Text Mining Scientific Articles using the R Language

Carlos A.S.J. Gulo[1,2] and Thiago R.P.M. Rúbio[1,3]

[1] Faculdade de Engenharia da Universidade do Porto (FEUP)
Departamento de Engenharia Informática (DEI)
[2] LAETA-Laboratório Associado de Energia, Transporte e Aeronáutica (FEUP)
PIXEL Research Group - UNEMAT/Brazil (http://goo.gl/tcg6S7)
Web: http://lattes.cnpq.br/0062065110639984 - sander@unemat.br
[3] LIACC Research Group (FEUP)
reis.thiago@fe.up.pt

Abstract. The aim of this study is to develop a solution for text mining scientific articles using the R language in the "Knowledge Extraction and Machine Learning" course. Automatic text summary of papers is a challenging problem whose approach would allow researchers to browse large article collections and quickly view highlights and drill down for details. The proposed solution is based in social network analysis, topic models and bipartite graph approaches. Our method defines a bipartite graph between documents and topics built using the Latent Dirichlet Allocation topic model. The topics are connected to generate a network of topics, which is converted to bipartite graph, using topics collected in the same document. Hence, it is revealed to be a very promising technique for providing insights about summarizing scientific article collections.

Keywords: Text Mining, Topic Model, Topic Network, Systematic Literature Review

1 Introduction

With the overwhelming amount of textual information presented in scientific literature, there is a need for effective automated processing that can help scientists to locate, gather and make use of knowledge encoded in literature that is available electronically. Although a great deal of crucial scientific information is stored in databases, the most relevant and useful information is still represented in domain literature.

The literature review process consists of: to locate, appraise, and synthesize the best-available empirical evidence to answer specific research questions. An ideal literature search would retrieve all relevant papers for inclusion and exclude all irrelevant papers. However, previous research have demonstrated a number of studies that are not fully indexed, as well as a number that are indexed incorrectly [7,15,8].

The purpose of this paper is to highlight text mining techniques as a support to identify the relevant literature from a CSV (Comma Separated Value)

collection searched in different journal repositories. Data set will be analyzed quantitatively in order to obtain a systematic review literature process involving the research domain: *high performance computing as support to computer aid diagnostic systems*. This research domain is the first author's scientific field of interest.

Text Mining is a common process of extracting relevant information using a set of documents. Text Mining provides basic preprocessing methods, such as identification, extraction of representative characteristics, and advanced operations as identifying complex patterns [11,1,5]. Document classification is a task that consists of assigning a text to one or more categories: the name of its class of subject, and main topics. This paper only addresses the summarization of *Abstracts*. Researchers are interested in the number of times certain keywords associated with specific content appear in each document.

This paper is organized as follows: in the next section is described a summary about text classification. The experiments performed using the R code and the results obtained with the sets of scientific articles considered in the automatic text summary and text classification, are discussed in Section 3, which is followed by the concluding remarks in Section 4.

2 Related Work

This section summarizes some achievements on text classification from various pieces of the literature. In general, text classification is a problem divided into nine steps. Those steps include data collection, text processing, data division, feature extraction, feature selection, data representation, classifier training, applying a classification model, and performance evaluation [12,18].

- Data Collection: In text classification, the first step is collecting data. The sample data are texts that belong to a limited scientific domain, i.e., "high performance computing as support to medical image processing" [17]. Each sample text must be labeled with one or more tags indicating a *label* to a certain class.
- Text preprocessing: Actually is preprocessing a trial to improve text classification by removing worthless information. It may include removal of punctuation, stop words (any prepositions and pronouns), and numbers [18,9]. In the context of this paper, we consider root extraction and word stemming as part of the feature extraction step [12], which will be discussed in the Feature extraction item.
- Data division: Next step divides the data into two parts, training data and testing data. Based on training data, the classification algorithm will be trained to produce a classification model. The testing data will be used to validate the performance of the resulting classification model. There is no ideal ratio of training data to testing data. The text classification experiments presented have been used 25% for training and 75% for testing. The classification performance is the average performance of implemented classification models[19,12].

– Feature extraction: Texts are characterized by features that: *a)* are not related to the content of the text, such as author gender, author name, and others; and *b)* reflect the text content, such as lexical items and grammatical categories. Considering single words, the simplest of lexical features, as a representation feature in text classification has proven effective for a number of applications. The result of this step is a list of features and their corresponding frequency in the training data set [18].

