Regrouper les données textuelles et nommer les groupes à l'aide de classes recouvrantes

Marian-Andrei Rizoiu[*,**], Julien Velcin[*,***]
Jean-Hugues Chauchat[*,****]

[*]Laboratoire ERIC, Université Lumière Lyon2,
5 av. P. Mendès-France 69676 Bron Cedex, France
[**]Marian-Andrei.Rizoiu@univ-lyon2.fr
[***]Julien.Velcin@univ-lyon2.fr
[****]Jean-Hugues.Chauchat@univ-lyon2.fr

Résumé. Organiser les données textuelles et en tirer du sens est un défi majeur aujourd'hui. Ainsi, lorsque l'on souhaite analyser un débat en ligne ou un forum de discussion, on voudrait pouvoir rapidement voir quels sont les principaux thèmes abordés et la manière dont la discussion se structure autour d'eux. Pour cela, et parce que un même texte peut être associé à plusieurs thèmes, nous proposons une méthode originale pour regrouper les données textuelles en autorisant les chevauchements et pour nommer chaque groupe de manière lisible. La contribution principale de cet article est une méthode globale qui permet de réaliser toute la chaîne, partant des données textuelles brutes jusqu'à la caractérisation des groupes à un niveau sémantique qui dépasse le simple ensemble de mots.

1 Introduction

L'extraction d'information à partir de données non structurées, en particulier textuelles, est un domaine de recherche très actif. Internet constitue une véritable mine où l'on trouve actuellement ce type d'information : articles, blogs, *chats*, forums, débats, etc. Cette profusion explique les nombreux travaux qui cherchent à extraire le plus d'information "utile", ayant du sens, à partir de ces données. Le présent travail a été mené en étroite collaboration avec une jeune entreprise qui organise et analyse des débats en ligne.

Supposons que nous ayons un ensemble de textes qui traitent des conséquences économiques d'une décision politique. Nous aimerions disposer d'un outil capable d'extraire les principaux thèmes associés aux réactions à cette décision, et ce de manière automatique et avec un minimum de connaissance préalable sur les textes. Cet outil doit regrouper les textes puis proposer un ou plusieurs nom(s) à chacune de ces catégories. Les thématiques pourraient alors être : la "politique du gouvernement", les "propositions de l'opposition", les "réactions syndicales", l'"efficacité économique", la "justice sociale", etc. Les textes en langage naturel étant naturellement associés à plusieurs thématiques Cleuziou (2007), nous avons choisi de développer un système capable si besoin de classer un même texte dans plusieurs catégories.

Pour cela, nous proposons un outil automatisé qui puisse traiter des données textuelles et adaptable à plusieurs langues. Notre travail est organisé en deux étapes : la première met en évidence les thématiques abordées dans le corpus et regroupe les textes autour de celles-ci de manière recouvrante ; la deuxième étape caractérise chaque thématique en proposant des mots, ou des suites de mots, extraits directement des textes. Une fois les groupes de documents obtenus et nommés, l'information contenue dans chaque texte particulier peut être synthétisée à l'aide des noms des différentes catégories auxquelles il appartient. De cette manière, deux objectifs sont atteints : dégager automatiquement une liste de thématiques abordés dans le corpus, et affecter chaque document dans une ou plusieurs de ces thématiques. Ceci permet d'appréhender rapidement un corpus complexe qui regroupe de très nombreux textes.

Nos contributions peuvent être ainsi résumées :

– *une méthode globale* qui 1) inclut un algorithme de clustering recouvrant et 2) le couple à une méthode d'extraction de motifs fréquents. Cette méthode fonctionne avec peu de connaissances (liste de mots-outils) et peut donc être employée avec différentes langues.
– *de nouvelles expérimentations* qui complètent celles réalisées précédemment dans la littérature. Ainsi, nous comparons plusieurs mesures de pondération des mots pour décrire les textes.
– *une évaluation des noms basée sur des experts.* Il est toujours difficile d'évaluer objectivement la qualité des noms car celle-ci comporte une part de subjectivité. Nous proposons une évaluation basée sur un système de notes données par des experts.

La section 2 présente les approches existantes dans la littérature pour le problème du clustering recouvrant et pour l'extraction des expressions fréquentes. La section 3 expose notre solution ; elle est suivie, dans la section 4, par une série d'expérimentations menées en langue française et en langue anglaise. La section 5 conclut et propose des perspectives de recherche.

2 État de l'Art

Geraci et al. (2006) observent que le problème de l'extraction des thématiques peut être divisé en deux sous-problèmes : le regroupement des documents similaires d'une part, et l'extraction d'expressions clés à partir du corpus d'autre part. A la lumière de cette remarque, notre recherche bibliographique se divise naturellement en deux parties.

