Marcos-Pablos, S., & García-Peñalvo, F. J. (2019). Information retrieval methodology for aiding scientific database search. Soft Computing, doi:10.1007/s00500-018-3568-0

# Information retrieval methodology for aiding scientific database search

Samuel Marcos-Pablos · Francisco José García-Peñalvo

**Abstract** During literature reviews, and specially when conducting systematic literature reviews (SLRs), finding and screening relevant papers during scientific document search may involve managing and processing large amounts of unstructured text data. In those cases where the search topic is difficult to establish or has fuzzy limits, researchers require to broaden the scope of the search and, in consequence, data from retrieved scientific publications may become huge and uncorrelated. However, through a convenient analysis of these data the researcher may be able to discover new knowledge which may be hidden within the search output, thus exploring the limits of the search and enhancing the review scope. With that aim, this paper presents an iterative methodology that applies text mining and machine learning techniques to a downloaded corpus of abstracts from scientific databases, combining automatic processing algorithms with tools for supervised decision making in an iterative process sustained on the researchers' judgement, so as to adapt, screen and tune the search output. The paper ends showing a working example that employs a set of developed scripts that implement the different stages of the proposed methodology



**Keywords** Information retrieval · Systematic literature review · Text mining · Vector Space Model · Support Vector Machine

Samuel Marcos-Pablos
GRIAL Research Group, Research Institute for Educational Sciences, University of Salamanca, 37008 Salamanca, Spain
Tel.: (+34) 923294500 ext. 3433
E-mail: samuelmp@usal.es

Francisco J. García-Peñalvo
GRIAL Research Group, Research Institute for Educational Sciences, University of Salamanca, 37008 Salamanca, Spain

## 1 Introduction

Literature review is a key step in performing any kind of research. It allows the evaluation of the available literature, providing the state of the art on a specific topic and giving an overview of what the strengths of the area of interest are, and which weaknesses are in need of further improvement. As such, researchers look for the best means to access and exploit the knowledge contained in scientific publications.

However, through the continuous growth of on-line available documents, the task of organising the retrieved data and focus the research within a specific topic becomes one of the principal issues. One of the key underlying problems is that the terminology in the different research fields is not well established. In areas like computer science, for instance, terminology from different research fields is integrated into research that focus on solving computer science related problems. As a consequence, terms may vary between studies depending on the field the research is based on. On the other hand, computer science papers may not be published just in computer science related venues, but also in others where computer science solutions are applied (Ros et al., 2017). As stated in (Kitchenham and Charters, 2007) the amount of literature on a certain topic may be small in the field, hence it might be of use to search for literature in connected fields.

Still, new problems may arise when broadening the scope of the search in scientific databases, as the amount of retrieved heterogeneous documents unrelated to the search topic can increase dramatically and at some point become unmanageable. Reached that point, retrieved data meets two of the three characteristics that have emerged as a common framework to describe big data, also known as the three V's: high volume, high variety and / or high velocity (Gandomi and Haider, 2015). As stated in (LHeureux et al., 2017), according to this definition big data are not just typified by their specific size metrics, but rather by the fact that new approaches are needed to process them because of their size, heterogeneity or velocity.

Nevertheless, volume and variety of retrieved documents in a broaden scientific document search may not just be seen as a hindrance, but as means of exploiting the potential of big data, that is, the use of large and wide-ranging amounts of data to enable knowledge discovery and better decision making (Mayer-Schnberger and Cukier, 2013). Working with big data involves various approaches, technologies, and tools such as those from text analytics, business intelligence, data visualization, and statistical analysis (LHeureux et al., 2017). However, in many cases improving traditional methodologies or creating them from scratch is needed in order to get the most out of such data (Hordri et al., 2017), (Eachempati and Srivastava, 2017).

Taking the above into account, in this paper we present an approach for aiding in the tedious process of relevant document search in literature research databases, based on an iterative methodology that applies text mining and machine learning techniques to a corpus of abstracts obtained from regular searches in research databases. We particularly (but not only) focus in those cases where the search topic is difficult to establish in advance or has fuzzy limits, which in turn forces the researcher to broaden the search scope and thus produces large amounts of uncorrelated results (high variety and / or high volume). As literature reviews, and more specifically systematic literature reviews (SLRs), have reached a consid-

erable level of adoption in many research fields, the proposed methodology can be very useful in order to assist the task of SLR conduction, specifically during the preparation stage which involves searching for relevant documents and screening the results in order to gather relevant scientific documents in the province of the research.

The rest of the paper is structured as follows. Section 2 presents some background on SLRs and the text mining methods and approaches employed through the rest of the sections. Then, section 3 describes the proposed methodology, depicting the proposed algorithm and its stages. Section 4 shows a working example of the proposed methodology. Finally, conclusions are given in section 5.

