Automatic Textual Aggregation Approach of Scientific Articles in OLAP context

BOUAKKAZ Mustapha
LIM Laboratory
University of Laghouat, Algeria
m.bouakkaz@lagh-univ.dz

LOUDCHER Sabine
ERIC Laboratory
University of Lyon, France
sabine.loudcher@univ-lyon2.fr

OUINTEN Youcef
LIM Laboratory
University of Laghouat, Algeria
ouinteny@lagh-univ.dz

***Abstract*—In the last decade, Online Analytical Processing (OLAP) has taken an increasingly important role in Business Intelligence. Approaches, solutions and tools have been provided for both databases and data warehouses, which focus mainly on numerical data. These solutions are not suitable for textual data. Because of the fast growing of this type of data, there is a need for new approaches that take into account the textual content of data. In the context of Text OLAP (OLAP on text or documents), the measure can be textual and need a suitable aggregation function for OLAP operations such as roll-up. We present in this paper a new aggregation function for textual data. Our approach is based on the affinity between keywords and uses the search of cycles in a graph to find the aggregated keywords. We also present performances and a comparison with three other methods. The experimental study shows good results for our approach.**

***Keywords*-OLAP, textual data, aggregation function, graph.**

I. INTRODUCTION

In many complex fields, such as health, safety, security and transport, decision-makers require helpful indicators and tools to make decisions. Online Analytical Processing (OLAP) has emerged to assist users in the process of decision making. The model used in OLAP is based on the multidimensional structure and facilitates the navigation and the aggregation of data. The model represents both the subjects to analyze (facts), the indicators to assess the facts (measures) and the analysis axis (dimensions with hierarchies). For the navigation and the visualization, OLAP uses operations such as roll-up, drill-down, slice and dice. In a roll-up operation, in order to change the detail level of data and to aggregate the measure according to the hierarchy of a dimension, OLAP uses an aggregation function (sum, average, etc.).

The OLAP tools are effective when data are numerical but there are not suitable for unstructured data such as text. Because of the fast growing of textual data, there is a need for new approaches that take into account the textual content of data in OLAP analysis; and it is called Text OLAP. In Text OLAP, there are many open issues for handling the textual content of data. In this paper we are more interested by an aggregation function suitable for a textual measure such as a list of keywords. Our contribution is to provide an aggregation function of keywords. The methods for keyword aggregation can be classified into three categories. The first one is based on linguistic knowledge, the second one on external knowledge, while the last uses statistical methods. Our work falls in the latter category. The existing approaches using statistics focus mainly on the term frequencies. The originality of our approach is to use the graph theory in order to express the notion of affinity between two terms. We introduce a new aggregation approach called TAG for *Textual Aggregation by Graph*. More, we also present performances and a comparison with three other methods [12], [14], [11]. The experimental study shows good results for our approach.

The rest of the paper is organized as follows: Section 2 introduces the background of our work. Then, Section 3 is devoted to related work to textual aggregation. In Section 4, we present our contribution, followed by the presentation of the mostly used benchmarks in Section 5. In Section 6, we present the experimental study which includes a comparison of four approaches. Finally, Section 7 concludes the paper and gives future developments.

II. BACKGROUND

In the context of data warehousing, the traditional conceptual model is the multidimensional model [1]. More specifically, it is widely recognized that there are at least three specific notions that any conceptual data model for data warehousing should include: facts, measures and dimensions. A fact is the subject of decision-oriented analysis. The measure is an indicator to assess the facts. A dimension corresponds to a perspective under which facts can be analyzed [1]. A hierarchy is a set of attributes which represent different levels of granularity of a dimension. A typical example of hierarchy for the dimension *time* is *day* $\rightarrow$ *month* $\rightarrow$ *year*. When users want to navigate into multidimensional data, they need OLAP operations such as roll-up, drill-down, slice and dice. In a roll-up operation, in order to change the detail level of data and to aggregate the measure according to the hierarchy of a dimension, OLAP uses an aggregation function. In traditional OLAP, measures are numeric and the aggregation functions are *max*, *min*, *sum*, *average*, etc [2]. OLAP operations are applied on a data representation called date cube or cube [3]. A cube is a group of facts expressed with a measure and arranged by dimensions chosen by user.

