. Rapport technique 1690, IRISA.

PEVZNER, L. et HEARST, M. (2002). A critique and improvement of an evaluation metric for text segmentation. *Computational Linguistics*.

PONTE, J. M. et CROFT, W. B. (1998). A language modeling approach to information retrieval. *In Proc. of SIGIR*, Melbourne, Australie.

REYNAR, J. C. (2000). *Topic Segmentation : Algorithms and applications*. Thèse de doctorat, University of Pennsylvania.

ROBERTSON, S. E., WALKER, S. et HANCOCK-BEAULIEU, M. (1998). Okapi at TREC-7 : Automatic Ad Hoc, Filtering, VLC and Interactive. *In Proceedings of the 7th Text Retrieval Conference, TREC-7*, pages 199–210.

SALTON, G. (1975). *A Theory of Indexing*. Regional Conference Series in Applied Mathematics. Society for Industrial and Applied Mathematics, Philadelphy.

SALTON, G., WONG, A. et YANG, C. S. (1975). A vector space model for automatic indexing. *Comm. of the ACM*, 18(11).

SAVOY, J. (2005). Comparative study of monolingual and multilingual search models for use with asian languages. *ACM Transactions on Asian Languages Information Processing*, 4(2).

SITBON, L. et BELLOT, P. (2004). Évaluation de méthodes de segmentation thématique linéaire non supervisées après adaptation au fran cais. *In Actes de la conférence Traitement automatique des langues*, Fez, Tunisie.

SPÄRCK JONES, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28(1).

SPÄRCK JONES, K., WALKER, S. G. et ROBERTSON, S. E. (2000). Probabilistic model of information retrieval : Development and comparative experiments. *Information Processing and Management*, 36(6).

STEIN, B. (2007). Principles of hash-based text retrieval. *In Proc. of SIGIR*, Amsterdam, Pays-Bas.

TIRILLY, P. (2010). *Traitement automatique des langues pour l'indexation d'images*. Thèse de doctorat, Université de Rennes 1.

UTIYAMA, M. et ISAHARA, H. (2001). A statistical model for domain-independent text segmentation. *In Proceedings of the 9th conference of the ACL*.