## 2 Background

Whether the data are big or small, extracting value from data requires multiple steps that conform what is known as the data analysis pipeline. As described in (Labrinidis and Jagadish, 2012) the first step of the pipeline is data acquisition, where data must be pre-processed, filtered and prepared (e.g. removing duplicates, entries with wrong format) in such a way that useful information is not discarded. Next, as collected data are usually in a format that is not ready for analysis, it is necessary transform their structure and semantics and store them in a way that is suitable for analysis, that is, in forms that are ready to be processed by computer based algorithms. Then data must be analysed taking into account that the analysis techniques and algorithms applied will depend on the previously selected data representation format, and certain designs will be more suitable for certain purposes than others. Finally, analysed data is of no use if users cannot understand and exploit its results, so tools for decision-making are needed in order to interpret and get the most of them, which may also involve finding possible sources of error such as model bugs or bad assumptions.

When working with scientific document searches in SLRs, following this data analysis pipeline can ease the process of managing large datasets, allowing the researches to pay more effort in the tasks that require expert interpretation and judgement, and less on those that require repetitive well-defined steps to be fulfilled (Olorisade et al., 2016), (O'Mara-Eves et al., 2015),(Marshall and Brereton, 2013), (Felizardo et al., 2010). The next subsections provide an overview of the necessary steps needed to conduct a systematic literature review, with special focus in the document retrieval stage, along with the required background in text mining techniques and processes needed for preparing and representing the retrieved data obtained from document searches in scientific databases. This background will in turn form the basis of the data representation format and analysis employed in our proposed methodology for aiding scientific database search.



### 2.1 Systematic Literature Review

As stated before, SLRs are a systematic approach for the search and evaluation of the available literature in a given topic, providing a strong foundation of the state of the art and giving an overview of the current strengths and weaknesses of the field of interest. For that reason, and even though conducting a systematic

**Table 1** Different steps, tasks and stages of a SLR. Adapted from (Tsafnat et al., 2014)

| Task | Description |
|---|---|
| 1. Formulate review question | Decide on the research question of the review |
| 2. Find previous systematic reviews | Search for systematic reviews that answer the same question |
| 3. Write the protocol | Provide an objective, reproducible, methodology for peer review |
| 4. Devise search strategy | Decide on databases and keywords to find all relevant trials |
| 5. Search | Aim to find all relevant citations |
| 6. De-duplicate | Remove identical citations |
| 7. Screen abstracts | Based on titles and abstracts, remove definitely irrelevant trials |
| 8. Obtain full text | Download or request copies from authors |
| 9. Screen full text | Exclude irrelevant trials |
| 10. Snowball | Follow citations from included trials to find additional ones |
| 11. Extract data | Extract relevant information to help with the synthesis and conclusions |
| 12. Synthesize data | Convert extracted data to a common representation |
| 13. Re-check literature | Repeat search to find new literature published since the initial search |
| 14. Meta analyse | Statistically combine the result from all included trials |
| 15. Write up review | Produce and publish final report |

review requires a great amount of work and dedication (Petticrew and Roberts, 2008), SLRs have reached a considerable level of adoption in many research fields (Eachempati and Srivastava, 2017), (Hordri et al., 2017),(Franco-Bedoya et al., 2017), (Al-Ruithe et al., 2018),(Nelson and Olovsson, 2016).

Table 1 shows the different stages needed to perform a SLR (Tsafnat et al., 2014). In the initial stages, which involve the definition of the review protocol and the search strategy, the tasks mainly rely on the researcher's knowledge, expertise and judgement, whereas once the review protocol has been established more technical and repetitive tasks are performed during the following stages.

Due to the complexity of conducting SLRs, there have been different proposals for the automation of different stages of the SLR process (Tsafnat et al., 2014). Most of the automation aiding proposals have focused on the technical and repetitive tasks in the retrieval, appraisal and synthesis stages (Table 1). However, less interest has been given to the preparation stage, for example in the task of building a convenient string that allows to retrieve as many results as possible related to the topic of interest. Even though studies show that many researches face difficulties when constructing the search string (Mergel et al., 2015), the retrieval stage has mainly been left to the knowledge and judgement of the researchers conducting the review. Next sections describe the background related to text mining and the proposed methodology for aiding in these stages of the SLR process.



## 2.2 Vector Space Model

The information contained in unstructured texts or documents cannot simply be used for further processing by computers, which typically handle text as simple sequences of character strings. Therefore, specific preprocessing methods and algorithms are required in order to extract useful patterns (Hotho et al., 2005). In this sense, the text mining field deals with this machine supported analysis of text. The objective of text mining algorithms is to try and discover new and unknown information (patterns, keywords, concepts and other attributes) from large amounts of unstructured text data.

**Definition 1** (*Vector Space Model*) In the text mining domain, the Vector Space Model (VSM) is an algebraic model that translates the string contents of text documents into vectors. The components of these vectors could represent, for instance, the presence or absence of a term or a *n-gram* (a contiguous sequence of $n$ items from a given sample of text), the frequency of a term or n-gram in a given document ($tf$), or even the importance of a term ($tf$-$idf$)

In order to map documents into a vector space, it is necessary to create a dictionary of features present in the document, for example extracting all terms from a document and convert each of them into a dimension of the vector space. This process is usually complemented with other techniques such as stop word removal and stemming or lemmanization procedures.