We position our work in the context of Text OLAP (OLAP on text or documents) where measures or dimensions can be textual. For instance, we want to study scientific articles published by authors at a certain date, facts can be the ARTICLES. The measures can be numerical (such as the number of citations of the article) or textual (a bag of words like the title of the article or a list of keywords). In this example, we can find four dimensions: AUTHOR, CONFERENCE, TIME and DOCUMENT that contains the whole text of the article. DOCUMENT is a documentary or a textual dimension. The goal of Text OLAP is to take into account the textual content of data into OLAP analysis. In this article, we are more interested by textual measures such as a list of keywords. The classical aggregation functions operation are adapted only for numerical measures and they are not suitable for textual measures. So there is a need for a textual aggregation function.

III. RELATED WORK

The approaches, which describe a corpus of documents through the most representative keywords, found in the literature can be classified into three categories. The first one is based on linguistic knowledge; the second one is based on the use of external knowledge, while the last uses statistical methods.

The approaches based on linguistic knowledge consider a corpus as a set of the vocabulary mentioned in the documents; but the results in this case are sometimes ambiguous. To overcome this obstacle, techniques based on lexical knowledge and syntactic knowledge previews have been introduced. In [5][6] the authors described a classification of textual documents based on scientific lexical variables of discourse. Among these lexical variables, they chose nouns because they are more likely to emphasize the scientific concepts, rather than adverbs, verbs or adjectives.

The approaches based on the use of external knowledge select certain keywords that represent a domain. These approaches often use models of knowledge such as ontology. Ravat et al. proposed an aggregation function that takes as input a set of keywords extracted from documents of a corpus and that outputs another set of aggregated keywords [7]. They assumed that both the ontology and the corpus of documents belong to the same domain. Oukid et al. proposed an aggregation operator Orank (OLAP rank) that aggregated a set of documents by ranking them in a descending order using a vector space representation [8].

The approaches based on statistical methods, use the occurrence frequencies of terms and the correlation between terms. Landauer et al. proposed the method LSA (Latent Semantic Analysis) in which the corpus is represented by a matrix where the rows represent the documents and the columns represent the keywords [9]. An element of the matrix represents the number of occurrences of a word in a document. After decomposition and reduction, this method provides a set of keywords that represent the corpus. Hady et al. in [10] proposed an approach called TUBE (Text-cUBE) to discovering associations among entities, the model adopts a concept similar to data cube designed for relational databases and is applied to textual data, which cell contains keywords, and they attach to each keyword an interestingness value. Bringay et al. proposed two aggregation functions, the first one is based on a new adaptive measure of tf.idf which takes into account the hierarchies associated to the dimensions [11]. The second one is build dynamically and is based on clustering. Christian et al. [12] proposed another method called TOPIC in which they use the standard k-means clustering algorithm as described in [13]. Their method starts with the selection of two elements for the two first clusters. The other terms are then assigned to the cluster they are closest. Ones all the terms have been assigned, the process is repeated for each cluster with a diameter larger than a specified threshold value.

Ravat et al. [14] proposed a second aggregation function called TOP-Keywords to aggregate keywords extracted from a corpus. They compute the frequencies of terms using the tf.idf function, then they select the first k most frequent terms. El-Ghannam et al. in [15] propose a technique for extracting summary sentences for multi-document using the weight of sentences and documents.

The approaches of the two first categories use additional information (linguistic and external) and the ones of the third category rely on the choice of the number of keywords k to represent a corpus. The approach we propose does not need additional information as it falls in the third category. Furthermore it does not need the specification of k in advance.

