

# Decision support tools for SLR search string construction

Samuel Marcos-Pablos  
GRIAL Research Group,  
Research Institute for Educational Sciences,  
University of Salamanca  
37008 Salamanca, Spain  
samuelmp@usal.es

Francisco José García-Peñalvo  
GRIAL Research Group,  
Research Institute for Educational Sciences,  
University of Salamanca  
37008 Salamanca, Spain  
fgarcia@usal.es

## ABSTRACT

Systematic literature reviews (SLRs) have gained popularity during the last years as a form of providing state of the art about previous research. As part of the SLR tasks, devising the search strategy and particularly finding the right keywords to be included in the search string is a difficult and critical step, as it will determine what evidence will be identified in the different searched sources and thus condition the rest of the review. In order to support the search process, this paper presents an iterative methodology for search string construction along with a set of decision support tools that help in building the search string by finding appropriate key terms related to the topic of interest in order to assist the researcher in the SLR conduction.

## CCS CONCEPTS

• **Applied computing** → **Document searching** • **Applied computing** → **Document analysis** • *Information systems* → *Document filtering* • *Information systems* → *Recommender systems*

## KEYWORDS

Systematic literature review, decision support tool, text mining.

## 1 INTRODUCTION

Literature reviews, and more specifically Systematic Literature Reviews (SLR) have reached a considerable level of adoption in many research fields. A literature review is a search and evaluation of the available literature in a given topic, providing state of the art about previous research and giving an overview of what are the strengths of the area of interest and which weaknesses need further improvement. Different methodologies have been proposed for performing literature reviews but most of them share the same base structure defined the SALSAs (Search, Appraisal, Synthesis and Analysis) framework [6]. Within this framework, the aim of the Search phase is to gather a preliminary list of publications to analyze. During the Appraisal stage the papers collected during the previous stage are analyzed, eliminating those that are irrelevant. Finally, the Synthesis and Analysis stages extract relevant data from the selected papers and draw conclusions from them in order to produce a final report.

A Systematic Literature Review (SLR) is a form of literature review that has gained popularity during the last decade. SLRs use a well-defined methodology to identify, analyze and interpret all available evidence related to a specific research question in a way that is unbiased and (to a degree) repeatable [1]. When performing a systematic literature review, the main SLR stages can be mapped against the SALSAs framework. However, different SLR guidelines propose different number and order of discrete activities to be carried out depending on the resolution of the protocol. For example, in [2] they divide the process of performing a systematic review into 15 steps and their associated tasks (see Table 1).

As shown in Table 1 the amount of work needed to carry out a systematic review is immense, and in many cases it is necessary to pilot some stages in order to find mistakes in the protocol (e.g. how the data is collected, the methodology to address the research questions, the results extraction and synthesis methods, etc.) [1]. This means having to redo certain stages of the SLR, which in turn makes the process to become even more tedious and complex. Thus, undertaking a systematic review may require a large amount of resources and can take several months or even years.

The process of performing a SLR is in part creative and part technical [2]. In the initial stages of the process, which involve the definition of the review protocol and the search strategy, the tasks rely mainly on the researcher's knowledge, experience and

judgment, whereas once the review protocol has been established more technical and repetitive tasks are involved in the following stages.

Automatization can ease the process of Systematic Reviewing, 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.

Although the ideal would be to automatize the whole SLR process, some tasks are not amenable to automation or have not been examined for the potential to be automated. Decision support tools for these tasks could be considered instead [12].

The present paper presents a set of decision support tools that help in the process of building the search string in order to assist the task of SLR conduction. As part of our current research, we wanted to perform a SLR focused on ecosystems aimed to provide services to elderly dependent people. We faced the problem of building a relevant search string that could provide meaningful outputs from the scientific databases related to that particular topic. To overcome those problems, we developed a tool that applies Text-Mining techniques and Machine Learning algorithms in order to extract recommended relevant terms for building the search string and easy the task of screening the returned abstracts. The following sections describe the related work, the problem in detail as well as the developed tools along with usage examples.

## 2 RELATED WORK

There have been different proposals regarding the automatization of different stages of the SLR process, most of them being focused on the technical and repetitive tasks in the retrieval, appraisal and synthesis stages (Table 1). Rather than performing a complete state of the art on the field which would be out of the scope of the present paper, we will refer to a very recent scientific report from the European Food Safety Authority which deeply describes the existing automation methods for the different tasks of a systematic literature reviews [7].