2.1 Regrouper les documents

Le domaine du regroupement des données textuelles (*text clustering*) a été beaucoup traité dans la littérature. Des résultats significatifs ont été obtenus, plus particulièrement pour la tâche d'extraction des thématiques (*topic extraction*). Beaucoup d'outils existants (AGAPE par Velcin et Ganascia (2007), ARMIL par Geraci et al. (2006)), offrent la possibilité de construire une partition des textes (*crisp clustering*), ce qui signifie que chaque document ne peut appartenir qu'à une seule catégorie à la fois. Considérant que les textes en langage naturel évoquent généralement plusieurs thématiques, nous avons centré notre travail sur les techniques de clustering recouvrant. Par recouvrant, nous entendons dans cette section les techniques qui construisent un recouvrement au sens mathématique (un document appartient exactement à une ou plusieurs thématiques), mais également celles qui construisent une pondération sur l'appartenance aux catégories (un document appartient plus ou moins aux différentes thématiques).

Singular Value Decomposition (SVD). La Décomposition en Valeurs Singulières est utilisée dans LINGO par Osinski (2003) pour obtenir des catégories qui se chevauchent. L'idée principale de l'algorithme consiste à décomposer la matrice termes/documents, chaque document étant représenté par un vecteur de mots, dans un produit de trois matrices : $A = USV^T$. $d_i = \sum_{j=1}^{k} \alpha_j e_j$, où les e_j peuvent être considérés comme les centres des k clusters.

Latent Dirichlet Allocation (LDA) proposé par Blei et al. (2003) est un modèle probabiliste génératif. Il considère les documents comme des collections de mots et chaque mot est considéré comme l'échantillon d'un modèle de mélange où des composants du mélange sont des représentations des sujets traités ou thématiques. On obtient des distributions de probabilités qui peuvent être interprétées comme des probabilités d'appartenance des documents aux différentes catégories. LDA présente le même type de sortie "floue" que l'approche SVD.

Overlapping K-Means (OKM) est une extension de l'algorithme classique des K-Means, proposée par Cleuziou (2007). Il partage les mêmes grands principes que celui-ci et il a pour objectif de minimiser une fonction critère basé sur un principe de distorsion. La principale différence de l'algorithme OKM par rapport aux K-Means est qu'un document peut être attribué à plusieurs clusters. La section 3.3 décrit la méthode OKM de manière plus détaillée.

2.2 Extraction des motifs fréquents

Bien que le regroupement des documents, présenté dans la section précédente, constitue déjà une bonne méthode pour organiser une collection de textes (Pons-Porrata et al. (2007)), il est nécessaire de synthétiser davantage l'information lorsque celle-ci devient trop volumineuse. Une solution consiste à fournir aux utilisateurs une description intelligible des catégories. Une bonne description pour une catégorie est un motif qui contient tous les mots consécutifs possédant une signification qui dépasse celle des mots isolés (par ex. "fouille de données"). Nous appelons ces motifs des "expressions" (*phrase*). Notons que cette description peut comporter des prépositions et des articles qui font sens pour le lecteur humain (dans notre exemple le "de" est très important). Une expression clé est alors "une séquence d'un ou plusieurs mots jugés pertinents quand ils sont pris ensemble", tandis qu'un mot clé est "un mot isolé très pertinent" (Hammouda et al. (2005)). Roche (2004) divise les algorithmes d'extraction de mots clés en 3 catégories, en fonction de leur approche : linguistique, numérique et hybride.

Les approches linguistiques. Dans Roche (2004), trois systèmes linguistiques sont présentés : TERMINO, LEXTER et INTEX & FASTR. Tous ces systèmes utilisent des informations morphologiques et syntaxiques sur les mots dans les textes. Avec des textes ainsi marqués, chaque système a une approche différente pour découvrir les expressions clés. TERMINO utilise un analyseur lexico-syntaxique pour décrire les phrases et certains motifs (*patterns*) sont utilisés pour découvrir les motifs clés (ex. : <Tete> <Groupe Prépositionnel> <Groupe Adjectivale>). LEXTER utilise les informations morphologiques pour extraire du texte les groupes nominaux maximaux.