To evaluate the performance of the cited approaches, three evaluation metrics are mainly used which are recall, precision and the F-measure. The recall is the ratio of the number of relevant documents retrieved to the total number of relevant documents assigned by aggregated keywords. Precision is the ratio of the number of relevant documents retrieved to the total number of irrelevant and relevant documents retrieved assigned by aggregated keywords. The F-measure, which combines precision and recall, is the harmonic mean of precision and recall, the traditional F-measure or balanced F-score is defined as follows:

$$F = 2.\frac{recall.precision}{recall + precision} \tag{1}$$

IV. OUR CONTRIBUTION

In order to create a suitable environment for the online analysis of textual data, we intend to propose a new method which performs aggregation of keywords of documents based on graph theory. This function produces the main aggregated keywords out of a set of terms representing a corpus. In the rest of this paper we focus on the aggregation of keywords extracted from scientific articles using the keyword measure. Our aggregation approach, that we call TAG (Textual Aggregation by Graph), aims at extracting from a set of terms a set of the most representative keywords for the corpus using a graph. The function takes as input the set of all extracted terms from a corpus, T, and output an ordered set, KW, containing the most representative keywords. Thus we have KW $\subset$ T. The process of aggregation goes through the following steps: (1) Extraction of keywords with their frequencies, (2) Construction of the affinity matrix and the affinity graph, and (3) Cycle construction and aggregated keywords selection.

A. *Extraction of keywords*

The set of terms T is obtained after the analysis of the documents of a corpus, cleaning stop words, lemmatization and the selection of the most significant terms. There are different ways to select such terms. We use the weight (frequency) of a term as a criterion for the selection of the most commonly used terms. The weight of a term represents the degree of its importance in the document. In our case we take the terms with frequencies greater than 30% i.e.

$$\forall t_i \in T, w_i = \frac{TF_i}{\sum TF_i} \qquad (2)$$

Where w_i is the weight of term t_i, TF_i is the frequency of occurrence of term t_i in the corpus.

B. *Construction of the affinity matrix and the affinity graph*

In order to build the affinity matrix between terms (keywords), we need to compute the frequency matrix of term (FM) where the rows represent the documents and the columns represent the terms. An element of the matrix represents the frequency of a term in a document. Let $FM(i,j) = TF_{ij}$ where TF_{ij} is the frequency of occurrence of term t_j in document d_i.

The affinity matrix $(AffM)$ between keywords is obtained from the frequency matrix, which is a symmetric square matrix where the rows and columns represent the keywords. It is defined as follows:

$$f(x) = \begin{cases} \sum_k TF_{kj} \quad if \quad i = j \\ \sum_k (TF_{ki} + TF_{kj}) \ else \end{cases}$$

The diagonal of AM contains the frequency TF_{ij} of the term t_j in the document d_i. The other elements, define the degree of affinity of each pair of keywords as the sum of TF_{ki} and TF_{kj} for only the values of k where TF_{ki} and TF_{kj} are different from 0. The affinity graph is formally defined by a weighted graph $G = (N, L, f)$. where $N = t1, t2, ..., tn$ is the set of nodes of G, representing the terms of the corpus; L is the set of edges defined by the couples (t_i, t_j) and f is a weigh function define by $f(t_i, t_j) = AffM[i, j]$.

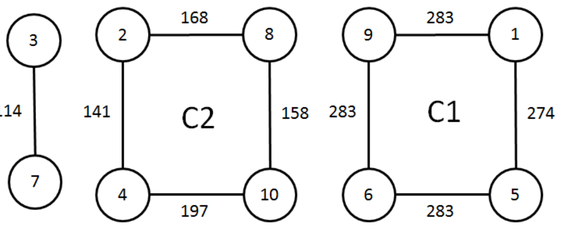

Figure 1. Affinity cycles obtained with TAG

C. *Cycle construction and aggregated keyword selection*

Our approach consists of finding p cycles with big weights, then selecting the one with the highest weight. The nodes of such a cycle will give us the aggregation of the initial keywords. The process of automatically finding the aggregation of keywords using the affinity graph is described by the following steps.

1) Set $p = 1$
2) Start with an empty cycle $C_p = \emptyset$, and set $m = 1$.
3) Select a first node t_i randomly from N. set $c_m = t_i, C_p = C_p \cup \{c_m\}, m++$.
4) if $m < 2$ then find $t_i \neq t_j / f(t_i, t_j) = Maxf(t_i, t_k)$ for $t_k \neq t_i$ else find $t_i \neq t_j$ and $t_j \neq c_{m-1} / f(t_i, t_j) = Maxf(t_i, t_k)$ for $t_k \neq t_i$ and $t_k \neq c_{m-1}$
5) $c_m = t_j, C_p = C_p \cup \{c_m\}, m++$.
6) If $c_m \notin C_p$ then goto 4, else $N = N - C_p, p{+}{+}$ goto 2.