However, less interest has been given on the preparation stage, and particularly 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 different studies show that many researches find difficulties when constructing the search string [4], this task has mainly been left to the knowledge and judgment of the researchers conducting the review.

**Table 1. Different steps, tasks and stages of a SLR. Adapted from [2]**

Task	Description
1. Formulate review question	Decide on the research question of the review
2. Find previous systematic reviews	Search for SR that answers 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

There are some examples in the literature though, as in [3], where they explore a machine learning approach to support semi-automated search and selection in SLRs. Or in [4], where they propose an iterative visual text mining process as a decision-support tool for building the search string. However, when looking for implemented solutions supporting the specific task of search string building, we found that there are a little number of functional tools accessible. The most convenient database for searching for available tools for SLRs is the Systematic Review Toolbox, which is a community-driven, searchable, web-based catalog of tools that support the systematic review process across multiple domains [5]. Performing a search with the following search criteria:

**(Underlying Approach: "any"; Discipline: "any"; Cost: "any"; Features: ("Protocol Development" OR "Automated Search"))** provides a total of 54 related software tools. Evaluating the results, we found two that could be of interest: *Search Builder 1.0*, which claims to be an Excel-based search string builder for Embase and Pubmed but which is not available, and *SLR.qub*, which is the implementation of the algorithm described in [4]. As for the rest of the results, most of them are focused on clustering and evaluating already performed search results, many of them have scarce or no documentation at all, and many others have been discontinued.

As such, the only relevant tool that could adequate to our requirements is the one proposed by [4]. However, it presents the inconvenience of only working on the IEEE database and it requires manual labeling of the obtained search results in order to iterate through the search string refinement process.

### 3 PROBLEM STATEMENT

As part of one of our current research projects, we were interested in performing a Systematic Literature Review (SLR) focused on ecosystems aimed to provide services to elderly people and/or people with special care needs.

Following the steps proposed by [1] and [2], as part of the research strategy a SLR must include preliminary searches aimed at both identifying existing systematic reviews and assessing the volume of potentially relevant studies. Having decided which sources to search, the next step is to identify the search terms or keywords to be included in the search string. As stated in [4], building the search string is one of the main challenges when performing an SLR. In order to answer the proposed research questions, a Systematic Literature Review needs to be capable of capturing all possible results that relate to the topic of interest. With that aim, the employed search string must contain key terms related to that topic, so that the obtained results are relevant.

In this sense, Kitchenham [1] suggests identifying the main search terms from the PICOC model (Population, Intervention, Comparison, Outcomes and Context) as well as from the research questions. However, as we were interested in two populations simultaneously, we started building our search string based on our population dimension: ecosystems on the one hand, and care & assistive services for older or dependent people on the other.

From here, a common approach for building the search string is to manually choose the search terms taking the researchers' knowledge of the field as a basis, and refining the chosen terms by performing pilot searches until a search string that produces adequate results is found. Alas, in our particular case piloting different searches in the databases with combinations of different considered relevant terms gave us very diverse results. An example of this variability of results with four different search strings used in the Scopus database is shown in Table 2. Searches were performed on Article title, Abstract and Keywords and with no year of publication range limit:

**Table 2. Examples of pilot searches on the Scopus database and obtained results (as for May - 2018)**

Search	Employed string	Number of results obtained
1	ecosystem AND older person	7
2	ecosystem AND elderly	258
3	ecosystem AND (elderly OR care)	3,997
4	("digital ecosystem" OR "software ecosystem" OR "technological ecosystem") AND (elderly OR care)	5

As for the first considered population, performing different searches with the term "ecosystem" if included as a single search keyword produces a great amount of results that are not related to the computer sciences field (Table 2). On the contrary, during those pilot searches, we identified several combinations of terms that are employed to refer to ecosystems from a computer science related perspective: Software ecosystems, digital ecosystems, and technological ecosystems, [9]. If we restrict our search as to include combinations of those terms, the obtained results are more related to the computer sciences field, but the amount of outputs decreases drastically. This result is consistent to what is stated in [1]: *the amount of literature on a certain topic may be small in the field of software engineering, hence it might be of use to search for literature in connected fields.*

In terms of the second population, research related to providing care and other services for older and dependent persons has become one of the hot topics during the last years, mainly in developed countries where there is an inverted population pyramid. The range of different technologies employed in this field varies from robotics, to medical devices [8]. For this reason, it was hard to figure out the proper research terms to address this population. In addition, scarce results were obtained if trying to use inclusive and appropriate language when searching the databases. For example, the term "elderly" is still widely employed by the scientific community, even though it is not the recommended term to be used when referring to people of advanced age [10]. Table 2 shows some examples of the amount of results obtained in the Scopus database for different combinations of search terms.