Les approches numériques. Ces algorithmes n'utilisent que des informations numériques (statistiques) afin de découvrir les thématiques. Certains mesurent la dépendance entre deux mots dans une collocation binaire, aussi appelée "bigramme". Pour chaque couple de mots dans le texte, l'information mutuelle est calculé. Cela permet de calculer la dépendance entre

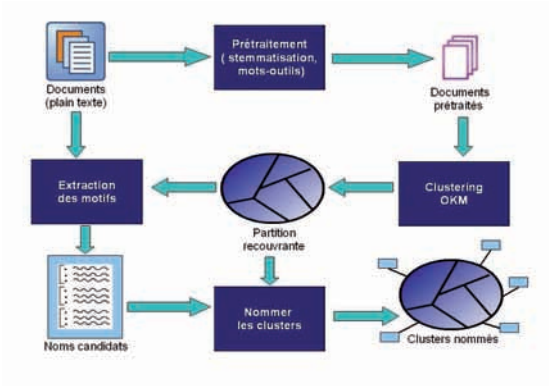

FIG. 1 – *Fonctionnement général du système.*

deux mots lorsqu'ils se trouvent l'un à côté de l'autre ou dans une fenêtre de dimension précise. Dans Anaya-Sánchez et al. (2008) par exemple, une fenêtre de dimension 11 est considérée autour d'un mot (5 mots avant + le mot centre + 5 mots après).

A partir de l'algorithme d'extraction des bigrammes, certains auteurs (EXIT de Roche (2004), ESATEC de Biskri et al. (2004)) proposent de construire des n-grammes en combinant de façon itérative les bigrammes, ou alors en ajoutant un nouveau mot a l'un des (n-1)-grammes existants. Beaucoup de mesures statistiques ont été proposées pour calculer la force de la relation entre deux mots : β-similitude par Anaya-Sánchez et al. (2008), la mesure LocalMaxs combinée à la Probabilité Symétrique Conditionnelle et la *Mutual Expectation* par da Silva et al. (1999) et Dias et al. (2000). Nikos et al. (2002) effectuent des expériences sur certains des mesures les plus connues (t-score, , *Pearson's* χ*-square*, *log-likelihood ratio*, etc.) afin de comparer leur capacité à identifier des bigrammes.

D'autres approches ne reposent pas sur la détection des n-grammes. Dans LINGO (Osinski (2003)Osinski et Weiss (2004)), les expressions clés sont découverts avec un approche basée sur un arbre des suffixes. Cet algorithme est décrit plus en détail dans la section 3.3.

Les approches hybrides. Un système hybride ajoute des informations linguistiques à un système essentiellement numérique. Ce processus améliore généralement les résultats (Biskri et al. (2004)).

3 Approche proposée

Dans ce travail, nous proposons une solution modulaire globale pour réaliser toute la chaîne de traitement, partant des données textuelles brutes, fournissant une caractérisation des catégories extraites à un niveau sémantique qui dépasse le simple ensemble de mots clés. Nous avons cherché une solution qui serait le plus possible indépendante de la langue utilisée dans le texte.

La figure 1 permet de se faire une idée du fonctionnement général de notre système. L'entrée du système est une collection de documents rédigés en langage naturel. La première étape

consiste à prétraiter cette collection pour la rendre utilisable par l'algorithme d'apprentissage. A partir de cette collection prétraitée, les documents sont regroupés en utilisant l'algorithme OKM (cf. section 2.1). Cet algorithme a été choisi, de préférence aux autres approches discutées précédemment, principalement car il produit un recouvrement des classes. Il n'est donc pas nécessaire de fixer un seuil pour transformer les degrés d'appartenance en une relation d'appartenance stricte. De plus, cet algorithme est simple à implémenter et très efficace en terme de temps de calculs, ce qui est adapté aux données volumineuses que nous souhaitons traiter.

Le module d'extraction des motifs reprend le corpus de textes dans son format initial. La raison qui a motivé ce choix est que le module de prétraitement élimine les mots-outils (*stopwords*), ce qui se révèle problématique dans l'extraction des noms de clusters. L'utilisation d'ensemble de mots lemmatisés, séparés de leurs articles considérés comme "superflus", produit des noms quasi vides de sens pour un lecteur humain. Pour trouver des noms-candidats de manière la plus automatisée possible, nous avons choisi d'utiliser une approche basée sur un arbre des suffixes (cf. section 2.2).

Dans la dernière étape, notre système associe les noms découverts à l'étape précédente avec les catégories recouvrantes trouvées par OKM. Les noms sont choisis pour qu'ils soient représentatifs de tous les documents de la catégorie.

Dans la réalisation de notre système, nous avons choisi de travailler dans le cadre classique de la fouille de textes, en utilisant une représentation vectorielle des documents Salton et al. (1975). Nous avons fait ce choix afin de réduire la dépendance de notre système à la langue, et afin de pouvoir traiter efficacement de grandes quantités de données textuelles ; nous savons cependant que des structures de données plus complexes ont été développées pour le langage naturel et qu'elles conservent mieux l'information ; mais elles semblent difficiles à appliquer à de très grands corpus. Nous détaillons à présent chaque étape de notre système.

