AJAS / SPAS

African Journal of Applied Statistics
Vol. 5 (1), 2018, pages 377–401.
DOI: http://dx.doi.org/10.16929/ajas/377.221

ISSN 2316-0861

Classification of the Gambian Online Newspapers by keywords: an unsupervized study and Data Streaming Platform

Babanding Sanyang [1], **Saidou MS Badjie**[2] **and Gane Samb Lo**[3,*]

[1] Senior Forestry Officer, Head of Communication & Extension Unit, Department of Forestry, 5 Marina Parade, Banjul, The Gambia
[2] Planning and Policy Unit, Ministry of Energy and Petroleum, Petroleum House, Brusubi Roundabout, Bijilo, West Coast Region The Gambia, West Africa
[3] LERSTAD, Gaston Berger University, Saint-Louis, Sénégal (main affiliation).
LSTA, Pierre and Marie Curie University, Paris VI, France.
AUST - African University of Sciences and Technology, Abuja, Nigeria
gane-samb.lo@edu.ugb.sn, gslo@aust.edu.ng, ganesamblo@ganesamblo.net
Permanent address: 1178 Evanston Dr NW T3P 0J9,Calgary, Alberta, Canada.

Received on June 01, 2018; Accepted on July 15, 2018; Published Online on July 27, 2018

Abstract. In this paper, we begin a regional project of knowledge retrieval process from African online newspapers. We first focus on the Gambian context and undertake an unsupervized learning process from such journals. (See page 378 for the full abstract).

Key words: Data-Mining; Web-Mining; Keywords and patterns; Knowledge retrieval; Unsupervized learning process; Statistical learning; Computer packages
AMS 2010 Mathematics Subject Classification: 62-07; 62H30; 60-04; 60-08;

Presented by Olusegun Sunday Ewemooje, PhD, Federal University of Technology, Akure, Nigeria
Corresponding Member of the Editorial Board.

---

*Corresponding author Gane Samb Lo: gane-samb.lo@ugb.edu.sn, gslo@aust.edu.ng
Babanding Sanyang: sanyangbaba@yahoo.com
Saidou MS Badjie: saidoumsbadjie@yahoo.com

Full Abstract In this paper, we begin a regional project of knowledge retrieval process from African online newspapers. We first focus on the Gambian context and undertake an unsupervized learning process from such journals. With the help of appropriate and specifically designed computer packages, we study the keywords that likely discriminate the categories of articles (agriculture, health, politics, etc). We found 681 words that would efficiently help building a very efficient classifier of categories of articles and that would serve to building a metric from which regular classification or Big data classification methods are operated. Future studies are announced.

Résumé (French) Dans cet article, nous commençons un projet régional de processus de recherche de connaissances à partir de journaux en ligne africains. Nous nous concentrons d'abord sur le contexte gambien et entreprenons un processus d'apprentissage non supervisé à partir de ces revues. Avec l'aide de logiciels appropriés et spécialement conçus à cet effet, nous cherchons à déterminer les mots-clés qui discriminent les catégories d'articles (agriculture, santé, politique, etc.). Nous avons trouvé 681 mots qui pourraient aider à construire un classificateur très efficace de catégories d'articles et qui servirait à construire une métrique à partir de laquelle des méthodes de classification régulière ou de classification de données massives (Big Data) sont appliquées. Le succès de l'étude est un prétexte pour établir une plate-forme de flux de données des régions données du monde, Afrique par exemple, pour la détection continue de mots-clés et de leur adoption.

1. Introduction

Data mining is rephrased as Web mining when applied to the web context. Its main objective is retrieving information or patterns from Data. As such, it may be claimed that Statistics is part of Data Mining. Rather, it is formed by a number of sub-disciplines which have the common goal of finding features and knowledge from data, but each of them may have its own paradigm, methods and underlying theory. For example, while Statistics may serve as a way to get knowledge from observations through its *descriptive statistics* branch, it is expanded into a mathematical sub-field based on highly theoretical orientation.

In its most common conception, Data mining is seen as a set of algorithms destined to extract knowledge from the data and to test that knowledge and, to recommend or not its use as decision tools. The data are so complex and, some time for example, the appropriate statistical theory to handle such data does not exist yet. Recently, the functional data approach appeared in the statistical terminology as an attempt to handle data depending on the time, out of a time series analysis.

Also, in the context of the Web, the data might be huge or massive and we speak of Big data. As well, data might arise from a continuous process and we receive it such that neither we can store all of it nor we are able to analyse it at the time we receive it. We speak about Data streaming. Both Big data and Data Streaming are

two major concerns of Web mining.

Since it is not possible to exactly compute or to exhaustively describe the objects, we are interested in approximated solutions, without paying a great cost of time we cannot afford. We then search algorithms which are much less time consuming and provide acceptable estimations of the objects. The different algorithms which are available for each specific study compete with respect to the less time used and the less error of the estimation.

Although there are many references on the subject, the book of Leskovec *et al.* (2014), which can be found in the website *mmmds.org*, is a very appropriate introduction to Web mining as it is used here.

In this study we are interested in text data from online newspapers. Specifically, we aim to know if articles have specific keywords by categories, meaning that for any category of articles (like *Sports*, *Politics*, etc.), there are typical words that are more likely to be found in articles in that category and are less likely to be read in other categories.

This problem is more general and may be addressed on the whole set of web-pages, when divided into categories, or on any subclass of text data. Also, in other types of data, words may be replaced by other keywords while the approach will be the same. So the paper, beyond its actual scope, may serve as a model for handling other classes of text data or other type of data.

As a beginning, we focus on text data, that is articles sampled form online newspapers. The main question becomes: is-it possible to empirically reconstitute the categories by a classification method?