#### 4 PROPOSED APPROACH

In order to overcome the problematic described above in terms of establishing our search string, we employed a mixed approach. On the one hand, we selected a set of search terms for both population dimensions based on our previous knowledge and the performed pilot searches. On the other hand, we propose a methodology for looking over relevant search terms, which could additionally be employed for aiding the subsequent abstract screening stage. The following sections describe each step, a set of developed decision supporting tools and provide examples of usage along with the obtained results.

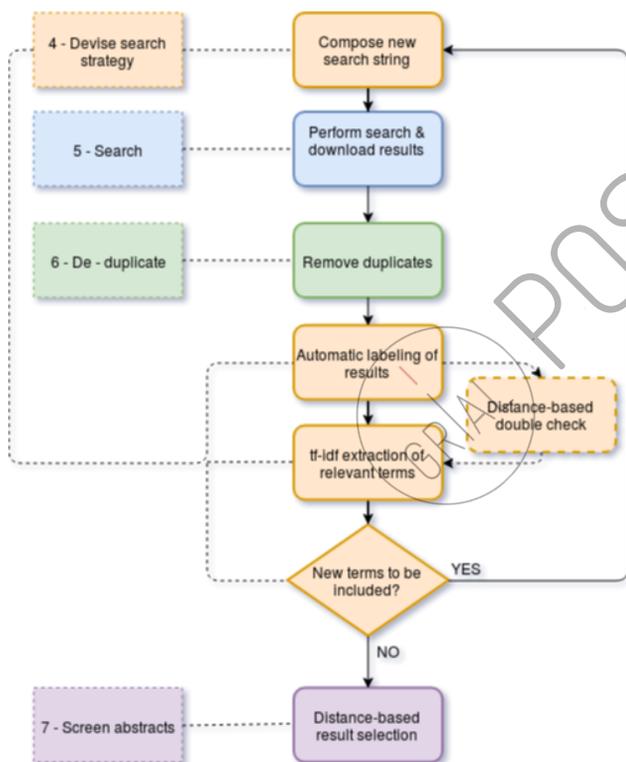


Figure 1. Diagram of the proposed approach along with the associated SLR tasks described in [2] (left)

##### 4.1 Extraction of relevant terms

In order to look for search terms that could be relevant for the topic of interest, we started by using a similar text mining methodology as the one proposed in [4]. In their paper, they suggest an iterative methodology for key term search recommendation, which basically involves the following steps:

- Perform an initial broad search with a search string built from what researchers consider relevant search terms
- Label resulting studies as relevant (R) or non-relevant (NR) for the current review
- Use Text mining techniques on both R and NR groups of documents to extract and expose new key terms and detect irrelevant former ones
- Update search string and return to the first step

The above process may be repeated until the search string produces refined results.

Following this methodology, we started with a broad search in the Scopus database. We chose this particular database for our tests because our institution has access to it, it includes scientific entries related to different research topics ranging from medicine to technical disciplines and it provides the results in .csv format (among others), which allows to directly import them with software libraries for data manipulation and analysis such as pandas.

As an initial search string, we used a general term for each of the population dimensions: ecosystem AND elderly, obtaining a total list of 258 publications in .csv format (<http://bit.ly/2JfdS4f>). We didn't apply any additional restrictions to our search (publication venue, year of publication, etc.).

Labeling the results as relevant and preprocessing them was performed based on titles and abstracts. Using a spreadsheet, we looked for possible duplicates and excluded those results that had no abstracts available or did not provide significant information about the publication content. After this process, 15 papers were excluded, 63 papers were labeled as relevant (R) and 175 papers were labeled as non-relevant (NR).