– Feature selection: The result of the feature extraction step is a long collection of features, however, not all of these features are good for classification for many reasons: first, some classification algorithms are negatively affected when using many features due to what is called "curse of dimensionality" next, the over-fitting problem may occur when the classification algorithm is trained in all features and finally some other features are common in most of the classes. To solve these problems, many methods were proposed to select the most representative features for each class in the training data set. In this paper, the most frequently used methods have been Chi Squared (CHI), term frequency (TF), document frequency (DF) and their variations. Other than statistical ranking, features with higher frequency were used. Word stems are also used as feature selections where words with the same stem are considered as one feature [19,18,17].

– Data representation: The results obtained from the previous step are represented in matrix format, and will be used by the classification algorithm. Usually, the data are in matrix format with n rows and m columns wherein the columns correspond to the selected feature, and the rows correspond to the texts in the training data. Weighting methods, such as term frequency inverse document frequency (TFIDF) and term frequency (TF) are used to compute the value of each cell in this matrix, which represents the weight of the feature in the text [9].

– Classifier training: The classification algorithm is trained using the training matrix that contains the selected features and their corresponding weights in each text of the training data. Support Vector Machine (SVM) and Naïve Bayes (NB) are the classical machine learning algorithms that have been the most used in text classification [10,1]. The result is a classification model to be tested by means of the testing data. The same weighting methods and the same features extracted from the training data will be used to test the classification model [16,19].

– Classification model evaluation: Evaluation techniques are assessed to estimate future performance by measures such as accuracy, recall, precision, and f-measure, and to maximize empirical results [17].

Table 1. Total of articles searched in journal repositories.

Repositories	Publication	
	Searched Queries	Papers
ACM Portal	("medical image") and ("high performance computing" or "parallel computing" or "parallel programming")	1
Engineering Village	(((((("medical imag*") WN KY) AND (("high performance comput*") WN KY)) OR (("parallel comput*") WN KY)) OR (("parallel programm*") WN KY)), Journal article only, English only	19
IEEE Xplore	((medical imag*) AND (("high performance comput*" OR "parallel programm*") OR "parallel comput*"))	69
ScienceDirect	"medical image" AND ("high performance computing" OR "parallel computing" OR "parallel programming")[Journals(Computer Science,Engineering)]	390
Web of Science	TOPIC: ("medical imag*") AND TOPIC: ("high performance comput*") AND TOPIC: ("parallel")	27
Total		506

3 Experiments and Discussion

This section describes the infrastructure used to perform the experiments and also illustrates and discusses the results obtained. Data set[4] used in experiments were collected from repositories showed in Table 1, and composed by 7 variables (*id*, *Title*, *Journal*, *Year*, *Abstract*, *Keywords* and *Recommend*) and 494 observations (after removing duplicated records).

We are interested in what the characteristics are *Abstract* that tend to group the article in a specific topic, and in future work recommend the prioritized observations based on high scores of topics. The analyzed variable is text data, the *Abstract*, and its unstructured data. Unstructured data has variable length, one observation contains an academic text, it has variable spelling using singular and plural forms of words, punctuation and other non alphanumeric characters, and the contents are not predefined to adhere to a set of values - it can be on a variety of topics [6,3].

To create useful data, unstructured text data should be converted into structured data for further processing. The preprocessing step, described in Section 3.1, involves extraction of words from the data and removal of punctuation and spaces, eliminates articles and other words that we are not interested in, replaces synonyms, plural and other variants of words with a single term and finally, makes the structured data, which is a table where each word becomes a variable with a numeric value for each record.

[4] Project code and data set is available in https://github.com/carlosalexsander/ECAC_Project

The test infrastructure used was composed of the RStudio development suite, available to download through the RStudio website [5]. A graphic card GeForce GT 540 (NVIDIA) with 192 CUDA cores and 2GB of GDRAM was used with a portable computer equipped with an Intel(R) Core(TM) i7-2630QM 2.0 GHz, 8GB of RAM (DDR3 1333 MHz), Linux Debian Wheezy (64 bits) operating system.