Let us place ourselves in the Gambian context and suppose we use the six categories: Agriculture, Business, Health, Politics, Society and Sports. Suppose that we have collected a number of articles from the web-pages of a set of newspapers and that we have applied a classification method that stops when six classes have been formed. Can we expect that each class will contain only articles of the same category? We refer to this success as a perfect classification.

The following example is given in Leskovec *et al.* (2014). The ordered pair of measurements (height, weight) concerning a number of dogs are studied. The dogs are of three varieties: (A) chihuahuas, (B) beagles, (C) dashshunds. The representation of the coordinates as in Figure 1 clearly suggests that a perfect classification is possible.

But a such perfect classifications are rare and, we usually face a rate of misclassification which may be more or less large.

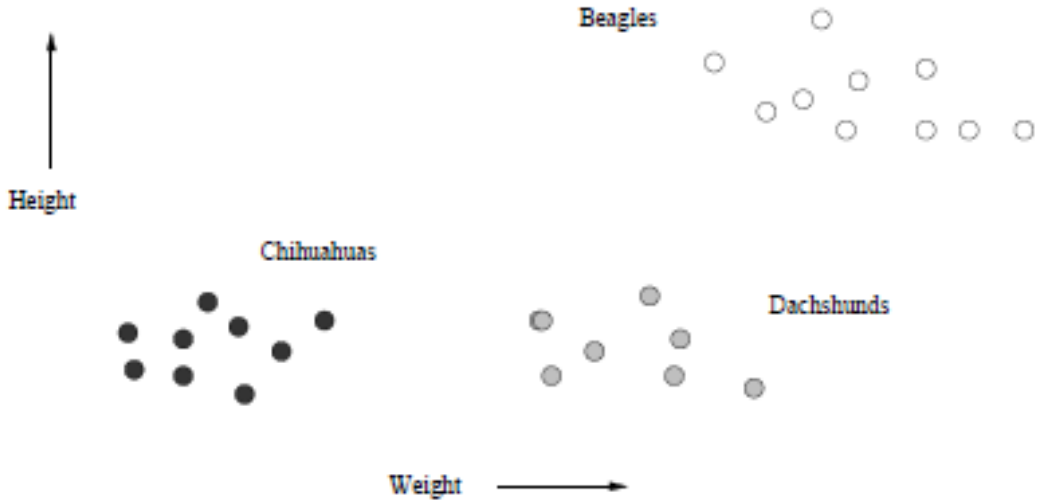

Fig. 1. Representation of heights and weights of dogs

In order to apply classification methods, we need a metric d on the texts. One of the most common metric on texts is the Jacquard distance. The edit distance is also useful. Let us use the first cited just to illustrate our point. The Jacquard metric between two sets $sh(A)$ and $sh(B)$ is simply defined by

$$J(sh(A), sh(B)) = \frac{Cardinality\ of\ sh(A) \cap sh(B)}{Cardinaly\ of\ sh(A) \cup sh(B)}.$$

Actually J is applied to two sets $sh(A)$ and $sh(B)$ which are formed from files A and B. Each of them is treated as follows. For example, to form $sh(A)$, we begin by getting rid of the little words (of less than three digits) and the blanks from A which and we get a global string variable, denoted as $string(A)$, of length ℓ. Next an integer k is fixed ($k = 3$ or $k = 4$, ...). Finally the string $string(A)$ is used to build the set $sh(A)$ as follows: the first elements of $sh(A)$ is the word formed by the first k letters of $string(A)$, the second word is formed by the first k letters from the second letter of $string(A)$, the third element is the word formed by the k letters from the third letter of $string(A)$, etc. We get $\ell - k + 1$ elements in $sh(A)$.

Although the simplicity of its mathematical formula, the Jacquard is over-time consuming. For example, computing it for two texts of the size of a Gospel requires around 10 minutes. This is a considerable time in the Web context where we have to compute a large number of distances at one time. So applying a classification method on texts of the cited size would require a considerable time since a classification includes quite a number of distances to compute. The reader is referred to Leskovec *et al.* (2014) as a general introduction on similarity distances

and to Soumaila and Lo (2015) and to Soumaila and Lo (2017) for an empirical study of the Jacquard distance of texts in real life and on its probabilistic aspects.

Of course, in the context of web mining, the *Rajaraman and Ulman* algorithm is applied as done in Soumaila and Lo (2015) and in Soumaila and Lo (2017). Although it gives interesting results with acceptable approximations in small times, we think that we may do far better if we exploit extra-knowledge provided by a preliminary study of the context.

This will be the case in our study if, through an empirical preliminary study, we are able to have for each category j, a set

$$\mathcal{W}_j = \{w(h,j),\ h = 1, 2, ...\},$$

of keywords, which are given such that the frequency of a keyword $w(h,j)$ is greater than that of the keyword $w(h+1,j)$. This implies that $w(1,j)$ is the most frequent word for articles from category j. Let us denote the relative frequencies of those keywords by

$$\mathcal{F}_j = \{freq(h,j),\ h = 1, 2, ...\}.$$

Now, if the following facts

(C1) the keywords are discriminatory enough between the categories

and

(C2) the differences of two frequencies of common keywords for two categories are also significant,

are validated by the data, we will be able to have a powerful tool for comparing categories and classifications based on the keywords and their frequencies can be implemented. This, surely, should lead to small misclassification rates.

We may even create a metric based on the set $\mathcal{W}_Q$ of all keywords of position not greater than Q in their category:

$$\mathcal{W}_Q = \{w(h,j),\ 1 \leq h \leq Q,\ 1 \leq j \leq number\ of\ categories\}.$$

For an article f, we denote $c(f,h,j)$ the (relative) frequency of the keyword $w(h,j)$ in f. Let *nbrCateg* be the number of categories. For two articles f_1 and f_2, we define the pseudo-metric

$$d_Q(f_1,f_2) = \left(\sum_{1 \leq j \leq nbrCateg} \sum_{1 \leq h \leq Q} \left(\frac{c(f_1,h,j) - c(f_2,h,j)}{c(h,j)} \right)^2 \right)^{1/2}. \quad (1)$$

It is easy to see that the distance d_Q is surely discriminatory between the categories if Conditions (C1) and (C2) above are validated. The choice of Q is important. If

the first keywords are really discriminatory between categories, we will be able to take Q to be as small as possible. This is a dimension reduction problem. As well, the choice of a small Q leads to shorter computation time.

