Expert Systems with Applications 41 (2014) 1498–1508

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Automatic text classification to support systematic reviews in medicine

J.J. García Adeva [a,*], J.M. Pikatza Atxa [a], M. Ubeda Carrillo [b], E. Ansuategi Zengotitabengoa [b]

[a] Erabaki Group, Department of Computer Languages and Systems, University of the Basque Country UPV/EHU, Manuel Lardizabal 1, 20018 Donostia-San Sebastián, Spain
[b] Donostia University Hospital, 20014 Donostia-San Sebastián, Spain

ARTICLE INFO

Keywords:
Medical systematic reviews
Machine learning
Text mining
Text classification

ABSTRACT

Medical systematic reviews answer particular questions within a very specific domain of expertise by selecting and analysing the current pertinent literature. As part of this process, the phase of screening articles usually requires a long time and significant effort as it involves a group of domain experts evaluating thousands of articles in order to find the relevant instances. Our goal is to support this process through automatic tools. There is a recent trend of applying text classification methods to semi-automate the screening phase by providing decision support to the group of experts, hence helping reduce the required time and effort. In this work, we contribute to this line of work by performing a comprehensive set of text classification experiments on a corpus resulting from an actual systematic review in the area of Internet-Based Randomised Controlled Trials. These experiments involved applying multiple machine learning algorithms combined with several feature selection techniques to different parts of the articles (i.e., titles, abstract, or both). Results are generally positive in terms of overall precision and recall measurements, reaching values of up to 84%. It is also revealing in terms of how using only article titles provides virtually as good results as when adding article abstracts. Based on the positive results, it is clear that text classification can support the screening stage of medical systematic reviews . However, selecting the most appropriate machine learning algorithms, related methods, and text sections of articles is a neglected but important requirement because of its significant impact to the end results.

1. Introduction

Medical Systematic Reviews support the conversion of medical research into practice by bringing together the collection of existing studies that are relevant to a specific medical question. This synthesis of current evidence benefits different stakeholders such as clinicians and policymakers.

Although Systematic Reviews started as early as the 18th century (Lind, 1753), their production exploded after the second half of the 20th century along with a significant increment of publications in medical, nursing, and allied health care (Shonjania & Bero, 2001). Unfortunately, the significant growth of clinical trials in the last decades, has not been matched by a suitable number of systematic reviews produced (Bastian et al., 2010). An analysis of the situation at the time revealed that because the amount of work required to produce reviews is increasing, there was a majority of systematic reviews with many years out of date (Shojania et al., 2007).

The general process for creating a systematic review is based on three main steps: (i) conducting broad searches in the relevant literature, (ii) manually screenning titles and abstract of retrieved citations, and (iii) reviewing full articles of those citations identified as relevant. No matter how critical and necessary these steps are, they are very time consuming, especially the screening of citations and the review of candidate studies.

Multiple text mining techniques have been gaining popularity over the past years as a consequence of the ever increasing amount of available digital documents of unstructured text and, thus, the necessity of analysing their content in flexible ways (Hearst, 1999). From these techniques, one of the most prominent is text classification using machine learning, which consists of automatically predicting one or more suitable categories for unstructured texts written in natural language (e.g., English, Spanish, etc.). Text classification is currently a major research area with many commercial and research applications in a large number of domains. Medicine is one of the most evident areas where text mining methods have multiple applications, such as the discovery of new literature (Swanson, 1986), concept-based search (Ide, Loane, & Demner-Fushman, 2007), or automatic bibliographic update in clinical guidelines (Iruetaguena et al., 2013).

This work was motivated by the hypothesis that text classification could assist the production of Systematic Reviews by supporting reviewers in their process of manually screening published articles. Although this assumption is not new, as there has been recently an incipient while still modest body of research in this

* Corresponding author. Tel.: +34 943 018153.
E-mail address: jjga@ehu.es (J.J. García Adeva).

0957-4174/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2013.08.047

direction (Thomas, McNaught, & Ananiadou, 2011), our contribution is focused on: (i) studying the application of a comprehensive selection of machine learning algorithms, (ii) combining these algorithms with multiple feature selection methods and different numbers of features, (iii) selecting different parts of citations (i.e., title, abstract, or both), and (iv) applying these methods to the medical domain of Internet-Based Randomised Controlled Trials.

In such a way, an automatic text classification system could be trained with a set of articles from the medical domain in question after the collection of studies had been already manually screened. As these articles describing primary studies had been manually labelled as either *relevant* or *irrelevant*, they fit well with the paradigm of a two-class text classifier. Once the system was trained, it was ready to automatically classify unseen articles, therefore providing input into the screening process similarly to a human expert. In consequence, this system would not aim at replacing the persons involved in the decision process but to complement and assist them. Contrary to other previous studies covered by Section 4, where they directly selected either the abstract of the full article to train and test the classifiers, we were interested in investigating what sections of the articles provided the best results. We also applied a bigger variety of classifiers than other previous studies, in addition to multiple feature selection methods.