After the construction of the different cycles we calculate the average affinity for each cycle, and we select the cycle that has the highest average. For example, if we have a cycle C_p that has three keywords w_1, w_2 and w_3 the average affinity associated to C_p is calculated as follows in Equation 3.

$$AVG(C_p) = \frac{AffM[w_1, w_2] + ... + AffM[w_3, w_1]}{3} \qquad (3)$$

D. *Example*

We take an example in which we analyze the major keywords of scientific articles according to their author and the year of publication. This example consists of thirteen (13) documents $D_1, ..., D_{13}$ and ten (10) terms: $(T1 = OLAP), (T2 = XML), (T3 = Datamining), (T4 = Query), (T5 = datawarehouse), (T6 = Document), (T7 = System), (T8 = Function)(T9 = Cube), (T10 = Network)$. The frequency matrix is defined in Table I. The computed affinity matrix between keywords given in Table II, defines the affinity graph which is in this case a complete graph. The application of the algorithm described above produces two cycles C_1 and C_2 as shown in Figure 1.

Table I
FREQUENCY MATRIX (FM)

	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10
D1	10	9	22	15	9	20	15	9	28	39
D2	15	22	26	0	9	16	11	0	25	0
D3	5	15	0	15	22	0	15	0	0	0
D4	0	16	0	0	15	10	0	0	0	0
D5	16	12	2	13	16	12	0	12	2	0
D6	21	0	19	21	17	9	0	0	10	0
D7	13	0	14	0	0	15	1	0	17	0
D8	17	0	8	0	0	8	0	18	20	0
D9	22	14	0	0	14	21	0	17	0	0
D10	0	7	0	0	7	0	15	18	20	0
D11	5	18	10	5	15	15	15	18	20	0
D12	20	4	7	17	4	7	0	5	3	105
D13	1	10	11	1	10	17	0	16	10	0

Table II
AFFINITY MATRIX (AFFM)

	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10
M1	145	198	259	180	274	200	119	125	283	175
M2	198	127	135	141	200	280	120	168	197	157
M3	259	135	119	143	155	238	114	112	242	173
M4	180	141	143	87	136	152	80	111	145	197
M5	274	200	155	136	129	283	117	171	199	157
M6	200	280	238	152	283	150	108	170	283	171
M7	119	120	114	80	117	108	67	57	132	54
M8	125	168	112	111	171	170	57	95	161	158
M9	283	197	242	145	199	283	132	161	143	175
M10	175	157	173	197	157	171	54	158	175	144

We compute the average affinity of the two cycles :

$$AVG(C_1) = \frac{283 + 274 + 283 + 283}{3} = 280.75 \quad (4)$$

$$AVG(C_2) = \frac{168 + 158 + 197 + 141}{3} = 166 \quad (5)$$

Then we select C_1. The thirteen documents of the example are, thus, represented by the following keywords: {OLAP, Datawarehouse, Document, Cube}.

V. TEXTUAL BENCHMARKS

There are several publicly available benchmarks for evaluating keywords. Hulth in [16] use as dataset to test their approach containing 800 journal article abstracts from Inspec[1], published between the years 1998 and 2002. Nguyen and Kan in [17] compiled a dataset containing 120 computer science articles from 4 to 12 pages. Wan and Xiao in [18] developed a dataset of 308 documents taken from DUC 2001. Schutz in [19] compiled a collection of 500 medical articles from PubMed[2]. Krapivin et al. [20] used 680 articles from the same source from 2003 to 2005, with author assigned keywords. Su Nam Kim in [21] collect a dataset of 100 articles from the ACM Digital Library (conference and workshop papers), ranging from 6 to 8 pages, including tables and figures. Medelyan et al. in [22] propose a tool that generates automatically a dataset using keywords assigned by users of the collaborative citation platform CiteULike [3]. These corpuses are summarized in table III.