The main objective is to carry on an supervised learning on data collected from the Gambian online newspapers and to try to scrutinize conditions (C1) and (C2) above. Upon success, the distance $d_Q(f_1, f_2)$ becomes an efficient tool for classification purposes on the text data class under study.

Indeed, for the Gambian data, the results are striking. The keywords, as listed in Figures 2 and 3 by category, are really discriminatory for the categories with very low dimensions. Even for $Q = 1$, the classification gives good trends. For $Q = 2$ or $Q = 3$, the learning misclassification rates are very low.

The rest of the paper is organized as follows:

In Part (A) of this paper, we undergo an unsupervised learning process or simply an exploratory study. In this part, we describe the data, their codification, and the algorithms we are going to use. It is written in such a manner that the reader who is interested in replicating the techniques on his own computer language will be able to do so. As a matter of fact, we wish that the Gambian study might be replicated in other countries for comparison purposes and enrichment of the keywords sets.

In part (B), we describe how the Gambian study might be the beginning of larger platform covering several countries, using a data streaming setting. That part can be seen as recommendations of Part (A) towards a continuous study on this kind of data in the context of data streaming.

Part (A): Unsupervised Learning Process.

As promised, let us start by the description of the data, which are collected from the online newspapers from Gambia. Here, each data is an article in the form of a text file. In a text file, we have letters and each letter takes the place of a bit in the *RAM*. In our study, we select six categories:

(a) Agriculture (1), Business (2), Health (3), Politics (4), Society (5) and Sports (6),

and three newpapers (outlets)

(b) Foroyaa, The point, Standard.

The total number of files is 654, distributed over the outlets and the categories as on Table 1.

Type/newspapers	Foroyaa	The Points	Standard	Total
Agriculture	37	10	17	64
Business	36	11	26	73
Health	50	30	28	108
Politics	50	49	50	149
Society	50	50	10	110
Sports	50	50	50	150

Table 1. Distribution of articles

Our first guess is that it would be possible to classify the articles using the frequency of keywords. To assess that hypothesis, methods based on automatic classifications or clustering may be applied.

A successful answer to this problem, meaning we might determine the categories of an article that would likely interest a web reader based on the words he/she submits when researching them in a search platform as Google, would be an interesting thing as recommendation systems. Such systems are automatically used by the HighTech companies like Netflix (movies platform), Yahoo (information Platform), Youtube (videos platform), etc.

Up to our knowledge, our study is original and has not been preceded by a similar one, at least for African data. As a consequence, we need to proceed to an unsupervised learning process. The *mining* or the *knowledge* we will retrieve from our data will be important facts to be stored and used for any further study in the Gambia, or in the African Area.

From the obtained results, we will base general methods and strategies in order to apply some usual algorithms of Data mining.

We also mention that we do not know whether or not there exists suitable computer programs which handle our objects as we wish it here. We then have to produce our own packages. This computer science is part of our studying on time consuming constitutes also an important contribution.

2. Data Exploration and Computational Aspects

Let us begin by explaining the organization of the data. This will help people interested in applying our program to other areas, using the same Visual Basic Software or adapting our algorithms into their own software (Python, R, etc.).

We fix the number of categories (nbrTyp=6) and give them names

```
typ(1) = "agric" : (for agriculture)
typ(2) = "bness" : (for Economy and business)
typ(3) = "hlth"  : (for health)
```

```
typ(4) = "pol"    : (for politics)
typ(5) = "soc"    : (for society)
typ(6) = "sports": (for sports),
```

next, the number of chosen outlets - online newspapers - (nbrNews = 3) and store their names in the array below

```
news(1) = "foroyaa"
news(2) = "point"
news(3) = "standard"
```

For each outlet i and category j, $nbrArt(i, j)$ is the number of articles from Outlet i of Category j. The numbers are displayed in Table 1.

An easy way to rename the file that facilitates their use in computer programs is the following: rename the file corresponding to the outlet j, to the category i and the k^{th} among them by

news(j)_typ(i)_k.txt.

If k has one digit as $1, ..., 9$, we add '0' before k. Here are some examples:

```
foroyaa_agric_05.txt}
standard_pol_09.txt
standard_hlth_12.txt
point_bness_49.txt
```

Another representation of the file $f = news(j)\_typ(i)\_k.txt$ that will be useful in the next lines, is the key

$$key(f) = j * 1000 + 100 * i + k, \tag{2}$$

which never gives the same value to two different files.

Basic Facts.

(1) Total number of words is 343 242. The program in Appendix (A1) finds this number, the maximum number of words by file (1937) and stores the words in a file *wordsTous.txt* and takes around three minutes to complete the task.

(2) The program in Appendix (A2) finds the distinct words (non repeated) and the number of occurrences respectively stored in *words.txt* and *wordsEff.txt*.

In the program in Appendix (A3), these files were simultaneously sorted by the frequencies (descreasingly) and saved in the files *wordsOr.txt* and *wordsEffOr.txt*.

From the data we had, the number of distinct words was $dimWords = 19247$.

It was not a surprise that the most common words (the, and, that, for, with, are, was, said, this, their, have, will, they, Gambia, from, his, not, has, etc.) had the greatest frequencies of occurrence. For example, the words given in the brackets came in that order in highest frequency and are the only ones with more than one thousand appearances. A number of $nbrCW = 115$ common words is stored the file *words_common.txt*. In a continuous learning process, specially in a data streaming environment, that file should be updated on a regular basis.