**Definition 2** (*Stop words*) Stop words usually refer to the most common words in a language (such as articles or prepositions) and do not provide additional information when performing text analysis. Thus, they are usually removed from the text corpora in information retrieval approaches. The list of stop words is language dependent, and as such different text processing approaches use the same list of stop words.

**Definition 3** (*Stemming and lemmanization*) Stemming is the process of reducing words to their word root (stem). It allows to reduce document terms which are related to the same meaning but which appear in different morphological forms. Lemmatization is the process of grouping together the inflected forms of a word into the word's canonical form or lemma, so they are all considered as variations of the same unit. The difference between stemming and lemmanization is that stemming does not take into account the context, as it converts the words of a sentence to its non-changing portions. However, the stemming process is typically easier to implement and faster.

After applying these preprocessing steps to a group of documents (i.e. stop word removal and stemming or lemmanization), a dictionary of terms can be created, that is, an index vocabulary of the terms or n-grams present in the document corpus. Based on this vocabulary of terms, a document can be converted into a vector space by mapping each extracted term into each vector component. This mapping can be done, for example, in a binary representation of the presence / absence of a given term, or by the frequency of a particular term in a given document. For example, term frequency ($tf$) mapping of a document in the vector space can be represented as:



$$tf(t, D_n) = \sum_{d \in D} f(d, t) \tag{1}$$

where $D_n$ is the $n$ document of the document collection, $d$ is the vocabulary of terms for a given document $D_n$ and $f(d, t)$ represents the number of times the term $t$ is present in the document $D_n$. On the other hand, $f(d, t)$ is defined as:

$$f(d, t) = \begin{cases} 1, \text{ if } d = t \\ 0, \text{ otherwise} \end{cases} \tag{2}$$

A document $D_n$ can thus be represented in the vector space as:

$$\overrightarrow{\mathbf{v_n}} = (tf(t_1, D_n), tf(t_2, D_n), ..., tf(t_m, D_n)) \tag{3}$$

Usually, document collections are represented in matrix form with $N \times M$ shape and stored as a table structure for computation purposes (Buttcher et al., 2010), where $N$ is the cardinality of the document space, and $M$ is the number of features (or the vocabulary size in the above equations).

### 2.3 tf-idf and vector normalization

In information retrieval systems, the term frequency based representation of documents is not considered a good discriminator as it tends to scale up frequent terms, and scale down the less frequent ones which generally contribute with more information. To overcome this problem, the $tf-idf$ (term frequency - inverse document frequency) is employed instead.

**Definition 4** (*Term frequency - inverse document frequency*) $tf-idf$ is a statistical calculation that considers both the occurrence of a term in a given document along with the cardinality of the document space. Before computing the $tf-idf$ representation, as some repeated terms in documents (keyword spamming) or particularly large documents in a document collection could lead to bias towards particular terms, the term frequency representation of a document in a vector space is first normalized:

$$\hat{\mathbf{v_n}} = \frac{\overrightarrow{\mathbf{v_n}}}{\left\|\overrightarrow{\mathbf{v_1}}\right\|_p} \tag{4}$$

where the $\hat{\mathbf{v_n}}$ is the normalized vector, and $\left\|\overrightarrow{\mathbf{v_1}}\right\|_p$ is the norm of the vector $\overrightarrow{\mathbf{v_n}}$ in the $L^p$ space

The inverse document frequency ($idf$) takes into account the specificity of a given term (Sparck Jones, 1988), and can be quantified as an inverse function of the number of documents in which it occurs:

$$idf(t, D) = \log \frac{N}{1 + |d : t \in D|} \tag{5}$$

where $N$ is the total number of documents in the corpus ($N = |D|$) and $|d : t \in D|$ is the number of documents where the term $t$ appears. The $tf-idf$ is then calculated as:

$$tf-idf(t) = tf(t, D) \times idf(t) \tag{6}$$

The components of the mapped documents using the $tf-idf$ approach are usually referred to as weights ($w$), and thereby a document $D_n$ is represented in vector space as:

$$\overrightarrow{\mathbf{v_n}} = (w_{n1}, w_{n2}, ..., w_{nm}) \tag{7}$$

where $w_{nm}$ is the computed *tf-idf* weight of the $m$ term in the $n$ document. A high weight $w_{nm}$ is reached when there is a high frequency of the $m$ term in a given document $D_n$ and a low document frequency of the same term in the whole document collection.