[1]http://www.theiet.org/resources/inspec/
[2]http://www.ncbi.nlm.nih.gov/pubmed/

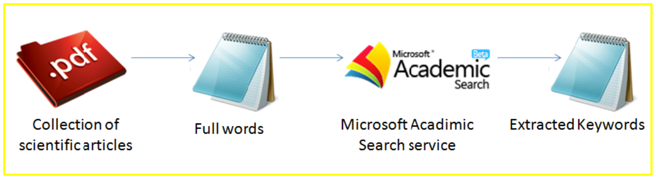
.pdf
Collection of scientific articles
Full words
Microsoft Academic Search
Microsoft Acadimic Search service
Extracted Keywords

Figure 2. The structure of our corpus generator

Table III
EXISTING BENCHMARKS

References	Corpus size
Hulth A. [17]	800
Nguyen and Kan [18]	120
Wan and Xiao [19]	308
Schutz [20]	500
Krapivin et al. [21]	680
Su Nam Kim [22]	100

In this work we compiled a corpus from the IIT conference [4] (conference and workshop papers) for the years 2008 to 2012. It consists of 600 papers ranging from 7 to 8 pages in IEEE format, including tables and figures. The keywords are extracted from the full words using Microsoft Academic Search [5] keywords. This process is illustrated in figure 2.

VI. EXPERIMENTAL STUDY

In this section, we report an empirical study to evaluate our aggregated keywords function using a real corpus and compare its performance with that of BienCube [11], TOPIC [12] and TOP-keywords [14]. We have implemented the four algorithms BienCube, TOPIC, TOP-Keyword, and our proposed function TAG using JAVA language. The experimentation was performed on a PC running the Microsoft Windows 7 Edition operating system, with a 2.62 GHz Pentium Dual-core CPU, 1.0 GB main memory, and a 300 GB hard disk. To test and compare the different approaches we have compiled a real corpus prepared in the previous section with 600 articles, 800000 words and 2182 keywords extracted.
To perform this comparison, we use three evaluation metrics, recall, recession and the F-measure. We also give the comparison of the complexities and an estimation of the runtime for the four algorithms. The results obtained from

[3]http://www.citeulike.org/
[4]http://www.it-innovations.ae/iit2014/index.html
[5]http://www.academic.research.microsoft.com/