(3) The next step was to drop the common words (in the file *words_common.text*) and all the words of one or two digits from *wordsOr.txt* to get the significant words. We used Program in Appendix (A4). The results, the words and their frequencies, are in files *wordsOrC.txt* and *wordsEffOrC.txt*.

(4) Finally, we get $dimWords = 681$ significant words or dimension words. What a decrease from $dimWords = 19247$ to $dimWords = 681$!

At a first step, these facts allow to represent files as vectors as follows.

For any file f, let us denote by $c_h(f)$ the frequency of the $w(h)$ in the file f and c_h its frequency in all the $N = 654$ files. We may represent f by the vector

$$f = (c_1(f), c_2(f), ..., c_m(f)^T,$$

where A^T stands for the transpose of a matrix A. If we denote

$$e_i = (\delta_i j)_{1 \leq j \leq m}$$

as the m-dimensional vector whose all components are zeros except the i^{th} which is one, we may use the vectorial notation

$$f = \sum_{h=1}^{m} c_h e_h.$$

We will later see how to design a metric for these vectors. But, for sure, the dimension is so large that we will face very long processing time, meaning we face big data concerns. We have to be sure that we can use a smaller set of dimension words that allows to discriminate the type of articles. Let us go further.

Let us remark that the term frequency - inverse Document frequency index (of Google ™) of the dimension word $w(h)$ in the document f is given by

$$(TF.IDF)_{h,f} = \log_2(N/c_h) \max_{1 \leq \ell \leq m} \frac{c_h(f)}{c_\ell(f)}. \quad (3)$$

Unfortunately, due to the time limitation, we will not have the time to implement them.

5 (**Dimension Reduction**) For each of the six types, we may classify the frequency of all the dimensions words in a decreasing order. The program in Appendix (A6) implements this and saves the data in files, for example for agriculture, *EffectOr_Coordinates_agric.txt* (for the frequencies) and (*wordsOr_Coordinates_agric.txt*) for the words order as the frequencies, etc. We get for each type j, the most frequently words that appear in that type of fields, which are denoted

$$w(h, j),\ i = 1, 2, ...$$

Hence $w(1, 1)$ is the most used dimension word in agriculture file (since typ(1)="agric"), $w(2, 6)$ is the second most used word in sports file, etc. For j fixed, the frequencies associated with $w(h, j)$ are read in the file *EffectOr_Coordinates+typ(j).txt* and the dimension words, ranked accordingly, *wordsOr_Coordinates+typ(j).txt*.

Agriculture		business		health	
Food	123	business	86	health	501
rice	120	people	56	women	347
development	115	country	51	people	225
production	108	2018	50	violence	198
farmers	101	2017	48	rights	186
agricultural	84	project	46	care	167
agriculture	83	development	45	human	148
national	64	market	40	blood	141
project	60	government	37	right	139
2017	59	trade	37	treatment	138
country	58	economic	36	children	122
support	57	support	35	medical	120
women	55	between	35	risk	116
rural	54	youth	32	countries	112
people	47	young	32	girls	107
government	40	event	30	symptoms	98
management	40	price	30	men	94

Fig. 2. Top frequent words by Categories (a)

Figures 2 and 3, which display the first top words by category, suggest that the top frequent words differ by categories. So it makes sense that we may have a shorter list of dimension words which may help in discriminating the articles categories. So we are doing this:

1. Fix a number Q
2. Take Q first dimension words in each category
3. Form a set of dimension words for those choices
4. Call that set: Reduced dimensionWords of order Q

Politics		Society		Sports	
Political	260	music	526	2018	126
people	204	song	141	football	106
government	174	album	139	team	98
party	167	jazz	133	league	83
candidates	164	first	122	game	70
country	134	released	102	first	68
2018	107	songs	94	second	65
elections	107	working	90	points	58
women	102	year	84	coach	55
development	99	years	78	players	53
parties	92	people	77	national	52
election	86	international	70	played	49
peace	85	work	70	club	48
former	78	produced	69	match	47
candidate	76	artists	68	last	46
president	70	2015	67	international	44
citizens	67	band	67	goal	42
years	64	young	66	side	41
process	63	musical	66	top	41
democracy	60	artistes	64	final	32

Fig. 3. Top frequent words by Categories (b)

5. Name dimWordsReduced_Q the size of that set
6. Adopt the same notations as above

For example, by taking $Q = 1$, we surely have 6 dimensions at the place of the 681. One may think that choosing $Q = 1$ could lead to a poor approximation. But, by the miracle of Data mining, the dimension $Q = 1$ gives interesting results with misclassification rate up tp 12.5%.

Before we go any further, we mention that the program in Appendix (A5) also gives the coordinates of all files f in files named as $Coordinate + key(f).txt$ and the coordinates by categories in files $Coordinate + typ(f).txt$.

All the programs run in less than four minutes. The program that requires the most times (three minutes) is the one in Appendix (A1) which sets the list of all numbers. Although, it may take two to three days for conceiving and implementing the programs, they may be run, all of them, in 30mn. From that point, all the characteristics are stored in files. From that point, further data mining knowledge retrieving may be added and should run quickly.

Now we are ready to proceed to a supervized Learning process.

3. Supervised Learning Process: Classification

The first section was devoted to an unsupervised statistical learning process, meaning that we were searching facts, features and knowledge without having in mind any model to validate or not. In such a process, we applied automatic procedures such as counting elements, displaying graphics and scatter plots and time evolution graphs, computing empirical parameters, etc. We also followed our own intuition and guess, based on the nature of the data.