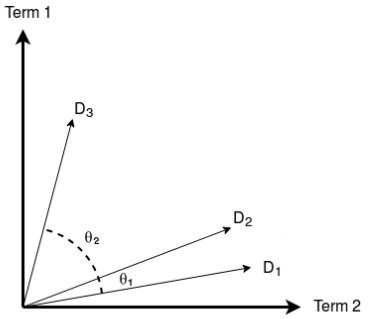


**Fig. 1** Image illustrating the cosine similarity between documents projected in two dimensions of the vector space that correspond to two different term frequency components. As $D_1$ ($\overrightarrow{\mathbf{v_1}}$) and $D_2$ ($\overrightarrow{\mathbf{v_2}}$) have similar component value for those terms their $\theta_1$ is lower. In the case of $D_1$ ($\overrightarrow{\mathbf{v_1}}$) and $D_3$ ($\overrightarrow{\mathbf{v_3}}$) their component value is dissimilar so they have a higher $\theta_2$

## 2.4 Cosine similarity

**Definition 5** (*Cosine similarity*) The cosine similarity between two non-zero vectors is based in the calculation of the cosine of the angle between said vectors. Thus, it is a metric that measures the difference in orientation between said vectors as opposed to their magnitude. In terms of the document space, instead of taking into consideration the documents' magnitude (as could be obtained by mere word count or $tf - idf$ computation), cosine similarity allows to consider the angle between document vectors in the projected space. The cosine similarity measure can be applied to any number of dimensions and is commonly used in positive spaces, as is the case in text mining where vectors are created so that each dimension in the vector space corresponds to the frequency of a particular term in a document (Tan et al., 2005).

Computing the cosine similarity between two vectorized documents $\overrightarrow{\mathbf{v_1}} = (w_{11}, w_{12}, ..., w_{1m})$ and $\overrightarrow{\mathbf{v_2}} = (w_{21}, w_{22}, ..., w_{2m})$ involves solving the equation of the dot product between those two vectors for the $cos(\theta)$:

$$\overrightarrow{\mathbf{v_1}} \cdot \overrightarrow{\mathbf{v_2}} = \|\overrightarrow{\mathbf{v_1}}\|\|\overrightarrow{\mathbf{v_2}}\| cos(\theta) \tag{8}$$

$$cos(\theta) = \frac{\overrightarrow{\mathbf{v_1}} \cdot \overrightarrow{\mathbf{v_2}}}{\|\overrightarrow{\mathbf{v_1}}\|\|\overrightarrow{\mathbf{v_2}}\|} = \frac{\sum_{i=1}^{m} w_{1i} w_{2i}}{\sqrt{\sum_{i=1}^{m} w_{1i}^2} \sqrt{\sum_{i=1}^{m} w_{2i}^2}} \tag{9}$$

where $w_{1i}$ is the *tf-idf* weight computed for the $i$ term in the document $D_1$ mapped as $\overrightarrow{\mathbf{v_1}}$.

As such, documents with great cosine value (close to 1) will point in similar direction in the vector space and as such will have great similarity, whereas vectors with a zero or close to zero cosine value will have perpendicular orientations and can be considered as uncorrelated.

**Table 2** Overall procedure for term recommendation in aided database search

| |
|---|
| **Input:** A collection of words related to the topic of interest in the form of a search string, $S$; a stop words vector, $SW$; a minimum cosine similarity distance $\theta_{min}$;<br>**Output:** A collection of recommended new terms $T$ for building a new search string $S_1$; |
| **01.** use S as input search on academic databases<br>**02.** Construct an abstract corpus D:<br>**02-A.** Remove duplicates<br>**02-B.** Remove entries with empty abstracts<br>**03** Project D on vector space $\overrightarrow{\mathbf{v_D}}$:<br>**03-A.** Lemmanization<br>**03-B.** Stop words removal based on $SW$<br>**03-C.** Compute *tf-idf* term values<br>**04.** Manual or SVM based labelling of D as relevant (R), and non-relevant (NR)<br>**05.** For each term, compute term weights $w_{tD}$:<br>**05-A.** Compute the sum of weights in R, $w_{ti}$<br>**05-B.** Compute the sum of weights in NR, $w_{tj}$<br>**05-C.** Compute $w_{tD}$ as ($w_{ti}$ - $w_{tj}$) / $\lvert R \rvert$<br>**06.** Suggest new terms $T$ based on $w_{tD}$ sorted values<br>**07.** Construct a new search string $S_1$ and repeat from **01.** |

**Table 3** Relevant abstract screening

| |
|---|
| **Input:** A collection of relevant abstracts, $D$; a collection of prototype abstracts $P$; a collection of minimum cosine similarity distances $\theta_{min}$;<br>**Output:** A fine tuned collection of relevant abstracts sorted by similarity, $D_1$; |
| **01.** Project $D$ on vector space $\overrightarrow{\mathbf{v_D}}$<br>**01.** Project $P$ on vector space $\overrightarrow{\mathbf{v_P}}$<br>**03.** For each $d_i$ in $D$ and $p_j$ in $P$:<br>**03.A** Compute cosine similarity distance between $\overrightarrow{\mathbf{v_{d_i}}}$ and $\overrightarrow{\mathbf{v_{p_j}}}$: $\theta_{d_i,p_j}$<br>**03.B** If $\theta_{d_i,p_j} > \theta_{min}$ remove $d_i$ from D<br>**04** Sort D based on $\theta_{d_i,p_j}$ |



## 3 Methodology

Taking into account the previous discussion, in this section we propose an iterative methodology to assist the information retrieval from scientific electronic databases. As stated before, searching for recent findings in a research domain remains a tedious task, as researchers usually find it difficult to decide on the keywords to find relevant information, and also look for means to fast screening the obtained search results in order to remove documents not related to the topic of interest. The proposed methodology aims to improve the access to knowledge by alleviating these searches and the associated screening of results.