The next step consisted in extracting and exposing new key terms. To do so, the method proposed by [4] involves computing the tf-idf (term frequency - inverse document frequency) values of the different n-grams (a contiguous sequence of n terms) among the collection of abstracts (corpus) in order to assign them a value of relevance. The tf-idf is a statistic that reflects how important a word is to a specific document relative to all of the words in a collection of documents (the corpus). It increases proportionally to the number of times that a n-gram appears in the document, but is offset by the frequency of the n-gram in the corpus. As such, the following formula is applied:

$$tf - idf(t, D) = \left( \left( \sum_{di \in R} tf - idf(t, di) \right) - \left( \sum_{dj \in NR} tf - idf(t, dj) \right) \right) / |R| \quad (1)$$

where  $tfidf(t, D)$  is the final tf-idf value for a given n-gram  $t$ . The minuend is the frequency value obtained for that n-gram in studies marked as R, whereas the subtrahend is the frequency value in studies marked as NR. The subtraction is normalized by the cardinality of the R group. As an output, the above formula returns the tf-idf values of relevance for every n-gram identified in the abstracts.

In order to give an overview of the implications of the above formula, let us propose a simple example. Although due to the assumptions in the following calculations normalization would not be necessary, normalization is often used to compute term frequency, since terms in big text documents might have a greater frequency value than in smaller text documents, leading to a higher tf-idf value without necessarily indicating a greater relevance. Let us note that there are different normalization approaches such as adjusting for document length (divide tf by the number of words in the document), divide tf by the frequency of the most frequent term in that document, logarithmically scale the frequency, etc. Also, there are different approaches for computing the inverse term frequency, but we will consider the simplest of them all which consists in dividing the total number of documents by the number of documents containing the term, and then taking the logarithm of the result.

Let us consider that, after stop words removal, for a particular abstract in the Relevant group the term *technological* appears 1 time. If the total number of words in that abstract is 350, then the term frequency normalized by the abstract word count would be  $1/350 = 0.003$ . Now, if we assume we have 4000 abstracts marked as Relevant and the term *technological* appears in one-thousands of these, the inverse document frequency would be  $\log(4000 / 1000) = 0.6$ . Thus, the tf-idf weight for the term *technological* and for that particular abstract would be  $0.6 * 0.003 = 0,0018$ . Repeating the above computation for the term *technological* through all the abstracts in the Relevant group and summing the results would give us the minuend of the equation 1. For the sake of the example let's assume that all abstracts have a length of 350 words, so that the same tf-idf weight is obtained for that term in 1000 out of 4000 documents, so we obtain a minuend value of 1.8.

Now, let's assume that the term *paper* appears 1 time in all 3000 out of the total 4000 abstracts. As such, applying the same assumptions as above, we would obtain a tf of 0,003, but a idf of  $\log(4000 / 3000) = 0,12$  and a tf-idf value of  $0,003 * 0,12 = 0,0036$  and a minuend value of  $0,0036 * 3000 = 10,8$ .

It can be seen that tf-idf values tend to provide lower values to common terms across the abstract corpus, and could be considered as means of removing irrelevant terms such as stop words. However, as we compute the sum across all abstracts, common terms obtain a higher total sum value. On the other hand, performing the same calculations across the non-relevant abstract corpus (the subtrahend term in equation 1), common terms such as *paper* are expected to appear more frequently than non-common terms such as *technological*, and that difference of occurrence will be even higher for those terms that are more characteristic for the Relevant than the Non-Relevant group. As a result, the final value computed in equation 1 for "relevant non-common" terms in the Relevant group of abstracts will be higher, providing a list of words for term recommendation.

We have implemented the above method in a python script that uses scikit-learn, nltk and pandas (available at <http://bit.ly/2JfdS4f>). The script allows to set different values, such as a minimum or maximum frequency value for a n-gram to be taken into account, or provide a list of stopwords to be excluded from the analysis (stopwords are terms without informational value such as prepositions or articles, but the user could provide other undesired terms that are sometimes included in the downloaded abstracts such as the publisher name). After computing the tf-idf values, the script sorts them exposing the key relevant n-grams at the top. The top 10 key n-grams from our first search are summarized in Table 3.