As guessed, counting the frequencies of the significant words in each category of documents suggested that:

(a) for a small number Q, the sets of Q top frequent words for each category are distinct or have a very small number of common words. Let us denote those sets by $E(Q, j)$, where j is the category;

(b) we are in a position to classify any file f from Gambian online newspapers by comparing the sum of frequencies of each element of $E(G, j)$ denoted by $S(Q, j, f)$. And finally, we adopted following rule of classification:

Classify f in the category $\hat{j}$, which is the value of j maximizing $S(Q, j, f)$:

$$\hat{j} = \underset{1 \leq j \leq nbrTyp}{\text{Argmax}} \; S(Q, j, f).$$

In modern statistical learning, $c(j) = \hat{j}$ is called a classifier.

Now, we effectively adopt that classification method. How to validate it?

We take the actual data as *training data set*. Since the data contain the outcomes we are searching, the true classification, we may apply the method and compute the misclassification errors, or rate of error, defined by

$$error = \frac{\text{Number of misclassifications of f}}{\text{Number of f}}.$$

We are then in a supervized statistical learning since we may observe the error of the model on the training data.

The method is implemented in the program in Appendix (A7) where the number Q takes the values $Q = 1$, $Q = 5$, $Q = 1$, $Q = 30$. The reported errors are given for each category in Table 2. At the intersection of the line Q and the column of the category j, we have the rate of misclassification. For example, the greatest rate concerns the *Agriculture* category for $Q = 1$, which remarkably good for a classification based on the only word of *food*. From that table, we draw the following conclusions:

(a) The overall error is very small, for $Q = 10$ and negligible from $Q = 30$.

(b) The dimension words for categories Health and Politics better characterize their topics than the others.

Conclusion. Our first conclusion is that this classifier is very accurate and we recommend it for use. We also recommend to regularly update the dimension words by category as the data are continuously collected as time goes forward.

Q	j=1	j=2	j=3	j=4	j=5	j=6
1	12.5	19.17	2.7	10.06	0.90	1.3
5	3.12	4.1	7.40	2.01	2.72	0.66
10	1.56	0.14	4.63	0	0.072	0.066
30	0	0	0.02	0	0.072	0.013

Table 2. Training Errors by Categories in %

Testing Errors for the Model.

With additional data, the method has been successfully tested with similar rates of errors. This simply supports our conclusion we already gave.

Part (B): Collaborative Data Streaming Data Platform.

Part (A) makes us very eager to extend it in three directions:

(1) Expand the study on a continenal basis (Francophone and Anglophone Africa).

(2) Use representative samples of the collected articles according to a selected and reasonable design.

(3) Adopt a Data streaming scheme with a continuous update of the keywords.

Let us describe how to achieve this, based on the secured results of Part (A).

4. Streaming Approach

(a) - The main frame.

Let us see from the Gambian context how to adapt the procedures to a data streaming way. We suggest to proceed as described below.

(1) From a time t_0 taken equal to 0, we decide to observe all online newspapers in the Gambia.

(2) A list of all outlets is available as well as their categories. The number of categories and their extents might not be the same. A task for harmonizing them,

regrouping or splitting some of them might be a necessary job to do.

(3) Articles of online newspapers are posted any time, not necessarily on a daily basis. Suppose that the time is discretized and counted in hours ($T = 1$) or in halves of an hour ($T = 1H/2$), in every two hours ($T = 2H$), etc. So the observation times are $t = rT$, where $r = 1, 2, ..., r$ and T being chosen in function on the constraints.

So, we have the number of outlets nbrNews(gam,t), the number of categories $nbrTyp(gam, t)$ and the total number of released articles $n(gam, t)$ in Gambia at the time t (*gam* is the internet domain name of Gambia).

Now, suppose that the study is extended to the whole African continent, but restricted to newspapers written in English (abbreviated as *en*. If *D* is the class of all domain names of concerned countries, the total number of articles released at time t

$$n(t, en) = \sum_{dn \in \mathcal{D}} n(dn, t, en).$$

The available data amount to 1652504 letters which take roughly 200 mb. This quantity is quite small. Though this is only a sample, we still may say that the true total number of files would take at most some gigabytes (GB) in the *RAM*. We also might expect that, on continental level, the data, when accumulated should be of the order of some terabytes. With the accumulation of the data, we are not far from the big data domain.

But, the real source of Big data approach resides in the computation of distances between files. Since, we count the streaming files each period T, we would wish to be able to complete the computations in a period T and save the reports.

In conclusion, if the data are such that the algorithm in Part (A) could not be completely run in a period T for all the files from newspapers in the area, we have to use representative data and to appeal to streaming data techniques.

(b) - Possible way to collect the data.

A working and ideal hypothesis would be that there is a database to which all online newspapers post their data by providing: the country, the name of the newspapers, the title of the paper, the category, and the text file. Even if the text files are not posted for obvious copyrights protection, the three elements are enough for our purpose.

To simplify, let us consider the same problem as in Part (A). The treatment to be done for any sample of data consists in:

(a) Finding all the words of more than three letters.

(b) Concomitantly ordering these words and their frequencies.

(c) Detecting possible common words in the top frequent words, dropping them and enriching the common words file.

(d) Cleaning the list of dimension words by using the updated common word files.

The Data streaming should work as follows.

(1) The database of online newspapers and categories is updated each period of T.

(2) Since the number of countries is not so large, we may decide to consider a sample from each country (otherwise, we consider a sample of countries and from each country a sample of journals and from each journal a sample of articles from each category).

We fix a proportion a/b of data to treat in each countries, say $a/b = 1/10$. From the data from each country at time t, we select a pseudo random sample by using the following hashing method (where j stands for the category):

If $(t * j \mod b)$ is less that a, retain the article. If not, no.

If the number $n(dn, t)$ is large, the random sample should contain a good representation of the categories and the outlets. Even there is a chance that the sample might be biased, the repetition of the sampling each period of T will coerce the system to represent all the outlets and the categories. As well, if the number of categories is not too large, we may use it as a stratum, and select the articles from the whole class of outlets.