3.1 Prétraitement

Le prétraitement est une partie importante de tout algorithme de fouille de textes. Il s'agit ici de la seule qui dépende de la langue. Elle est composée de deux éléments : la stemmatisation (*stemming*) et l'*élimination des mots-outils* (*stopwords*). Pour l'anglais on utilise l'algorithme de Porter (1980) et pour le français l'algorithme proposée dans le projet CLEF [1] par le Département d'informatique de l'Université de Neuchâtel [2] Les listes de mots-outils ont été extraites de la même adresse. Comme la phase de découverte des noms-candidats requiert les textes dans leur forme initiale, une version non traitée des documents est également conservée.

3.2 Regroupement

Une fois les textes prétraités, le processus de regroupement des textes (*clustering*) peut commencer. Initialement, les documents sont traduits en un ensemble de vecteurs de mots en utilisant, au choix, quatre pondérations différentes : presence/absence de mots, fréquence de Mots, IDF (*Inverse Document Frequency*) ou TFxIDF (*Term Frequency x Inverse Document Frequency*). Dans la réalisation de notre système, nous nous sommes inspirés de Myung et al.

[1] `http://www.clef-campaign.org/`
[2] `http://members.unine.ch/jacques.savoy/clef/index.html`

(2009) pour pouvoir choisir facilement une mesure, et pour pouvoir combiner différentes mesures.

Avec les documents transformés en vecteurs, le regroupement est réalisé à l'aide de l'algorithme OKM qui autorise les recouvrements : un texte peut appartenir à une ou plusieurs catégories. Si avec les K-Means chaque document est affecté au centroïde le plus proche, OKM associe à chaque document une *image* qui correspond à un ensemble de centroïdes. L'image est le *centre de gravité* des centroïdes qui ont sont associés au document. Le document est alors ajouté, non plus à une unique catégorie, mais aux catégories qui permettent de minimiser la distance entre le document et cette image.

D'abord, on choisit au hasard k centroïdes dans l'ensemble des données, puis on réitère les deux étapes suivantes : 1) affectation des documents aux clusters à l'aide des centroïdes ; 2) mise à jour des clusters basée sur la nouvelle configuration calculée à l'étape 1. L'algorithme s'arrête lorsqu'il converge vers un minimum local. Notre implémentation de l'algorithme *OKM* respecte les indications données par les auteurs Cleuziou (2007) ; seule la condition d'arrêt a été modifiée. Dans sa forme originelle, l'algorithme s'arrête quand la partition ne varie plus. Mais, en acceptant des classes recouvrantes, les centroïdes peuvent encore changer à l'itération suivante car, dans *OKM*, la mise à jour des centroïdes dépend à la fois des documents présents dans leur propre groupe, mais aussi des autres centroïdes résultants de la dernière itération.

Le processus ultérieur d'attribution des noms aux classes étant fortement tributaire de ces centroïdes, il nous a semblé important de continuer à optimiser ces vecteurs même si le recouvrement, lui, ne change pas ; c'est pourquoi nous avons fixé un seuil minimum de variation de la fonction critère entre deux itérations à $\epsilon = 0,001$.

3.3 Extraction des noms-candidats

L'un des objectifs de notre système est de donner à l'utilisateur une description lisible des catégories. Pour ce faire, on extrait un ensemble de noms-candidats à partir du corpus des documents non traités. Osinski (2003) présente les quatre conditions qui doivent être remplies pour qu'une collocation (ou "terme") puisse être considérée comme un nom-candidat :

- apparaître dans le texte avec une fréquence déterminée. Ceci est basé sur l'hypothèse que les expressions qui apparaissent souvent dans le texte ont une plus grande puissance descriptive. De manière duale, les apparitions isolées ont une forte probabilité d'être des expressions incorrectes Roche (2004).
- ne pas traverser les frontières d'une phrase, parce que les fin de phrases sont des marqueurs de déplacements thématiques.
- être une expressions dite complète. Les expressions complètes ont plus de sens que les expressions incomplètes ("président Nicolas Sarkozy" plutôt que "président Nicolas").
- ne pas commencer, ni finir, avec un mot-outil. Par contre, les mots-outils peuvent être situés à l'intérieur d'une expression.

Nous avons choisi d'utiliser une approche basée sur un arbre des suffixes pour extraire les noms-candidats. L'algorithme fonctionne en deux phases : on trouve les expressions complètes à gauche et à droite dans un premier temps, puis on fait l'intersection des deux ensembles que l'on filtre pour obtenir l'ensemble des expressions complètes. Nous détaillons le fonctionnement de cet algorithme ci-dessous.