The overall procedure involves different steps which include: text mining techniques for analysing the results obtained from a regular search in scientific databases, automatic information retrieval for recommending search terms that enhance the search string, machine learning algorithms for classifying the retrieved abstracts in terms of their relevance, and a final classification stage for quick identification of relevant documents related to the topic of interest. An overview of the procedure is shown in table 2 and table 3.

Thus, the process starts with a regular literature search using the search engine(s) of one (or various) academic database(s) using one or more search terms based on the topic of interest and the researches knowledge on the field. As most common electronic academic databases allow to download search results including the abstract of the obtained list of documents, a document corpus is created from the downloaded abstracts. However, an initial preprocessing step is necessary in order to construct a consistent abstract corpus from the downloaded results, which will involve structuring the data downloaded from different sources, and removing duplicates along with those entries with no abstract information. The resultant abstract corpus will be denoted as $D$.

The next stage involves text mining for recommended term extraction (*term* may refer to a word or a n-gram), and is based on the one proposed by (Mergel et al., 2015). First, documents in the corpus are manually labelled as relevant (R) and non-relevant (NR), depending on their relation to the search topic of interest. A *tf-idf* based approach (section 2.3) is then applied in order to project the abstracts into the vector space (section 2.2) and obtain recommended terms from the corpus. A prior lemmanization of terms is performed in order to group together the inflected forms of the same word (section 2.2). For each term $t$, it involves:

- calculating the sum of *tf-idf* weights for each term $t$ in the R group of documents: $\sum_{R_i \in R} w_{ti}$. Where $w_{ti}$ is the computed *tf-idf* weight for the document $R_i$ in the documents marked as R.
- calculating the sum of *tf-idf* weights for each term $t$ in the NR group of documents: $\sum_{NR_j \in NR} w_{tj}$. Where $w_{tj}$ is the computed *tf-idf* weight for the document $NR_j$ in the documents marked as NR.
- calculating the final frequency based value for the term $t$ by subtracting the non-relevant frequency value from the relevant frequency value, and then dividing the result by the number of documents marked as relevant ($|R|$ or the cardinality of R group):

$$w_{tD} = \frac{\sum_{R_i \in R} w_{ti} - \sum_{NR_j \in NR} w_{tj}}{|R|} \tag{10}$$

The above process produces a list of term weights which can be sorted in order to obtain recommended relevant search terms for refining the search. Thus, terms with higher $w_{tD}$ value are expected to be frequent in the relevant documents and in consequence more related to the search topic, whereas those with lower $w_{tD}$ value are expected to be terms that should not be considered in the search.

The described steps can be iterated through consecutive searches in order to refine the search string. However, usually the amount of information retrieved from a search increases if new terms broaden the search scope, making the process of manually labelling the documents as relevant or not relevant non-viable. For that reason, prior to subsequent iterations our methodology includes supervised machine learning techniques to automate this step.

The vectorized documents are categorized based on the previous labelling of relevant and non-relevant documents and a Support Vector Machine (SVM) is trained based on said categorization and considering vector weights as features.

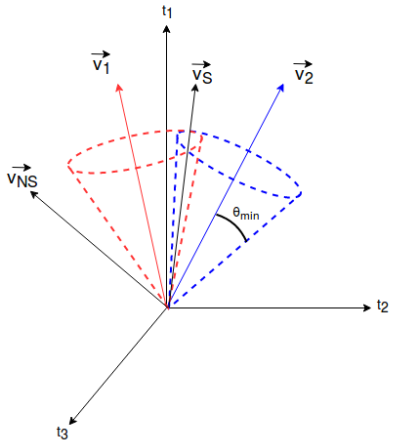


**Fig. 2** 3D representation of cosine similarity based document selection

Over the years, SVMs have been proven as one of the most powerful learning algorithms for text categorization (Joachims, 1998), (Islam et al., 2017), as they have the potential of handling large feature spaces. The trained SVM can then be used for automatic labelling of the downloaded documents (abstracts) in subsequent iterations.

The previous steps may be repeated until obtaining a search string that produces a good amount of relevant results on the topic of research. As a final step, in order to ease the process of fast screening and fine tune the final group of relevant retrieved documents (section 2.1) a cosine similarity approach is carried out (section 2.4). To do so, it is necessary to provide the algorithm with a set of prototype documents which the researcher considers that better resemble the intended output. This set of documents can be easily obtained from any of the groups of relevant documents obtained during iterations.

After doing so, the cosine similarity coefficient can be computed between the documents resultant from the last search and those in the prototype set, obtaining a list of vector document similarities towards the ideally desired output. The obtained similarity values can then be used to screen, sort or even trim, the final retrieved document corpus. For example, the researcher could fine tune the results by trimming the obtained group of documents based on a minimal distance $\theta_{min}$ to the documents in the prototype set, which in result is a form of classification of the document vectors in the vector space.