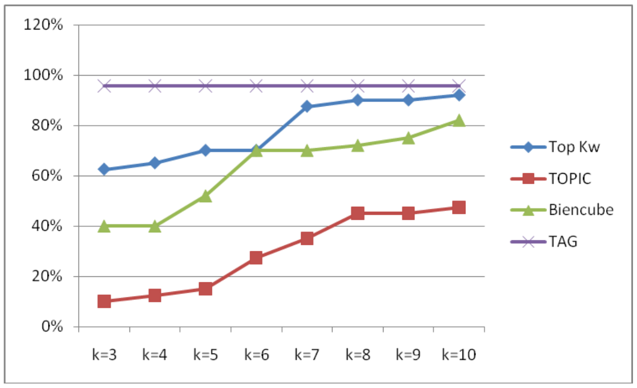

Figure 3. Comparaison of recall for the four approches

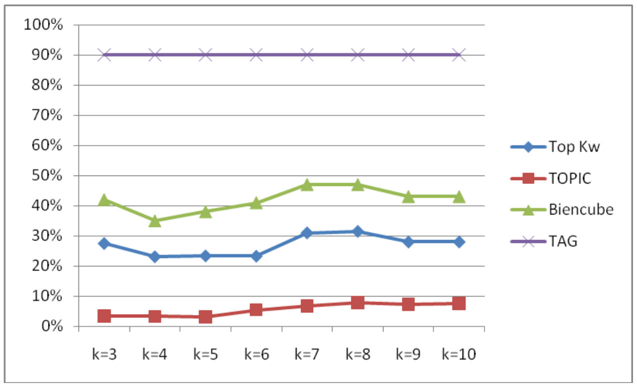

Figure 4. Comparaison of precision for the four approches

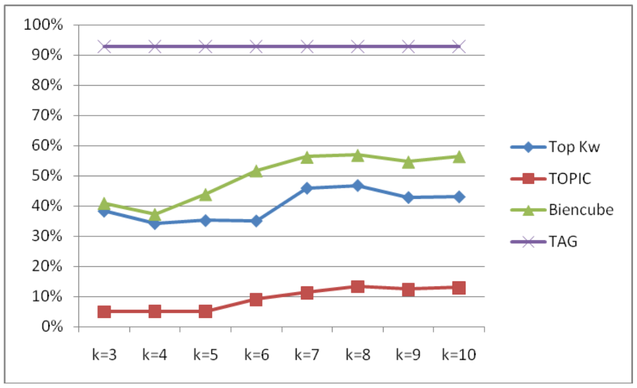

Figure 5. Comparaison of F-mesure for the four approches

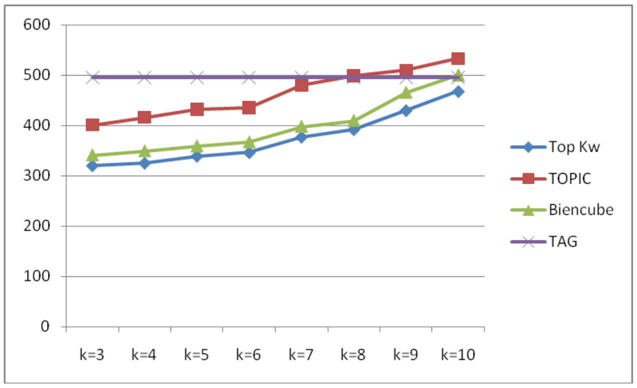

Figure 6. Comparaison of Runtim for the four approches

the four approaches are summarized in Figures (3), (4), (5) and (6). We note that our approach does not depend on a chosen number of aggregated keywords k, which means that a user does not need to specify the value of k in advance. It produces automatically the number of aggregated keywords with high precision. Foremost, our approach produces highest values of the recall, the precision and F-measure. For instance, in the case of k=3, we obtained a recall of 96% compared with 63%, 10% and 40% obtained by Topkeyword, TOPIC and BienCube respectively. We also obtained a precession of 90% compared with 28%, 3% and 43% obtained by Topkeyword, TOPIC and BienCube respectively. As for the F-measure, we obtained a value of 93% compared with 38%, 5% and 4% obtained by Topkeyword, TOPIC and BienCube respectively. In the case of k=10, the value we obtained of 96% is to be compared with 63%, 10% and 40% obtained by Topkeyword, TOPIC and BienCube respectively. The precession obtained of 90% is compared with 28%, 3% and 43% obtained by Topkeyword, TOPIC and BienCube respectively. As for the F-measure, the value of 93% is compared with 38%, 5% and 4% obtained by Topkeyword, TOPIC and BienCube respectively.

The four approaches have the same complexity which is $O(n^2)$. The complexity of TOP-Keyword and BienCube is that of a sort algorithm which is $O(n^2)$ [7][11], the complexity of TOPIC is that of the k-means algorithm which is $O(n^2)$ [12]. The complexity of our approach TAG is that of the construction of the affinity cycles which is $O(n^2)$. In Order to determine the runtime for each approach we carried out 10 executions of each algorithm. The average runtime of these executions are represented in the Figure 6. We note that the runtime of our approach stays the same because it does not depend on k, while the runtime of the other three approaches increase with k.

VII. Conclusion

In this research , we have introduced a new aggregation algorithm for textual data in OLAP context. Unlike traditional aggregation used in Information Retrieval for textual data that mainly relies on keyword frequency information, our aggregation function uses the strength of the affinity and the similarity between the keywords of a corpus. We used the affinity between keywords taken by pairs to define an affinity graph. The extraction of the aggregated keywords is performed though the search of cycles in that graph. Our experimental results on a real corpus show better performances for our approach TAG. Furthermore, unlike TOP-Keyword, BienCube and TOPIC aggregation functions, our function does not require the specification of the number of aggregated keywords in advance. We used the recall, precision, F-measure, runtime and the theoretical complexity as an aggregated keywords criteria to compare our approach with the three mentioned approaches. As future work, we intend to look for other factors of comparisons and try to introduce the semantic aspect of keywords as well using other corpuses.