From this output, we chose what we considered were the two most relevant terms that corresponded to each of the two population dimensions: platform and care and performed a new search in the Scopus database with the following string:

**(ecosystem OR platform) AND (elderly OR care)**

Limiting the output of this search to those documents that were written in English provided a final total of 22,076 results.

## 4.2 Automatic labeling of results

It is evident that as the search string broadens the search scope, the amount of obtained results increases, which in turn makes the manual labeling of abstracts a tedious or even unfeasible task. For that reason, in order to automatize the labeling process, we built another script that python-based classifier (also available at <http://bit.ly/2JfdS4f>). The developed script allows to select a training dataset (which should be a R - NR annotated .csv file) and choose between four classifier approaches to be trained: Multinomial Naive Bayes, Bernouli Naive Bayes, k-Nearest Neighbors and Support Vector Machines.

In order to train the classifiers, we used the manually labeled list of documents obtained from our first search as train data. The script loads the .csv file and computes the tf-idf matrix of values of all the n-grams obtained from the abstract corpus. A prior lemmatization of terms is performed in order to group together the inflected forms of the same word. The resulting tf-idf values are employed as features for classification 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 n-gram frequencies as features, it employs a vector of booleans representing the presence of absence of an n-gram.

To test the performance of the trained classifiers, the developed script allows to perform a k-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 4. It can be seen that, in our case, the SVM model outperforms the rest of the classifiers, except for a slight difference in the True R Rate when compared with the Bernouli classifier. However, as the script shuffles the dataset before performing the validation, this results may slightly vary between iterations.

In order to continue processing the output of our second search, 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 did not apply any additional refinement to the results.

However, in order to speed up the process between iterative searches we decided to sort the output and download just the first 2000 results. At this point, different options are available: for example, sort by relevance or sort by highest citation. If sorting by relevance, the common applied criteria across databases is to check how often does a search term occur in the document and usually where do search terms occur: in the title, in the summary, etc. which in turn will produce more related articles and narrow the scope of suggested terms. On the other hand, citation count can be a measure of quality and help broaden the obtained results. However, citation counts are often distributed exponentially, become more probable from year to year with a few articles garnering disproportionate attention, and there is also a positive correlation between the citation frequency of publications and the number of co-authors of the work [6].

The obtained results for both approaches are also shown in Table 3. along with two additional searches. All files and results can be consulted in the example folder (<http://bit.ly/2JfdS4f>).

### 4.3 Distance-based result selection

As a final step, in order to ease the process of screening the resulting abstracts a third script has been developed which fits the tf-idf matrix of abstracts n-grams into scikit-learn NearestNeighbors function. Thus, the script allows to compute the distance between a selected representative abstract and the rest of abstracts in the corpus.

The chosen measure for computing the distance is the cosine similarity. The cosine similarity is widely employed in document comparison because it is a measure that determines how related re two documents by looking at the angle instead of magnitude, which in turn can be seen as a comparison between documents on a normalized space as it does not consider the tf-idf magnitude but the projected angle over the tf-idf feature space.

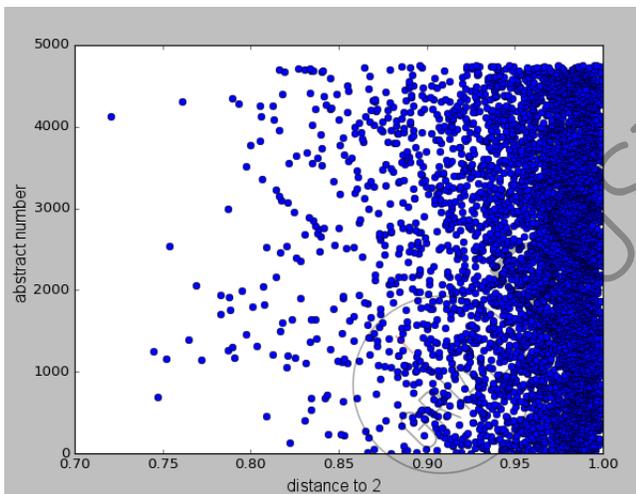
Sorting the distance results of the papers labelled as relevant provides not only means for fast screening the results for the following stages of the SLR, but also can be employed for double-checking the results of the automatic labelling step (see Figure 1), as relevant papers are expected to have lower distance to the prototype abstracts selected.