Figure 2 is a 3D graphical representation of the above example in the vector space, where $\overrightarrow{\mathbf{v_1}}$ and $\overrightarrow{\mathbf{v_2}}$ are two selected prototype documents. By choosing a minimum cosine distance to each prototype document vector ($\theta_{min}$), an imaginary cone is set around each of them which in turn will determine which documents are selected based on their vector orientation. Thus, vector documents that lie within the intersection of both cones are selected ($\overrightarrow{\mathbf{v_S}}$), whereas those which orientation lies outside the intersection are rejected ($\overrightarrow{\mathbf{v_{NS}}}$). This can be extrapolated to $n$ vectors in a $m$ dimensional space.

**Table 4** Evolution of search strings and top five relevant obtained terms

| Search string | Top 10 relevant obtained terms |
|---|---|
| ecosystem AND elderly | care, home, service, platform, older, activities, aal, technology, project, digital |
| (ecosystem OR platform) AND (elderly OR care) - RELEVANT | applications, monitoring, social, healthcare, mobile, health care, services, technologies, sensor, devices |
| (ecosystem OR platform) AND (elderly OR care) - CITED | health, services, health care, system, home, service, social, people, medical, design |
| (ecosystem OR platform) AND (elderly OR care OR medical) | health, system, services, health care, information, home, mobile, people, based, data |
| (ecosystem OR platform) AND (elderly OR care OR aal) | services, system, health care, home, service, people, mobile, social,monitoring, user |

**Table 5** Classifiers for a total of 350 documents in a 7 fold cross validation

| Classifier | Features | True NR Rate (tNR/tNR + fR) | True R Rate (tR/tR + fNR) | F1 score |
|---|---|---|---|---|
| Multinomial Naive Bayes | term frequencies | 0.907 | 0.893 | 0.902 |
| Bernoulli Naive Bayes | term ocurrences | 0.894 | 0.918 | 0.899 |
| KNN | term frequencies | 0.944 | 0.874 | 0.919 |
| SVM | term frequencies | 0.969 | 0.909 | 0.940 |



## 4 Implementation

We have implemented the methodology described in section 3 in a collection of python scripts that use scikit-learn, nltk and pandas (available at http://bit.ly/2PwL37v). The scripts allow to set different values, such as a minimum or maximum frequency value for a term or n-gram to be taken into account in the mapping into the VSM, or to provide a list of additional stop words. Also, different classifiers can be trained from a manually labelled abstract corpus, provided that it is constructed in .csv format. Finally, a script for cosine similarity computation is also included.

### 4.1 A working example

One of our current research projects focuses on technological ecosystems for elderly people and/or people with special care needs, so we were interested in conducting a SLR on that particular topic. In this case, the problem arises when trying to build the search string as the topic of interest consists in two very different and unrelated domains (technological ecosystems and elderly care) which implies that: few results are obtained when the search is restricted to those search terms; as the concept of "technological ecosystems" has emerged recently it is difficult to know in advance how the care domain refers and relates to it; and as the care domain includes many different disciplines, it produces thousands of uncorrelated results when broadening the search which are difficult to screen. Using the above scripts, we have performed an iterative document search for term recommendation and abstract screening in the Scopus and WoS databases. A more comprehensive review on the problematic and results is discussed in our forthcoming paper (Marcos-Pablos and García-Peñalvo, in press).

As part of the above work, table 4 shows an example of the evolution of the search strings obtained through iterations of our methodology as well as the top 10 relevant terms retrieved from the respective searches in the Scopus database (results are also included in http://bit.ly/2PwL37v). It has to be noted that the Scopus database only allows to download 2000 results with abstracts at a time in .csv format. To maintain a broaden search and for the sake of the example we didnt apply any additional refinement to the results. However, in order to speed up the process between iterative searches we decided to trim the output and download just the first 2000 results after ordering them by relevance and citation count (second and third entries in table 4).

The format of the search strings presented in table 4 is based on the advanced search features in most of the scientific databases (including Scopus) where logical connectors (AND, OR, NOT, etc.) can be employed to establish a logical relationship between the terms. Terms connected by AND will retrieve entries related to both terms, terms connected by OR will retrieve entries related to either term and so on. Each iterative search is based on the recommended terms obtained in the previous search. Following the methodology as described in table 2 we start with the term ecosystem, as including the term technological drastically limits the obtained results, and combine it with the term elderly. After applying the developed scripts, recommended terms are analysed and selected according to their relevance on the topic of interest to perform a new search, and concatenated with the previous terms using the adequate logical connectors.

It can be seen that terms like health, health care or services are obtained throughout consecutive searches, whereas terms like project, application or activities are dismissed. Logically, results will be biased by the new incorporated terms in the search string, so it is recommendable to follow the methodology described in table 3 for relevant abstract screening also between consecutive searches. This allows to assess if the obtained output keeps relation with the topic of interest (i.e. a reasonable amount of retrieved abstracts meet the condition of a minimal distance $\theta_{min}$ to the documents in the prototype set), or if on the contrary it has been highly diverted to undesirable results. For example, the terms health or health care appear as relevant in all searches, but if included in the search string produce a great bias towards results that fall within the province of strictly health related sciences which are not the subject of the search.