3.1 Results and Discussion

In the first step, it was necessary to install and load the R package Text Mining *tm* to process text documents. Once we have a corpus, the next step was to modify the documents in it, e.g., making everything lowercase, reducing words to their stem, removing numbers, removing punctuation, and removing common English stop-words. All this functionality is named a transformation concept, illustrated in Fig. 1. In general, all transformation work is done in all elements of the corpus applying the *tm_map* function.

```
toSpace <- content_transformer(function(x,pattern) gsub(pattern, " ", x))
corpus.m <- tm_map(paper.corpus, toSpace, "/|@|\\|")
corpus.m <- tm_map(corpus.m, content_transformer(tolower))
corpus.m <- tm_map(corpus.m, removePunctuation)
removeUnicode <- function(x) stri_replace_all_regex(x,"[^\x20-\x7E]","")
corpus.m <- tm_map(corpus.m, content_transformer(removeUnicode))
corpus.m <- tm_map(corpus.m, removeNumbers)
corpus.m <- tm_map(corpus.m, removeWords, stopwords("english"))
corpus.m <- tm_map(corpus.m, removeWords, c("using", "used", "propose", "can
    ", "also"))
library(SnowballC)
corpus.m <- tm_map(corpus.m, stemDocument, language = "english")
corpus.m <- tm_map(corpus.m, stripWhitespace)
corpus.dtm <- DocumentTermMatrix(corpus.m, control = list(minWordLength = 3,
      weighting = function(x) weightTFIDF(x, normalize = FALSE)))
```

Fig. 1. Preprocessing text documents applying function *tm_map*().

For many methods of text analysis, specifically the so called "bag-of-word" approaches, we created a common data structure for the corpus (Document Term Matrix - DTM) [13,17]. This is a matrix in which the rows represent documents and columns represent terms. The values represent how often each word occurred in each document. Not all terms are equally informative of the underlying semantic structures of texts, and some terms are rather useless for this purpose. We used the *term.statistics* function, created in order to produce text statistics, for instance, the most common words in the text, illustrated in Fig. 2.

For interpretation and computational purposes it is worthwhile to delete some less useful words from the DTM before fitting a model [14]. We can now filter out

[5] http://www.rstudio.com/products/rstudio/download/

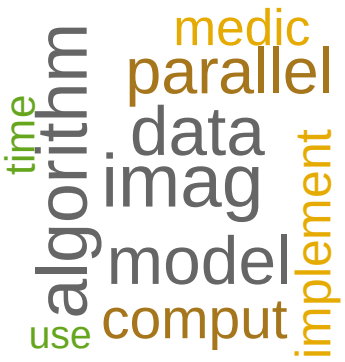

Fig. 2. Most frequented words in corpus represented by wordcloud.

words based on this information, select only the terms with the highest TFIDF score, and after apply the function *removeSparseTerms(DTM, S)* to retain the fewer (but more common) terms. The *sparse* argument value used was 0.75 to retain more words for classification than with a smaller *sparse* value, and those words were used in dictionary-based approach for term identification. When the removal of sparse terms function are applied, the number of terms is reduced from 4851 to 22.

TFIDF, is a numerical statistic which reflects how important a word is to a document in a collection or corpus. It is often used as a weighting factor in text mining. The TFIDF value increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus, which helps control the fact that some words are generally more common than others. Variations of the TFIDF weighting scheme are often used by search engines as a central tool in scoring and ranking a document's relevance given a user query[14]. TFIDF can be successfully used for stop-words filtering in various subject fields including text summary and classification. One of the simplest ranking functions is computed by summing the TFIDF for each query term; many more sophisticated ranking functions are variants of this simple model[10]. The TFIDF formula is:

$$TFIDF(i) = \frac{Frequency(i) * N}{df(i)}, \tag{1}$$

where df is the frequency of word (i) in all documents, and N is the number of words in the record/document. An interesting technique to use on a document-term matrix is the Latent Dirichlet Allocation (LDA) topic modeling. Topic modeling are statistical methods, essentially used to analyze the words of the original documents to discover the topics that run through them and how those

topics are connected to each other [6,2]. Before fitting the topic model, coded in Fig. 3, it is best to reduce the vocabulary by selecting only informative words.

```
best.model <- lapply(seq(2, 10, by = 1), function(d){LDA(term.tfidf.df, d)})
best.model.logLik <- as.data.frame(as.matrix(lapply(best.model, logLik)))
best.model.logLik.df <- data.frame(topics=c(2:10), LL = as.numeric(as.matrix
    (best.model.logLik)))
best.model.logLik.df.sort <- best.model.logLik.df[order(-best.model.logLik.
    df$LL), ]
best.model.logLik.df.sort
ntop <- best.model.logLik.df.sort[1,]$topics
lda <- LDA(term.tfidf.df, ntop)
```

Fig. 3. Code to produce a list of likelihood for each model and to optimize the LDA model.