### 4.4 Final search string

The above search methodology and screens have been combined with other terms based on the knowledge obtained while piloting the searches in order to build a suitable search string for the SLR currently under development. Thus, our final search string was:

**(ecosystem OR seco OR osseco OR platform) AND (elderly OR "older people" OR care OR "Ambient Assisted Living" OR aal OR dependent OR dependant )**

The search has been conducted on both the Scopus and WoS databases, and the obtained results can be consulted at (<https://goo.gl/usZyrrw>). It has to be noted that in this case we made use of the different refinement fields available on each database search tool in order to limit the results by publication year, relevant publication sources, and fields such as computer sciences and health.



**Figure 2. Cosine distances between the abstracts in the corpus and one of the selected reference abstracts**

Also, in order to screen and limit the obtained results we have computed the distance between each of the abstracts in the corpus and three selected prototype abstracts that we consider resemble the intended output. After sorting the results by proximity, we have obtained a cluster of documents which are similar to those that resemble our search topic of interest. An extract of one of those abstracts (number 2 in the Scopus search sheet) is as follows:

*Effective provision of care and assistance services in ambient assisted living requires the involvement and collaboration of multiple stakeholders. To support such collaboration, the development of an ecosystem of products and services for active ageing plays an important role. This article introduces a conceptual architecture that supports such care ecosystem. In order to facilitate understanding and better interrelate concepts, a 3-layered model is adopted: Infrastructure layer, Care and assistance services layer and Ambient Assisted Living ecosystem layer...(continues)*

Figure 2 shows a plot of the obtained distance values for this final search and the above abstract.

## 5 CONCLUSIONS

In the present paper a set of decision support tools that help in the process of building the search string in order to assist the task of SLR conduction have been presented. This set of scripts makes use of Text Mining and Machine Learning techniques, which allow to extract relevant terms from the abstracts obtained through iterative searchers. The proposed iterative methodology can thus be followed and combined with the researchers' knowledge in the field in order to come up with a search string that allows to retrieve as many results as possible related to the topic of interest.

Although the ideal would be to automatize the whole process, this is not an easy task as the search strategy not only establishes what keywords will be used in searches, but also another steps which could in turn modify the selected search terms: what databases will be searched, what parameters should be employed in the search, if and how citations will be tracked, what evidence will be identified in non-database sources, etc. [2].

**Table 3. Evolution of searches and distances of the group of relevant and non-relevant abstracts (as for May - 2018)**

Search	R size	NR size	Top 10 relevant obtained terms (excluding those employed in the search string)
ecosystem AND elderly	67	175	care, home, service, platform, older, activities, aal, technology, project, digital
(ecosystem OR platform) AND (elderly OR care) - RELEVANT	1119	398	applications, monitoring, social, healthcare, mobile, health care, services, technologies, sensor, devices
(ecosystem OR platform) AND (elderly OR care) -CITED	396	1577	health, services, health care, system, home, service, social, people, medical, design
(ecosystem OR platform) AND (elderly OR care OR medical)	999	257	health, system, services, health care, information, home, mobile, people, based, data
(ecosystem OR platform) AND (elderly OR care OR aal)	1176	341	services, system, health care, home, service, people, mobile, social, monitoring, user

**Table 4. Obtained results for the different classifiers ('Total papers classified:', 350) 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 occurrences	0.894	0.918	0.899
KNN	term frequencies	0.944	0.874	0.919
SVM	term frequencies	0.969	0.909	0.940

In this sense, it has to be noted that although designing the search protocol is distinct from conducting the search, a correct search strategy will provide a review that is not biased. Thus, automation of the search design should be carefully undertaken. As can be extracted from the results in Table 3, the number of relevant obtained papers varies drastically when changing the database search parameters and retrieval strategy. Also, as can be observed in (<https://goo.gl/usZyrw>), the amount and relevancy of obtained results differs between databases, making it difficult to come up with a unified set of search parameters.

Those parameters can provide a more extensive search with the inclusion of more evidence in the review, or a better targeted search (i.e., increased precision) which can reduce the number of papers to be considered in later stages.