This paper is organised as follows. Section 2 describes the methods used in this work to automatically classify articles. Section 3 describes the manual process for performing systematic reviews in medicine and how it can be supported by text classification. Previous efforts in this area of research are described in Section 4. The design and analysis of the experiments proposed to validate our hypothesis is provided by Section 5 The paper concludes with Section 6, which also suggests some ideas for future work.

2. Text classification

Text mining consists of discovering of previously unknown information from existing text resources (Hearst, 1999). It is also called intelligent text analysis, text data mining, or knowledge-discovery in text. Text mining is related to data mining, which intends to extract useful patterns from structured text or data usually stored in large database repositories. Instead, text mining searches for patterns in unstructured natural language texts (e.g., books, articles, e-mail messages, Web pages, etc.). Text mining is a multidisciplinary field that includes several tasks such as text analysis, clustering, categorisation, summarisation, or language identification.

Text classification is one of key text mining tasks that has gained significant popularity over the last decade or so. One of the main reasons for it is the increasing amount of digital documents available and thus the necessity to access their content in flexible ways (Sebastiani, 2002). Text classification is also referred to as Text Categorisation, Document Classification, or even Topic Spotting. The current approach to text classification is applying the machine learning paradigm that uses of a set of previously categorised documents to automatically build a categoriser by learning from this data (i.e., inductive inference). As part of this whole process, each text document is represented by a feature vector, thus dismissing the order of words and other grammatical issues, as this representation is able to retain enough useful information for the classification task (Salton, 1989).

The next sections describe the sequential steps that shape text classification.

2.1. Document preprocessing

The preprocessing stage starts by tokenising documents. In this step, a text document is transformed into smaller units known as words or terms. It is common that the process also involves the removal of certain characters such as non-alphabetical ones, as well as converting them into lower case. After tokenisation, there are two further steps performed: removal of stop words and stemming of words.

A stop word is a term that is considered not to add significant semantic meaning to sentences. Therefore, they can be safely removed without affecting the whole meaning of the sentence. They mainly consist of topic-neutral words like articles and prepositions.

Stemming is the process of normalising words by applying morphological rules that allow a speaker to derive variants of the same idea to evoke an action (i.e., verb), an object or concept (i.e., noun), or a property (i.e., adjective) (Lovins, 1968). For example, the words *activate*, *activating*, *activeness*, *activation* are derived from the same stem *activ* and all share an abstract meaning of action or movement. Stemming does the reverse process, deducing the stem from a fully suffixed word according to its morphological rules. These rules concern morphological and inflectional suffixes. The former type usually changes the lexical category of words whereas the latter indicates plural and gender. Because most languages have a large number of word stems, applying this technique will most probably reduce the number of global unique terms in all the documents.

These three preprocessing procedures described above (tokenisation, stop-word removal, and stemming) are highly dependent on the language in question. Therefore, the preprocessing of documents can be considered to be language dependent.

2.2. Document modelling

After the documents have been preprocessed, the extracted information from each document is used to build a model representing that particular instance. Feature Selection contributes to this goal by reducing the overall dimensionality of terms, thus allowing the posterior creation of feature vectors to represent the documents. This step is crucial as machine learning algorithms usually work better on low-dimensional data, and they may require too much time or memory when the dimensionality of the data set is high (Salton, 1989). In other words, Feature Selection consists of choosing the subset that contains the most relevant terms of all the existing ones in the collection of training documents.

Because text classification depends on a well-defined set of categories, Feature Selection can be local or global. Global Feature Selection consists of generating a subset of terms from all the terms in all categories, while local Feature Selection creates a subset for each document category, where the most relevant features of the category are included.

Term Frequency (TF) is a very simple yet effective term evaluation function based on counting how many times each term appears across all documents.

The higher this count, the more relevant this term is considered. Document Frequency (DF) and inverse document frequency (IDF) are very similar to TF and are based on the count of documents each term appears in. The reason for having these two complementary functions is that in some cases, and depending on the characteristics of the document collection, the feature selection may work better when only the terms that appear in the most documents are kept, while in other situation it may be just the opposite.

The term evaluation function χ^2 calculates the dependence between the occurrences of a term and each category based on the number of expected vs observed occurrences.