Construction de l'arbre des suffixes L'algorithme de découverte des expressions complètes à droite repose sur l'utilisation d'un arbre des suffixes. Un arbre des suffixes est un tableau trié alphabétiquement de tous les suffixes d'une chaîne de caractères. Dans notre cas, l'unité fondamentale n'est pas le caractère, mais le mot. L'un des problèmes les plus importants dans la construction de cet arbre est l'efficacité du tri des suffixes en termes de temps d'exécution et d'espace mémoire. Larsson (1998) présente deux solutions, Manber et Myers et l'algorithme de Sadakane, dont il compare les performances. Nous avons choisi d'utiliser la deuxième approche plus rapide. Comme une expression clé ne peut pas dépasser la limite d'une phrase, nous avons modifié l'algorithme proposé dans Osinski (2003) en construisant l'arbre des suffixes pour chaque phrase, puis nous avons concaténé tous les arbres en un seul.

Extraction des expressions Une fois que nous avons trouvé les expressions complètes "à droite", nous cherchons les expressions complètes "à gauche". Ceci peut être réalisé en appliquant le même algorithme que précédemment, mais en inversant tous les mots du document (le premier mot devient le dernier, et ainsi de suite). Les deux séries de mots peuvent ensuite être croisées en temps linéaire. Précisons ici que les noms-candidats extraits ainsi peuvent être à la fois des expressions complexes de plusieurs mots, mais également des mots isolés car il est parfois possible d'expliquer le contenu d'un cluster avec un seul mot, si celui-ci est suffisamment signifiant. La dernière phase de l'algorithme consiste à filtrer les noms-candidats. Tout d'abord, les candidats dont la fréquence ne dépasse pas un seuil donné sont éliminés. Il s'agit souvent d'expressions que l'on peut assimiler à du bruit. Dans la deuxième phase de filtrage, on élimine les mots-outils quand ils se trouvent au début ou à la fin des noms-candidats (cf. quatrième condition ci-dessus).

3.4 Associer les noms aux clusters

La dernière étape de notre système consiste à associer un nom-candidat à chacun des clusters en utilisant le recouvrement calculé à l'étape 2 et les candidats de l'étape 3. Nous proposons de choisir les candidats qui présentent une similarité maximum, au sens de la distance du cosinus, avec le centroïde de chaque catégorie. Rappelons que le centroïde est un individu (existant ou non) qui synthétise le mieux le contenu du cluster. Autrement dit, il s'agit d'un vecteur dont les termes les plus pertinents ont un poids élevé. Ici, nous postulons qu'un nom-candidat est d'autant plus représentatif de la catégorie qu'il est proche de son centre.

Pour cela, nous prenons tous les noms-candidats et les réintroduisons dans la collection de documents comme des "pseudo-documents". Après avoir appliqué le même prétraitement que celui prévu pour les documents originaux, nous les introduisons dans l'espace vectoriel en utilisant le même système de pondération des termes que pour les autres documents. Ensuite, il suffit de calculer la similarité entre chacun de ces "pseudo-documents" et le centre des clusters. Le nom-candidat qui présente le score le plus élevé est choisi pour étiqueter le cluster.

4 Expérimentations

Le système que nous proposons est évalué en deux parties. La première série d'expérimentations se concentre sur la phase de regroupement et la seconde sur la phase de caractérisation des clusters.

4.1 Jeux des Données

Nous souhaitions proposer un système aussi indépendant que possible de la langue. C'est pourquoi nous avons effectué nos expérimentations sur des corpus écrits en anglais et en français, avec des textes de styles d'écriture différents.

En ce qui concerne l'anglais, nous avons naturellement choisi le corpus Reuters [3]. Il contient des milliers d'articles de journaux comportant chacun entre 21 et 1000 mots. En raison principalement du coût élevé de l'évaluation du nom de chacun des clusters par des experts, nous avons choisi de travailler sur un sous-ensemble de ces données. Nous avons suivi la méthodologie présenté en Cleuziou (2007) en choisissant les documents qui ont été étiquetés par les experts comme présentant au moins un sujet. Cependant, pour tester la capacité de notre système à traiter des corpus plus grands, nous avons utilisé jusqu'à 2800 documents.

En ce qui concerne le français, nous avons utilisé des textes recommandés par notre partenaire industriel : des textes de forums en ligne, comme par exemple "Y at-il trop commémorations en France ?". Ces forums contiennent entre 200 et 300 documents (276 pour le forum de Libération) comportant chacun entre 3 et 300 mots. Contrairement aux articles de journaux de Reuters, le style d'écriture est ici très informel.

4.2 Evaluation du regroupement

Les expériences de Cleuziou (2007) évaluent déjà les performances de OKM comparativement à un algorithme non-recouvrant de type K-Means. Nous avons repris ces expérimentations et les avons enrichies en étudiant l'influence du système de pondération des termes sur le comportement de l'algorithme. Pour le regroupement, nous avons utilisé une méthode d'évaluation à base d'experts : le produit de l'algorithme est comparé au regroupement proposé par ces experts. Suite à l'expérimentation réalisée dans Cleuziou (2007), nous avons utilisé un ensemble de données de 262 documents correspondant à une partie du corpus Reuters.