(3) After a good sample has been treated, i.e. the above tasks (a) - (d) have been completed, the dimension words and their frequencies may be updated by taking the means over the number of periods observed. **We remind that the mean is easily handled for streaming data since only the cumulative sum is saved at each period**.

(4) Finally, to each t, the updated dimension words available at that time is the key tool used for classificatory and clustering purposes.

Part (C) - Conclusions, recommendations, Perspectives.

The Algorithm described above should give far better results than the static approach used in Part (A), mainly because of the randomization and the continuous information supply.

We recommend to consider this work as the beginning point of a continental study with two parts, one for the English speaking countries and another for the French speaking ones. The data collected will serve for the destined purpose but also for teaching materials.

Although the results are important, we will be devoted to a second study using the metric and explore the outcomes of classification methods. The packages that will be implemented will aslo be part of the platform.

References

Leskovec J., Rajaraman A. and Ullman J. D. (2014). *Mining of Massive Datasets*. Cambridge University Press. London Stanford Univ.

Saidou M.S. B. (2018). *Data Streaming in Web Mining*. Master of Sciences Dissertation. University of The Gambia. 2018

Breiman L. (2001) Statistical Modeling: The Two Cultures. *Statistical Science*, Vol. 16 (3), 199–231

Soumaila D. and Lo G.S. (2015). Probabilistic, Statistical and Algorithmic Aspects of the Similarity of Texts and Application to Gospels Comparison. *Journal of Data Analysis and Information Processing* (SciRes), Vol. 3, pp. 112-127. http://dx.doi.org/10.4236/jdaip.2015.34012

Dembele S. and Lo G.S. (2017). The exact probability law for the approximated similarity from the Minhashing method. *Afrika Statistika* . Vol. 12 (1), pp 1199-1218. Doi: http://dx.doi.org/10.16929/as/2017.1199.100

Appendix: packages.

(1) All the data can be found at : *www.jafristatap.net/archives/sanyangData.zip*.

(2) All the MS Visual basic™ which are listed below can be found in *www.jafristatap.net/archives/sanyangPackages.zip*.

Warning. People willing to run these programs in VB 6 should at least do this:

(a) Set the right working directory with the instruction

chemin = "C:/dataGG/webmining/data/"

by

chemin = "working directory/data/"

(b) Have their data in a sub-directory data and load their files there.

(c) Adapt the arrays typ(.), news(.) and nbrARt(.,.).

(d) Have valid files named following the explanation in Part (A).

(d) Create a sub-directory: Coordinates (in which the coordinates will be stored).

(A1) - Finding list or all words.

Also, finding maximum number of words (nbrK) in a file, storage of words in the file.

```
Dim i As Integer, j As Integer, rgk As String, pos As Integer, rr As Integer, word As String, chaine As String, nbr As Double
Dim lg As Integer, maxi As Integer, nbrK As Integer
chemin = "C:/dataGG/webmining/data/"
nbr = 0

typ(1) = "agric"
typ(2) = "bness"
typ(3) = "hlth"
typ(4) = "pol"
typ(5) = "soc"
typ(6) = "sports"

nbrTyp = 6

news(1) = "foroyaa"
news(2) = "point"
news(3) = "standard"

nbrNews = 3

nbrArt(1, 1) = 37
nbrArt(1, 2) = 36
nbrArt(1, 3) = 50
nbrArt(1, 4) = 50
nbrArt(1, 5) = 50
nbrArt(1, 6) = 50

nbrArt(2, 1) = 10
nbrArt(2, 2) = 11
nbrArt(2, 3) = 30
nbrArt(2, 4) = 49
nbrArt(2, 5) = 50
nbrArt(2, 6) = 50

nbrArt(3, 1) = 17
nbrArt(3, 2) = 26
nbrArt(3, 3) = 28
nbrArt(3, 4) = 50
nbrArt(3, 5) = 10
nbrArt(3, 6) = 50

nbrWords = 0
Open chemin & "words.txt" For Output As #2
maxi = 0
Rem For i = 1 To 1
Rem For j = 1 To 1
For i = 1 To nbrNews
For j = 1 To nbrTyp
    For k = 1 To nbrArt(i, j)
            nbrK = 0
            rgk = Trim(Str(k))
            If k < 10 Then
                rgk = "0" & Trim(Str(k))
            End If


            Open chemin & news(i) & "-" & typ(j) & "-" & rgk & ".txt" For Input As #1

                While Not EOF(1)
                    Input #1, word
                        word = Trim(word)
                        lg = Len(word)
                        If (lg > 2) Then
                                pos = 0
                                Rem chaine = ""
                                For rr = 1 To lg
                                        If (Mid(word, rr, 1) = " ") Then
                                            Rem chaine = chaine & " / " & Mid(word, pos + 1, rr - pos - 1)
                                            chaine = Mid(word, pos + 1, rr - pos - 1)
                                                If nbrWords = 0 Then
                                                    nbrWords = nbrWords + 1
                                                    nbrK = nbrK + 1
                                                    words(nbrWords) = chaine
                                                    Rem Print #2, chaine
                                                Else
                                                    If testWords(chaine, nbrWords, words) Then
                                                            nbrWords = nbrWords + 1
                                                            nbrK = nbrK + 1
                                                            words(nbrWords) = chaine
                                                            Rem Print #2, chaine
                                                    End If
                                                End If

                                            pos = rr
                                        End If
                                Next
                                Rem chaine = chaine & " / " & Mid(word, pos + 1, lg - pos)
                                chaine = Mid(word, pos + 1, lg - pos)
```

```
                                Rem ==
                                    If nbrWords = 0 Then
                                                nbrWords = nbrWords + 1
                                                nbrK = nbrK + 1
                                                words(nbrWords) = chaine
                                                Rem Print #2, chaine
                                            Else
                                                If testWords(chaine, nbrWords, words) Then
                                                        nbrWords = nbrWords + 1
                                                        nbrK = nbrK + 1
                                                        words(nbrWords) = chaine
                                                        Rem Print #2, chaine
                                                End If
                                    End If
                                Rem ==
                            End If
                    Wend
                Close #1
                If (nbrK > maxi) Then maxi = nbrK

    Next k
Next j
Next i
Close #2
MsgBox "ok / " & nbrWords & " / " & maxi
End Sub
```