### 4.2 Classifier evaluation

The created scripts allow to train different classifiers in order to label documents in the abstract corpus as relevant or non-relevant. Using the manually labelled list of documents obtained from the first search example in table 4, we have compared the performance of different classifiers for document labelling. The computed *tf-idf* values are employed as classification features for all classifiers except for the Bernouli Naive Bayes. In this case, as it is designed for binary/boolean features, instead of a numeric vector representing term frequencies as features, it employs a vector of booleans representing the presence or absence of an term. To test the performance of the trained classifiers, we performed a 7-fold cross-validation, computing the F1 score for each fold and then averaging for a mean accuracy on the entire set.

The results for each of the classifiers are shown in table 5. The first column of values displays the true negative rate of the classifiers, also called specificity, which indicates the proportion (between 0 and 1) of abstracts belonging to the NR group that are identified as such. The second column of values displays the true positive rate, also called sensitivity, which indicates the proportion of abstracts belonging to the R group that are correctly identified as such.

It can be seen that, in terms of specificity, the SVM classifier outperforms the others, being able to better identify abstracts that are not relevant to the topic and dismissing false negatives. However, in terms of correctly identifying relevant abstracts avoiding false positives the Bernoulli Naive Bayes gets a slightly better outcome than the SVM approach. Finally, the F1 score combines both the precision (the proportion of true positives on all relevant predictions) and the recall (the proportion of true positives on all relevant abstracts, which is equivalent to the sensitivity) into a single value. If one of these two values decreases, the F1 score also does. It can be seen in terms of the F1 score the SVM clearly outperforms the other classifiers.

In our particular case, as the final goal of the classification is to extract suggested search terms by combining abstracts from both the relevant and non-relevant groups (table 4), it is important to avoid both false positives and false negatives (high specificity and sensitivity). On the other hand, during the screening stage it is important that all results which are marked as relevant effectively belong to the relevant group (high precision), and that the classifier is able to identify most of the relevant abstracts from the abstract corpus (high recall). For these reasons, the SVM classifier is the one that better suits our approach.



## 5 Conclusions

In the present paper, an approach for aiding in the process of systematic literature reviews (SLR) and particularly during literature search in scientific databases has been presented, by using an iterative methodology that applies text mining and machine learning techniques to a downloaded corpus of abstracts. The proposed methodology can be very useful in order to assist the task of SLR conduction, which involves managing and processing large amounts of unstructured text data. Also, a working example that employs a set of developed scripts has been presented. The final goal of the proposed methodology is to allow the researcher to come up with a search string that retrieves as many results as possible related to the topic of interest.

The described approach is specially helpful in those cases where the search topic is difficult to establish or delimit, forcing the researcher to broaden the search scope and producing large amounts of highly diverse abstracts. Rather than a drawback, this high volume and variety of retrieved documents may be used as means of exploring the limits of the search scope, as a correct analysis of data may enable to discover new knowledge which may be hidden within the search output.

It has to be noted that the proposed methodology is not intended to fully automate the process of search string construction or abstract screening but to be combined with the researchers' knowledge on the field. During the initial stages of performing a systematic literature review that involve designing the review methodology, devising the search strategy, performing the search and screening

the results both creative and technical skills are needed. Also, often these stages are peer-reviewed so to ensure the fulfilment of the research goals, and usually searches are first piloted and the searching protocol and procedures are redesigned according to the obtained results. For those reasons, the approach described in this paper combines the automatic processing of the results with tools for supervised decision making in an iterative process, so as to tune and adapt the search based on the researchers' judgement.

Results show that the suggested procedure is able to produce relevant terms related to the topic of interest, but that researchers' assessment is needed in order to produce an unbiased output. Also in this sense, results obtained from different classifiers for automatic abstract labelling have been presented, where the SVM classifier has been selected as the one that performs best considering our approach. In this case, the reviewer would train the classifier to make the required specific judgement, the SVM would employ this judgement in order to classify abstracts in terms of relevancy, and finally the researcher would employ the proposed screening method to appraise the SVM output.

A limitation of the proposed methodology relies in employing only the retrieved abstract from each study. As the abstracts of publications contain way less the information that is contained in the full paper, which could lead to a bias in the search results. Also, as there could be different configurations of the parameters involved in the proposed stages (e.g. the minimum document frequency for a term to be considered, the stop words to be excluded, etc.) further analysis needs to be performed in order to come up, if possible, with the best set of parameters.

**Acknowledgements** This work has been partially funded by the Spanish Government Ministry of Economy and Competitiveness throughout the DEFINES project (Ref. TIN2016-80172-R) and the Ministry of Education of the Junta de Castilla y León (Spain) throughout the T-CUIDA project (Ref. SA061P17).

**Compliance with ethical standards**

**Conflict of interest.** The authors declare that they have no conflict of interest.
**Ethical approval.** This article does not contain any studies with human participants or animals performed by any of the authors.