Nous considérons deux documents comme étant associées s'ils appartiennent à une même classe dans le processus de regroupement. De plus, nous considérons que cette association est correcte si les deux documents ont une étiquette Reuters commune. La partition est ensuite évaluée en fonction du nombre total des associations (noté n_a), le nombre de associations correctes (noté n_b) et le nombre total des associations attendus (noté n_c). Avec ces indicateurs, on peut calculer la précision, le rappel et F_{score} :

$$precision = \frac{n_b}{n_a};\ \ recall = \frac{n_b}{n_c};\ \ F_{score} = \frac{(\beta^2 + 1) * precision * recall}{\beta^2 * precision + recall}$$

Nous avons effectué nos expérimentations en fixant $\beta = 1$, en faisant varier le nombre de classes de 5 à 30, et en répétant 10 fois l'algorithme de regroupement pour prendre le meilleur score obtenu. Les résultats de nos expérimentations sont données dans les figures 2 et 3.

La figure 2 confirme l'hypothèse initiale selon laquelle une approche par chevauchement est plus adapté à la fouille de textes en langage naturel. Alors que la qualité des partitions pour l'algorithme sans recouvrement baisse quand le nombre de groupes augmente, l'approche avec chevauchement maintient un score stable. Ces résultats tout à fait naturels confirment ceux obtenus par Cleuziou (2007).

[3] http://mlr.cs.umass.edu/ml/datasets/Reuters-21578+Text+Categorization+Collection

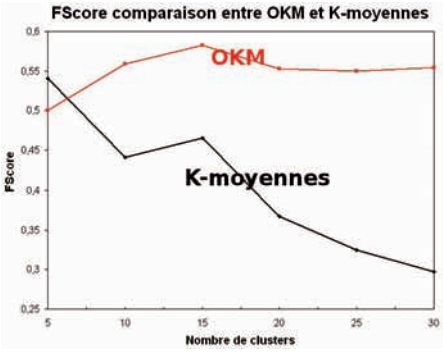

FIG. 2 – *Comparaison du FScore entre **OKM** et **KMeans**.*

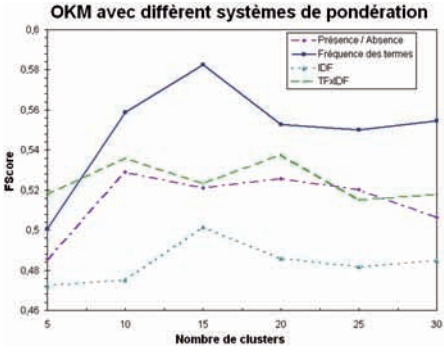

FIG. 3 – *F_{score} obtenu pour chaque système de pondération par OKM.*

Notre contribution dans ces premières expérimentations consiste à faire varier la pondération des termes. Les résultats de cette comparaison se trouvent dans la figure 3. Il est intéressant d'observer que la mesure qui obtient les meilleurs résultats ici est le TF ("Term Frequency"). En effet, la plupart des outils de fouille de textes utilise aujourd'hui la mesure du TFxIDF.

4.3 Evaluation des noms des catégories

Pour évaluer les noms des catégories formées par l'algorithme de clustering, nous avons choisi une approche basée sur les experts. Cela signifie qu'il faut que plusieurs personnes évaluent manuellement chaque résultat et lui donnent une note. Pour ces expérimentations, nous avons simplement utilisé la moyenne des notes données par les experts. Nous avons choisi cette approche car nous cherchons à obtenir des noms intelligibles pour un être humain. Bien que deux experts puissent juger différemment la pertinence d'un nom, nous postulons qu'un consensus est possible et permet de discriminer les noms intéressants des autres ; par ailleurs, la littérature actuelle ne propose pas de *gold standard* en la matière.

Nous avons effectué notre évaluation sur deux corpus différents : un en anglais (une sous partie du corpus Reuters), et un en français (une discussion en ligne – le forum "Y a-t-il trop de commémorations en France ? "). Pour chaque ensemble de données, nous avons exécuté l'algorithme 5 fois en faisant varier 2 paramètres : le système de pondération de termes et le nombre de clusters. Les résultats ont été distribués à nos experts qui ont noté chaque nom de cluster sur l'échelle suivante : 0 (zéro) = le nom du cluster est totalement dénué de sens, elle n'apporte aucune information sur le contenu des documents ; 1 (un) = le nom est assez intéressant ; 2 (deux) = le nom résume bien la teneur du groupe et apporte des informations utiles. Une fois les clusters notés, on calcule le pourcentage des noms de bonne qualité (2), de qualité moyenne (1) et de mauvaise qualité (0) pour chaque système de pondération de termes.