(A2) - Finding the distinct words and their frequencies

```
Dim i As Double, j As Double, rgk As String, pos As Integer, rr As Integer, word As String, chaine As String, nbr As Double
Dim lg As Integer, testCond As Boolean, timedeb As Date, timefin As Date, tempOrder As Date

chemin = "C:/dataGG/webmining/data/"
Open chemin & "wordsTous.txt" For Input As #2

Rem reading the data
timedeb = Time
nbrWords = 0
While Not EOF(2)
Input #2, chaine

    If (Len(chaine) > 2) Then
        nbrWords = nbrWords + 1
        wordsTous(nbrWords) = LCase(Trim(chaine))
    End If

Wend
Close #2

nbrModWords = 1
words(nbrModWords) = wordsTous(nbrModWords)
wordsEff(nbrModWords) = 1

For j = 2 To nbrWords
        testCond = True
        For i = 1 To nbrModWords
            If wordsTous(j) = words(i) Then
                testCond = False
                wordsEff(i) = wordsEff(i) + 1
                Exit For
            End If
        Next

        If testCond Then
                nbrModWords = nbrModWords + 1
                words(nbrModWords) = wordsTous(j)
                wordsEff(nbrModWords) = 1

        End If
Next
timefin = Time

MsgBox "Fin : " & nbrModWords & " / " & timedeb & " " & timefin

Open chemin & "words.txt" For Output As #3
Open chemin & "wordsEffectifs.txt" For Output As #4
For i = 1 To nbrModWords
Print #3, words(i)
Print #4, wordsEff(i)
Next
Close #3, #4

MsgBox "Fin : Files created / Ordonne"
```

(A3) - Creating files containing the ordered dimension words and frequencies

```
Dim i As Double, j As Double, rgk As String, pos As Integer, rr As Integer, word As String, chaine As String, nbr As Double
Dim lg As Integer, testCond As Boolean, timedeb As Date, timefin As Date, tempOrder As Date, prov As Integer, prove As String


chemin = "C:/dataGG/webmining/data/"

Open chemin & "words.txt" For Input As #1
Open chemin & "wordsEffectifs.txt" For Input As #2

Rem MsgBox chemin & "testEff.txt"

Rem Open chemin & "test1.txt" For Input As #1
Rem Open chemin & "testEff.txt" For Input As #2


i = 0
While Not EOF(1)
    i = i + 1
    Input #1, chaine
    words(i) = chaine
    Input #2, chaine
    wordsEff(i) = Val(chaine)
Wend
Close #2, #1

nbrModWords = i

MsgBox "ok / Continue : pour ordonner "
timedeb = Time

For i = 1 To nbrModWords
    For j = i + 1 To nbrModWords

        If wordsEff(j) > wordsEff(i) Then

            prov = wordsEff(j)
            wordsEff(j) = wordsEff(i)
            wordsEff(i) = prov

            prove = words(j)
            words(j) = words(i)
            words(i) = prove

        End If
    Next
Next

Rem chaine = ""
Rem For i = 1 To nbrModWords
Rem     chaine = chaine & wordsEff(i) & vbLf
Rem Next
Rem MsgBox chaine

Rem Exit Sub


Rem timefin = time


Rem MsgBox "Fin tri: " & nbrModWords & " / " & timedeb & " " & timefin

Open chemin & "wordsOr.txt" For Output As #3
Open chemin & "wordsOrEffectifs.txt" For Output As #4
For i = 1 To nbrModWords
Print #3, words(i)
Print #4, wordsEff(i)
Next
Close #3, #4

MsgBox "Fin : Files created / Ordonne"
```

(A4) - Dropping the short words and the common words

```
Dim i As Double, j As Double, rgk As String, pos As Integer, rr As Integer, word As String, chaine As String, nbr As Double
Dim lg As Integer, testCond As Boolean, timedeb As Date, timefin As Date, tempOrder As Date, prov As Integer, prove As String
Dim testWords(1 To 2000) As String, dimWords As Double, chaine2 As String, testCondFreq As Boolean

chemin = "C:/dataGG/webmining/data/"

Open chemin & "words_common.txt" For Input As #1
i = 0
    While Not EOF(1)
        Input #1, chaine
        i = i + 1
        testWords(i) = Trim(chaine)
    Wend
Close #1
dimWords = i

Open chemin & "wordsOr.txt" For Input As #1
Open chemin & "wordsOrEffectifs.txt" For Input As #2

Open chemin & "wordsOrC.txt" For Output As #3
Open chemin & "wordsOrCEffectifs.txt" For Output As #4

testCondFreq = True
j = 1
While (Not EOF(1) And testCondFreq)
    Input #1, chaine
    Input #2, chaine2

    If Val(chaine2) > 49 Then
        testCond = True
            For i = 1 To dimWords
                If (Trim(chaine) = testWords(i)) Then
                    testCond = False
                    Exit For
                End If
            Next

            If testCond Then
                Print #3, chaine
                Print #4, Val(chaine2)
                j = j + 1
            End If
    Else
            testCondFreq = False
    End If

Wend


Close #1, #2, #3, #4

MsgBox "New files created / " & j
```