References

[1] R. Torlone, Conceptual multidimensional models, Multidimensional Databases: Problems and Solutions, Idea Group,2003, pp. 69-90.

[2] R. Kimball , The data warehouse toolkit, Ed. John Wiley and Sons, 1996, 2 ed. 2003.

[3] The OLAP Council, the OLAP glossary 2013, Available: http://www.olapcouncil.org.

[4] O. Teste , Modelisation et manipulation d'entrepots de donnees complexes et historisees. PhD theses, Paul Sabatier university, Toulouse, 2000.

[5] C. Poudat et al., Categorisation de textes en domaines et genres, Complementarite des indexations lexicale et morpho syntaxique, Lavoisier-Hermes, 2006, pp. 61-76.

[6] U. Kohomban and W. Sun Lee, Optimizing Classifier Performance in Word Sense Disambiguation by Redefining Sense Classes, International Joint Conference on Artificial Intelligence , 2007, pp. 1635-1640.

[7] F. Ravat et al., OLAP Aggregation Function for Textual Data Warehouse, International Conference on Enterprise Information Systems, 2007, pp. 151-156.

[8] L. Oukid et al., CXT-Cube: Contextual Text Cube Model and Aggregation Operator for Text OLAP, presented at DOLAP, 2013, San Francisco USA.

[9] T. Landauer et al., An introduction to Latent Semantic Analysis (LSA). Discourse Processes, 1998, pp.259-284, 1998.

[10] W. Hady Lauw et al., TUBE (Text-cUBE) for discovering documentary evidence of associations among entities, the 22nd Annual ACM Symposium on Applied Computing, Seoul, Korea, 2007, pp. 824-828.

[11] S. Bringay et al., Bien cube: les donnees textuelles peuvent s'agreger, Conference internationale sur l'extraction et la gestion des connaissances, Hammamet, Tunisia, 2010, pp 585-596.

[12] C. Wartena and R. Brussee, Topic detection by clustering keywords. DEXA, 2008, pp. 54-58.

[13] F. Archetti and P. Campanelli, A hierarchical document clustering environment based on the induced bisecting k-means, H.L.Larsen, volume 4027 of Lecture Notes in Computer Science, Springer, 2006, pp. 257-269.

[14] F. Ravat et al., Top keyword extraction method for OLAP document, International Conference on Data Warehousing and Knowledge Discovery, 2008, volume 5182, Springer Verlag, 2008, pp. 55-64.

[15] F. Elghannam and T. Elshishtawy, MultiTopic MultiDocument Summarizer, International Journal of Computer Science Information Technology, Vol. 5 Issue 6, Dec 2013.

[16] A. Hult, Improved automatic keyword extraction given more linguistic knowledge, the Conference on Empirical Methods in Natural Language Processing, Japan, 2003, pp. 216-223.

[17] T. Nguyen and M. Kan, Key phrase Extraction in Scientific Publications. International Conference on Asian Digital Libraries. Hanoi, Vietnam, 2007, pp. 317-326.

[18] X. Wan and J. Xiao, CollabRank: Towards a Collaborative Approach to Single Document Keyphrase Extraction, the International Conference on Computational Linguistics, Manchester, UK, 2008, pp. 969-976.

[19] A. Schutz, Keyphrase Extraction from Single Documents in the Open Domain Exploiting Linguistic and Statistical Methods, Master thesis, National University of Ireland, 2008.

[20] M. Krapivin and M. Marchese, Large Dataset for Keyphrases Extraction, Technical Report DISI-09-055, University of Trento, 2009.

[21] K. Sunam et al., Automatic Keyphrase Extraction from Scientific Articles, Language Resources and Evaluation, Springer, Verlag New York, 2013, pp.723-742.

[22] O. Medelyan et al., Human-competitive tagging using automatic keyphrase extraction, the Conference on Empirical Methods in Natural Language Processing. Singapore, 2009, pp. 1318-1327.