Les figures 4 et 5 présentent les résultats obtenus sur les deux corpus. Chaque barre représente la fréquence des niveaux d'appréciation (bon, moyen, mauvais) par niveau de gris décroissant, selon le codage utilisé.

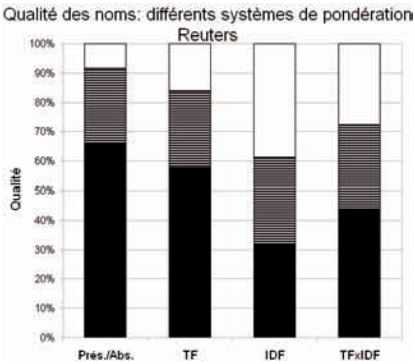

FIG. 4 – *Qualité des noms : différents systèmes de pondération (anglais)*

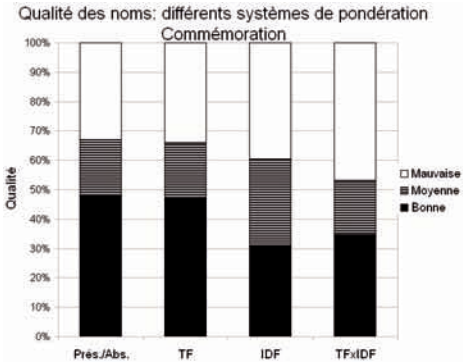

FIG. 5 – *Qualité des noms : différents systèmes de pondération (français)*

On observe que les codages en “Présence / Absence” et “Term Frequency” aboutissent aux noms jugés les meilleurs par les experts. Ceci est une confirmation expérimentale d’un fait théorique : la mesure IDF (également présente dans TFxIDF) pénalise les mots qui apparaissent dans de nombreux documents. Cette pénalisation, qui peut être utile pour construire les classes, perturbe la sélection des mots pour nommer les clusters : les expressions clés choisies doivent être représentatives du plus grand nombre de documents du cluster, et donc apparaître souvent dans les textes.

5 Conclusions et perspectives

5.1 Conclusions

Confronté à des quantités toujours plus importantes d’information textuelle, on cherche des moyens efficaces et rapides pour regrouper et synthétiser cette information. Le système que nous proposons permet de regrouper des documents et d’en extraire des thématiques compréhensibles. Les différentes approches proposées jusqu’ici laissent souvent de côté une caractéristique essentielle des textes en langage naturel : le fait qu’ils peuvent traiter de multiples sujets. C’est pourquoi les algorithmes de regroupement réduisent la qualité globale des résultats espérés s’ils ne prennent pas cette caractéristique en compte.

Nous avons donc recherché des solutions recouvrantes. A partir de l’algorithme l’OKM mis au point par Cleuziou (2007), nous avons essayé d’améliorer et de compléter les expériences antérieures avec différents systèmes de pondération des termes. Nous avons choisi une approche statistique pour l’extraction des expressions clés du texte pouvant fonctionner avec différentes langues. Les résultats obtenus en anglais et en français sur des ensembles de données de nature différente sont très encourageants et ouvrent des perspectives intéressantes.

5.2 Perspectives

Nous avons créé une application souple qui laisse place à de nombreuses perspectives, parmi lesquelles :

- **Comparaison avec de nouveaux algorithmes.** Nous pensons que les performances de l'algorithme OKM doivent être comparées avec des algorithmes de type SVD et LDA (cf. section 2.1). Le résultat du clustering n'étant pas de même nature, il faudra utiliser un seuil pour que les documents appartiennent exactement à un ou plusieurs clusters.
- **Meilleure association cluster - nom.** Dans l'état actuel de nos travaux, les noms sont associés aux catégories en se basant uniquement sur la similitude entre le centre de chaque cluster et le nom-candidat. Cela peut aboutir à donner des noms similaires à des catégories différentes. Nous pensons qu'il est possible de résoudre ce problème en prenant en compte la similitude avec les centres des autres clusters.
- **Mise a jour d'OKM.** wOKM Cleuziou (2009) est une version pondérée proposée récemment qui devrait permettre d'améliorer la qualité globale du processus de clustering.
- **Evaluation semi-automatisée des noms de cluster.** Nous pourrions davantage automatiser le processus d'évaluation par les experts. Nous envisageons une solution où on extraira tous les noms-candidats à partir des documents du corpus, puis on les fera noter indépendamment de l'algorithme de regroupement ; il sera alors possible d'évaluer automatiquement les noms extraits par notre système.
- **Plus d'expériences.** Nous envisageons, bien entendu, d'étendre les expérimentations réalisées sur des corpus plus grands, de styles différents et dans d'autres langues.