Weights were used to determine the most efficient way. The treatment is mathematically straightforward. [16,4,3], The best model fitted is depicted by the distribution of this likelihood by topic in Fig. 4. The number of topics with the highest log likelihood is −308886.3 and 10. These topics gave the best fit for the present data.

A different plot to see the final result is the topic network. It's created by multiplying the document-topic matrix transpose with itself by an adjacency matrix. Fig. 5 shows the topic network, with topic nodes labeled by the three most probable terms in the corresponding topic distribution. In the context of this study, we define *topic* to be a distribution over a fixed vocabulary, i.e. *image* topic has words about image with high probability and the *image processing* topic has words about image processing with high probability.

4 Conclusions

In this work we studied how to use R language to text mining, and the great importance of a good preprocessing of the data in order to obtain good quality results in the text classification step. In general, building a large amount of labeled training data for text classification is a labor-intensive and time-consuming task. That was the main problem during the classifier development, and it is defined as future work. We found that the simplest document representation was at least as good as representations involving more complicated syntactic and morphological analysis. Topic networks may be useful in analyzing documents collections.

5 Acknowledgments

The first author thanks Universidade do Estado de Mato Grosso (UNEMAT - Brazil), for the support given. The work of Thiago, has been funded through a

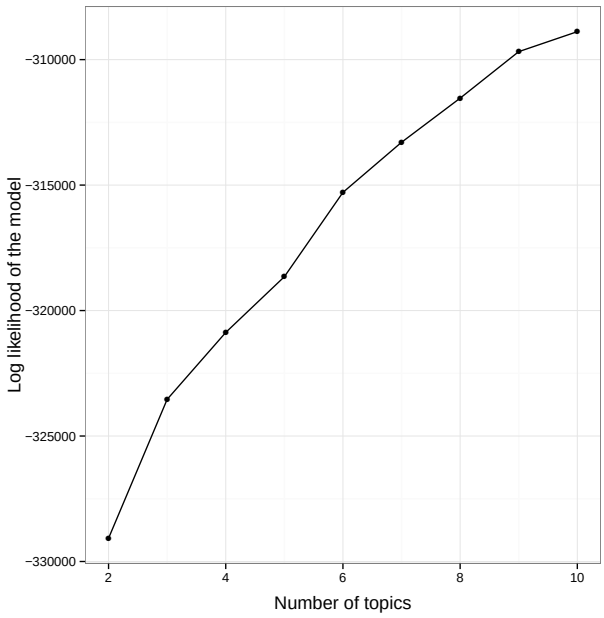

Fig. 4. The distribution of likelihood by topic.

IBRASIL Grant. IBRASIL is a Full Doctorate programme selected under Erasmus Mundus, Action 2 STRAND 1, Lot 16 and coordinated by University of Lille.

References

1. Aphinyanaphongs Y FAU Aphinyanaphongs, Y., Aliferis C FAU Aliferis, C.: Text categorization models for retrieval of high quality articles in internal medicine. In: AMIA Annual Symposium Proceedings. pp. 31–35. No. 1942-597X (Electronic) (2003), http://www.ncbi.nlm.nih.gov/pmc/articles/PMC1480096/
2. Blei, D.M.: Surveying a suite of algorithms that offer a solution to managing large document archives. Communication of the ACM 55(4), 77–84 (April 2012)
3. Blei, D.M., Lafferty, J.D.: Topic models (2009)
4. Blei, D., Ng., A., Jordan, M.: Latent dirichlet allocation. The Journal of Machine Learning Research 3, 993–1022 (2003)
5. Casualty Actuarial Society E-Forum: Text Mining Handbook (March 2010), http://www.casact.org/pubs/forum/10spforum/Francis_Flynn.pdf
6. Cohen AM FAU Cohen, A.M., Ambert K FAU Ambert, K., McDonagh M FAU McDonagh, M.: Cross-topic learning for work prioritization in systematic review creation and update. In: Journal of the American Medical Informatics Association?:

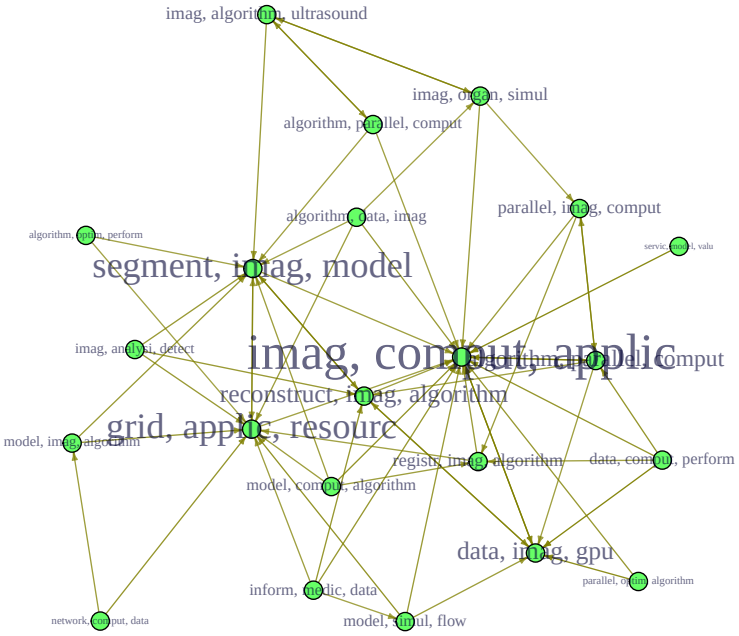

Fig. 5. Final result plotted in a Topic Network.

JAMIA. pp. 690–704. No. 1527-974X (Electronic) (2009), http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2744720/

7. Cooper, H.M.: The structure of knowledge synthesis, Knowledge in Society, vol. 1 (1988)
8. Edinger T FAU Edinger, T., Cohen AM FAU Cohen, A.M.: A large-scale analysis of the reasons given for excluding articles that are retrieved by literature search during systematic review. In: AMIA Annual Symposium Proceedings. pp. 379–387. No. 1942-597X (Electronic) (Nov 2013), http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3900186/
9. Feinerer, I., Hornik, K., Meyer, D.: Text mining infrastructure in R. Journal of Statistical Software 25(5), 1–54 (3 2008), http://www.jstatsoft.org/v25/i05
10. Hotho, A., Nürnberger, A., Paab, G.: A brief survey of text mining. LDV Forum - GLDV Journal for Computational Linguistics and Language Technology (2005)

11. Kao, A., Poteet, S.R. (eds.): Natural Language Processing and Text Mining. No. ISBN 1-84628-175-X, Springer (2007)
12. Khorsheed, M., Al-Thubaity, A.: Comparative evaluation of text classification techniques using a large diverse arabic dataset. Language Resources and Evaluation 47(2), 513–538 (2013), http://dx.doi.org/10.1007/s10579-013-9221-8
13. Lebanon, G., Mao, Y., Dillon, J.: The locally weighted bag of words framework for document representation. J. Mach. Learn. Res. 8, 2405–2441 (Dec 2007), http://dl.acm.org/citation.cfm?id=1314498.1314576
14. Munzert, S., Rubba, C., Meibner, P., Nyhuis, D.: Automated Data Collection with R: A Practical Guide to Web Scraping and Text Mining. No. ISBN 978-1-118-83481-7, John Wiley & Sons, Ltd, 1st edn. (2015)
15. Okoli, C., Schabram, K.: A guide to conducting a systematic literature review of information systems research. Sprouts: Working Papers on Information Systems 10(26) (2010), http://papers.ssrn.com/sol3/papers.cfm?abstract_id=1954824
16. Reed, C.: Latent Dirichlet Allocation: A Student Companion (2012)
17. Weiss, S.M., Indurkhya, N., Zhang, T.: Fundamentals of Predictive Text Mining. No. e-ISBN 978-1-84996-226-1, Springer (2010)
18. Weiss, S.M., Indurkhya, N., Zhang, T., Damerau, F.J.: Text Mining: Predictive Methods for Analyzing Unstructured Information. No. ISBN 0-387-95433-3, Springer (2005)
19. Zhao, Y.: R and Data Mining: Examples and Case Studies. Academic Press (2013), http://www.sciencedirect.com/science/article/pii/B9780123969637000015