(A5) - Finding the coordinates of the files and the categories in the dimension words

```
Dim h As Integer, i As Integer, j As Integer, rgk As String, pos As Integer, rr As Integer, word As String, chaine As String, nbr As Double
Dim lg As Integer, timedeb As Date, timefin As Date, thiabi As Integer, wordsEffectif(1 To 700) As Integer, wordsEffectifType(1 To 700) As Integer
Dim dimWords As Double, chaine2 As String, testCondFreq As Boolean
Dim wordsFile(1 To 20000) As String

chemin = "C:/dataGG/webmining/data/"

Open chemin & "wordsOrC.txt" For Input As #1
 i = 0
    While Not EOF(1)
        Input #1, chaine
         i = i + 1
        words(i) = Trim(chaine)
     Wend
Close #1
dimWords = i
nbr = 0
Rem MsgBox "Already done"
Rem Exit Sub

typ(1) = "agric"
typ(2) = "bness"
typ(3) = "hlth"
typ(4) = "pol"
typ(5) = "soc"
typ(6) = "sports"

nbrTyp = 6

news(1) = "foroyaa"
news(2) = "point"
news(3) = "standard"

nbrNews = 3

nbrArt(1, 1) = 37
nbrArt(1, 2) = 36
nbrArt(1, 3) = 50
nbrArt(1, 4) = 50
nbrArt(1, 5) = 50
nbrArt(1, 6) = 50

nbrArt(2, 1) = 10
nbrArt(2, 2) = 11
nbrArt(2, 3) = 30
nbrArt(2, 4) = 49
nbrArt(2, 5) = 50
nbrArt(2, 6) = 50

nbrArt(3, 1) = 17
nbrArt(3, 2) = 26
nbrArt(3, 3) = 28
nbrArt(3, 4) = 50
nbrArt(3, 5) = 10
nbrArt(3, 6) = 50

nbrWords = 0
timedeb = Time


For j = 1 To nbrTyp
Rem j = 1
        For rr = 1 To dimWords
            wordsEffectifType(rr) = 0
        Next

        For i = 1 To nbrNews
        Rem i = 3

                    For k = 1 To nbrArt(i, j)
                    Rem k = 1

                        rgk = Trim(Str(k))
                        If k < 10 Then
                            rgk = "0" & Trim(Str(k))
                        End If

                        thiabi = 1000 * j + 100 * i + k

                        For rr = 1 To dimWords
                            wordsEffectif(rr) = 0
                        Next

                        Open chemin & news(i) & "-" & typ(j) & "-" & rgk & ".txt" For Input As #2
                            nbrWords = 0
                            While Not EOF(2)
                            Input #2, word
                            word = Trim(word)
                            lg = Len(word)
                            pos = 0
```

```
                        For rr = 1 To lg
                                    If (Mid(word, rr, 1) = " ") Then
                                        chaine = Trim(Mid(word, pos + 1, rr - pos - 1))
                                        pos = rr
                                        If (Len(chaine) > 2) Then
                                            nbrWords = nbrWords + 1
                                            wordsFile(nbrWords) = Trim(chaine)
                                        End If
                                    End If
                        Next
                        chaine = Trim(Mid(word, pos + 1, lg - pos))

                        If (Len(chaine) > 2) Then
                            nbrWords = nbrWords + 1
                            wordsFile(nbrWords) = Trim(chaine)
                        End If
                    Wend
                    Close #2


                        For rr = 1 To nbrWords
                            For h = 1 To dimWords
                                If wordsFile(rr) = words(h) Then
                                    wordsEffectif(h) = wordsEffectif(h) + 1
                                    Rem Exit For
                                End If
                            Next
                        Next


                    Open chemin & "coordinates/Coordinates" & Trim(Str(thiabi)) & ".txt" For Output As #2
                        For h = 1 To dimWords
                            Print #2, wordsEffectif(h)
                        Next
                    Close #2

                    Rem MsgBox "Wait " & "/ " & nbrWords

                    For h = 1 To dimWords
                        wordsEffectifType(h) = wordsEffectifType(h) + wordsEffectif(h)
                    Next
                Next k
            Next i


                    Open chemin & "coordinates/Coordinates" & typ(j) & ".txt" For Output As #2
                        For h = 1 To dimWords
                            Print #2, wordsEffectifType(h)
                        Next
                    Close #2
Next j

MsgBox "ok / " & " / " & timedeb & " / " & Time & " / "
```

(A6) - Creating the files of most frequent words by categories

```
Dim h As Double, i As Double, j As Double, rgk As String, pos As Integer, rr As Integer, word As String, chaine As String, nbr As Double
Dim lg As Integer, testCond As Boolean, timedeb As Date, timefin As Date, tempOrder As Date, prov As Integer, prove As String

Rem MsgBox "Already done"
Rem Exit Sub

chemin = "C:/dataGG/webmining/data/"
typ(1) = "agric"
typ(2) = "bness"
typ(3) = "hlth"
typ(4) = "pol"
typ(5) = "soc"
typ(6) = "sports"
nbrTyp = 6

h = 6

Open chemin & "wordsOrC.txt" For Input As #1
Open chemin & "coordinates/Coordinates" & typ(h) & ".txt" For Input As #2



i = 0
While Not EOF(1)
    i = i + 1
    Input #1, chaine
    words(i) = chaine
    Input #2, chaine
    wordsEff(i) = Val(chaine)
Wend
Close #2, #1

nbrModWords = i

For i = 1 To nbrModWords
    For j = i + 1 To nbrModWords

        If wordsEff(j) > wordsEff(i) Then

            prov = wordsEff(j)
            wordsEff(j) = wordsEff(i)
            wordsEff(i) = prov

            prove = words(j)
            words(j) = words(i)
            words(i) = prove

        End If
    Next
Next


Open chemin & "coordinates/wordsOr_Coordinates_" & typ(h) & ".txt" For Output As #3
Open chemin & "coordinates/EffectOr_Coordinates_" & typ(h) & ".txt" For Output As #4
For i = 1 To nbrModWords
Print #3, words(i)
Print #4, wordsEff(i)
Next
Close #3, #4

MsgBox "Fin : Files created / Ordonne"
```