Références

Anaya-Sánchez, H., A. Pons-Porrata, et R. Berlanga-Llavori (2008). A new document clustering algorithm for topic discovering and labeling. In *CIARP '08 : Proceedings of the 13th Iberoamerican congress on Pattern Recognition*, Berlin, Heidelberg, pp. 161–168. Springer-Verlag.

Biskri, I., J. G. Meunier, et S. Joyal (2004). L'extraction des termes complexes : une approche modulaire semiautomatique. In *Données Textuelles, Louvain-La-Neuve, Belgique), Gérard Purnelle, Cédrick Fairon & Anne Dister (eds). Presses Universitaires de Louvain, Volume 1, pp 192201, ISBN*, pp. 2–930344.

Blei, D. M., A. Y. Ng, M. I. Jordan, et J. Lafferty (2003). Latent dirichlet allocation. *Journal of Machine Learning Research 3*, 2003.

Cleuziou, G. (2007). Okm : une extension des k-moyennes pour la recherche de classes recouvrantes. In M. Noirhomme-Fraiture et G. Venturini (Eds.), *EGC*, Volume RNTI-E-9 of *Revue des Nouvelles Technologies de l'Information*, pp. 691–702. Cépaduès-Éditions.

Cleuziou, G. (2009). Okmed et wokm : deux variantes de okm pour la classification recouvrante. In J.-G. Ganascia et P. Gancarski (Eds.), *EGC*, Volume RNTI-E-15 of *Revue des Nouvelles Technologies de l'Information*, pp. 31–42. Cépaduès-Éditions.

da Silva, J., G. Dias, S. Guilloré, et Pereira (1999). Using localmaxs algorithm for the extraction of contiguous and non-contiguous multiword lexical units. *Progress in Artificial Intelligence*, 849.

Dias, G., S. Guilloré, et J. G. P. Lopes (2000). Extraction automatique d'associations textuelles à partir de corpora non traités. In M. Rajman et J.-C. Chapelier (Eds.), *JADT 2000 - 5èmes Journées Internationales d'Analyse Statistique de Données Textuelles*, Volume 2, Lausanne, pp. 213–220. Ecole Polytechnique Fédérale de Lausanne.

Geraci, F., M. Pellegrini, M. Maggini, et F. Sebastiani (2006). Cluster generation and cluster labelling for web snippets : A fast and accurate hierarchical solution. pp. 25–36.

Hammouda, K. M., D. N. Matute, et M. S. Kamel (2005). Corephrase : Keyphrase extraction for document clustering. *MLDM 2005*, 265–274.

Larsson, N. J. (1998). Notes on suffix sorting. (LU-CS-TR :98-199, LUNDFD6/(NFCS-3130)/1–43/(1998)).

Myung, J., J.-Y. Yang, et S.-g. Lee (2009). Picachoo : a tool for customizable feature extraction utilizing characteristics of textual data. In *ICUIMC '09 : Proceedings of the 3rd International Conference on Ubiquitous Information Management and Communication*, New York, NY, USA, pp. 650–655. ACM.

Nikos, A. T., N. Fakotakis, et G. Kokkinakis (2002). Comparative evaluation of collocation extraction metrics. In *In Proceedings of the 3rd Language Resources Evaluation Conference*, pp. 620–625.

Osinski, S. (2003). An algorithm for clustering of web search results. Master's thesis, Poznań University of Technology, Poland.

Osinski, S. et D. Weiss (2004). Conceptual clustering using lingo algorithm : Evaluation on open directory project data. pp. 369–377.

Pons-Porrata, A., R. Berlanga-Llavori, et J. Ruiz-Shulcloper (2007). Topic discovery based on text mining techniques. *Inf. Process. Manage. 43*(3), 752–768.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program 14*(3), 130–137.

Roche, M. (2004). *Intégration de la construction de la terminologie de domaines spécialisés dans un processus global de fouille de textes*. Ph. D. thesis, Université de Paris 11.

Salton, G., A. Wong, et C. S. Yang (1975). A vector space model for automatic indexing. *Commun. ACM 18*(11), 613–620.

Velcin, J. et J.-G. Ganascia (2007). Topic extraction with agape. In *ADMA*, pp. 377–388.

Summary

Organizing textual data and derive meaning from it is a major challenge today. Thus, when one wishes to analyze an online discussion or forum, it is desirable to be able to quickly see which are the main themes and how the discussion is structured around them. From there and the observation that the text is generally associated with several themes, we propose a new method to combine the textual data by allowing overlaps, but also to assign a name to each group to appoint in the most readable way possible. The main contribution of this paper is a method which allows for the entire chain, starting from the raw textual data to the characterization of groups at a semantic level which exceeds the mere collection of words.