## 6 LIMITATIONS AND FUTURE WORK

The proposed search methodology and tools have proven to provide a convenient help in the selection of relevant terms. Also, one of the advantages of developing our own set of tools is that we gain control over every step in the supporting process, such as the machine learning algorithm and parameters, the text mining parameters (e.g. the minimum document frequency for a term to be considered, the maximum size of the n-grams, the stop words to be excluded) etc. On the other hand, as there are many different possible configurations, a further analysis needs to be performed in order to come up, if possible, with the best set of parameters.

Another limitation of the proposed methodology relies in employing only the retrieved abstract from each study. As stated in [1][11], 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 result.

Also, for that reason the computed distance similarity gives a small variability range, as can be observed in Figure 2. However, it has been proven to work reasonably well for an initial abstract assessment, allowing to discriminate relevant from non-relevant abstracts. But it should not be employed as a precise document selection tool as relevant documents could be mistakenly discarded.

For the above reasons, future work should include different studies to better assess the parameter selection of the different proposed algorithms, as well as looking alternatives for analyzing the relevant terms contained in the retrieved papers during the iterative search. Also, as suggested in [2] research into natural language processing techniques could provide means of identifying pertinent keywords from the research.

Last but not least, the development of a graphical user interface in order to easy the complete iterative process should be envisaged.

## ACKNOWLEDGMENTS

This work was partially supported by the MIUR-PRIN 2010-11 Project 2010ECA8P3 “DyNanoMag” and by the National Research Foundation, Prime Minister's office, Singapore under its Competitive Research Programme (CRP Award No. NRF-CRP 10-2012-03). This research work has been carried out within the University of Salamanca PhD Programme on Education in the Knowledge Society scope (<http://knowledgesociety.usal.es>) and was supported by the Spanish Ministerio de Educación, Cultura y Deporte under a FPU fellowship (FPU014/04783). 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).

## REFERENCES

- [1] B. Kitchenham and S. Charters. Guidelines for performing systematic literature reviews in software engineering. School of Computer Science and Mathematics, Keele University, EBSE Technical Report, EBSE-2007-01, 2007.
- [2] Tsafnat G, Glasziou P, Choong MK, Dunn A, Galgani F and Coiera E 2014. Systematic review automation technologies. *Systematic Reviews*, 3, 1-15.
- [3] Rasmus Ros, Elizabeth Bjarnason, and Per Runeson. 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 (EASE'17). ACM, New York, NY, USA, 118-127. doi:10.1145/3084226.3084243.
- [4] Germano Duarte Mergel, Milene Selbach Silveira, and Tiago Silva da Silva. 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 (SAC '15). ACM, New York, NY, USA, 1594-1601. doi:10.1145/2695664.2695902
- [5] Systematic Review Toolbox <http://systematicreviewtools.com> Accessed: 2018- 05- 15.
- [6] Grant, Maria J.; Booth, Andrew. A typology of reviews: an analysis of 14 review types and associated methodologies. *Health Information and Libraries Journal*, 26, pp.91-108, 2009.
- [7] Jaspers S, De Troyer E, Aerts M, 2018. Machine learning techniques for the automation of literature reviews and systematic reviews in EFSA. EFSA supporting publication 2018:EN-1427. 83 pp doi:10.2903/sp.efsa.2018.EN-1427
- [8] Calvaresi, D., Cesarini, D., Sernani, P. et al. 2017. Exploring the ambient assisted living domain: a systematic review. *J Ambient Intell Human Comput* (2017) 8: 239. doi:10.1007/s12652-016-0374-3
- [9] Konstantinos Manikas and Klaus Marius Hansen. 2013. Software ecosystems - A systematic literature review. *J. Syst. Softw.* 86, 5 (May 2013), 1294-1306. doi:10.1016/j.jss.2012.12.026
- [10] Mihelle Putnam. Replacing the elderly with older adults in jgsw publications. *Journal of Gerontological Social Work*, 58(3):229-231, 2015.
- [11] Pearl Brereton, Barbara A. Kitchenham, David Budgen, Mark Turner, and Mohamed Khalil. 2007. Lessons from applying the systematic literature review process within the software engineering domain. *J. Syst. Softw.* 80, 4 (April 2007), 571-583. DOI=<http://dx.doi.org/10.1016/j.jss.2006.07.009>
- [12] S. Marcos-Pablos and F. J. García-Peñalvo. 2019. Information retrieval methodology for aiding scientific database search. *Soft Computing* In Press. DOI:10.1007/s00500-018-3568-0.