## References

Al-Ruithe M, Benkhelifa E, Hameed K (2018) A systematic literature review of data governance and cloud data governance. Personal and Ubiquitous Computing DOI 10.1007/s00779-017-1104-3

Buttcher S, Clarke C, Cormack GV (2010) Information Retrieval: Implementing and Evaluating Search Engines. The MIT Press

Eachempati P, Srivastava PR (2017) Systematic literature review of big data analytics. In: Proceedings of the 2017 ACM SIGMIS Conference on Computers and People Research, ACM, New York, NY, USA, SIGMIS-CPR '17, pp 177–178, DOI 10.1145/3084381.3084422

Felizardo KR, Nakagawa EY, Feitosa D, Minghim R, Maldonado JC (2010) An approach based on visual text mining to support categorization and classification

in the systematic mapping. In: Proceedings of the 14th International Conference on Evaluation and Assessment in Software Engineering, BCS Learning & Development Ltd., Swindon, UK, EASE'10, pp 34–43

Franco-Bedoya O, Ameller D, Costal D, Franch X (2017) Open source software ecosystems: A systematic mapping. Information and Software Technology 91:160 – 185, DOI https://doi.org/10.1016/j.infsof.2017.07.007

Gandomi A, Haider M (2015) Beyond the hype: Big data concepts, methods, and analytics. International Journal of Information Management 35(2):137 – 144, DOI https://doi.org/10.1016/j.ijinfomgt.2014.10.007

Hordri NF, Samar A, Yuhaniz SS, Shamsuddin SM (2017) A systematic literature review on features of deep learning in big data analytics. International Journal of Advances in Soft Computing and its Applications 9(1):32–49, DOI https://doi.org/10.1016/j.ijinfomgt.2014.10.007

Hotho A, Nrnberger A, Paa G (2005) A brief survey of text mining. LDV Forum - GLDV Journal for Computational Linguistics and Language Technology

Islam MS, Jubayer FEM, Ahmed SI (2017) A support vector machine mixed with tf-idf algorithm to categorize bengali document. In: 2017 International Conference on Electrical, Computer and Communication Engineering (ECCE), pp 191–196, DOI 10.1109/ECACE.2017.7912904

Joachims T (1998) Text categorization with support vector machines: Learning with many relevant features. In: Nédellec C, Rouveirol C (eds) Machine Learning: ECML-98, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 137–142

Kitchenham B, Charters S (2007) Guidelines for performing systematic literature reviews in software engineering

Labrinidis A, Jagadish HV (2012) Challenges and opportunities with big data. VLDB Endowment 5(12):20322033

LHeureux A, Grolinger K, Elyamany HF, Capretz MAM (2017) Machine learning with big data: Challenges and approaches. IEEE Access 5:7776–7797, DOI 10.1109/ACCESS.2017.2696365

Marcos-Pablos S, García-Peñalvo F (in press) Decision support tools for slr search string construction. In: Proceedings of the 6th International Conference on Technological Ecosystems for Enhancing Multiculturality, ACM, New York, NY, USA, TEEM 2018

Marshall C, Brereton P (2013) Tools to support systematic literature reviews in software engineering: A mapping study. In: 2013 ACM / IEEE International Symposium on Empirical Software Engineering and Measurement, pp 296–299, DOI 10.1109/ESEM.2013.32

Mayer-Schnberger V, Cukier K (2013) Big Data: A Revolution That Will Transform How We Live, Work, and Think. Houghton Mifflin Harcourt, Boston, MA, USA

Mergel GD, Silveira MS, da Silva TS (2015) A method to support search string building in systematic literature reviews through visual text mining. In: Proceedings of the 30th Annual ACM Symposium on Applied Computing, ACM, New York, NY, USA, SAC '15, pp 1594–1601, DOI 10.1145/2695664.2695902

Nelson B, Olovsson T (2016) Security and privacy for big data: A systematic literature review. In: 2016 IEEE International Conference on Big Data (Big Data), pp 3693–3702, DOI 10.1109/BigData.2016.78410372

Olorisade BK, de Quincey E, Brereton P, Andras P (2016) A critical analysis of studies that address the use of text mining for citation screening in systematic

reviews. In: Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering, ACM, New York, NY, USA, EASE '16, pp 14:1–14:11, DOI 10.1145/2915970.2915982

O'Mara-Eves A, Thomas J, McNaught J, Miwa M, Ananiadou S (2015) Using text mining for study identification in systematic reviews: a systematic review of current approaches. Syst Rev 4:5

Petticrew M, Roberts H (2008) Systematic reviews in the social sciences: A practical guide. John Wiley & Sons

Ros R, Bjarnason E, Runeson P (2017) A machine learning approach for semi-automated search and selection in literature studies. In: Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering, ACM, New York, NY, USA, EASE'17, pp 118–127, DOI 10.1145/3084226.3084243

Sparck Jones K (1988) Document retrieval systems. Taylor Graham Publishing, London, UK, UK, chap A Statistical Interpretation of Term Specificity and Its Application in Retrieval, pp 132–142

Tan PN, Steinbach M, Kumar V (2005) Introduction to Data Mining, (First Edition). Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA

Tsafnat G, Glasziou P, Choong MK, Dunn A, Galgani F, Coiera E (2014) Systematic review automation technologies. Syst Rev 3:74