After feature selection, the documents are then modelled. A very commonly used algebraic model is the Vector Space Model (VSM), which represents text documents in a high-dimensional

space where each of its dimensions corresponds to a word in the document collection and each document is represented by a feature vector. One of the most common methods is term frequency related to its inverse document frequency (TF/IDF) (Baeza-Yates & Ribeiro-Neto, 1999; Salton & Buckley, 1988), that estimates a weight for each term in a document, where the term frequency in the given document offers a measure of the relevance of the term within a document, while the document frequency is a measure of the global relevance of the term within a collection of documents. In particular, considering a collection of documents D containing n documents, so that $D = \{d_0, \ldots, d_{n-1}\}$, where a single document is identified by $d_i 0 \leqslant i < n$ and contains a collection of terms $d_i = \{t_0, \ldots, t_{|d_i|-1}\}$, where $|d_i|$ indicates its size. Finally, the value of *TF/IDF* for a term contained in a document within a collection of documents was given by

$$\mathrm{TF/IDF}(d_i, t_j, D) = \mathrm{TF}(t_j, d) \cdot \log_2 IDF(t_j, D) = \frac{|t_j|}{\max\{TF(t_0, d_i), \ldots, \mathrm{TF}(t_{|d_i|-1}, d_i)\}} \cdot \log_2 \frac{|D|}{|D \supset t_j|}, \tag{1}$$

where $|t_j|$ is the number of times that the term t_j occurs in the document d_i (which is normalised using the maximum term frequency found in d_i) and $|D \supset t_j|$ indicates in how many documents t_j appears.

2.3. Machine learning algorithms

There is a wide variety of machine learning algorithms for data classification (Aggarwal & Zhai, 2012) that can be applied to text classification . However, it is important to take into account that words in text documents represent data attributes that are generally quite sparse and high dimensional, due to the combination of usually large dictionary size and the low frequencies of most of its words.

We selected a range of classification algorithms based on the diversity of the underlying machine learning methods that support them. The types of algorithms considered included linear, probabilistic, example-based, and profile-based. The following sections provide a brief description of them.

2.3.1. Naïve bayes

Naïve bayes is a probabilistic classification algorithm based on the assumption that any two terms from $T = \{t_1, \ldots, t_{|T|}\}$ representing a document d and classified under category c are statistically independent of each other (Lewis, 1998). This can be expressed by

$$P(d|c) = \prod_{i=1}^{|T|} P(t_i|c) \tag{2}$$

The category predicted for d is based on the highest probability given by

$$\mathrm{argmax} P(c|d) = \mathrm{argmax} P(c)\ P(d|c) = \mathrm{argmax} P(c) \prod_{i=1}^{|T|} P(t_i|c) \tag{3}$$

Two commonly used probabilistic models for text classification under the naïve bayes framework are the multi-variate Bernoulli and the multinomial models. These two approaches were compared in McCallum and Nigam (1998), with the multinomial model proving to perform significantly better than the multi-variate Bernoulli, hence motivating us to choose the first for this work.

2.3.2. k-Nearest neighbours

k-Nearest Neighbours (kNN) is an example-based classification algorithm (Yang & Chute, 1994) where an unseen document is classified with the category of the majority of the k most similar training documents. The similarity between two documents can be measured by the Euclidean distance of the n corresponding feature vectors representing the documents

$$s(d_i, d_j) = \sum_{f=1}^{n} (d_{i_f} - d_{j_f})^2. \tag{4}$$

All neighbours can be treated either equally or with an assigned weight that corresponds to their distance to the document being categorised. We selected two weighting methods: inverse to the distance ($1/s$) and opposite to the distance ($1 - s$). In cases where several of these k nearest neighbours are found to belong to the same category, their weights are added together, so that the final weighted sum is used as the probability score for that category. Sorting them by rank yields a list of categories where to assign the document.

Building a kNN categoriser also requires experimentally determining a threshold k, obtaining good results with $30 \leqslant k \leqslant 45$ (Yang & Liu, 1999). It is also interesting to note that increasing the value of k does not degrade the performance significantly.

2.3.3. Support vector machines

Support vector machines (SVM) are a group of supervised learning methods that can be applied to either classification or regression. It has become a popular algorithm in the last years due to its good performance, with reports of possibly being the current most accurate technique for text classification (Liu, 2011). It is also important to note that in certain situations (e.g., large set of support vectors), using SVM can be significantly expensive computationally speaking (Burges, 1998).

Its foundation is the Structural Risk Minimisation principle (Vapnik, 1995) from computational learning theory, which searches for a hypothesis with the lowest error. More specifically, given a set of two-class instances to classify, a SVM finds the unique hyperplane in a high dimensional space that separates the positive and negative instances with maximum margin (i.e., minimum error). Those instances defining an hyperplane are known as support vectors. In consequence, the answer to a classification problem is given by the support vectors that determine the maximum margin hyperplane. This is known as a linear classifier. In those cases where classes cannot be separated by a linear classifier, the features of the instances are mapped into a feature space using nonlinear functions called feature functions. The nonlinear mapping induced by the feature functions is computed with special nonlinear functions called kernels (Joachims, 2002). In this case, the solution is defined as a weighted sum of the values of certain kernel function evaluated at the support vectors.

In other words, given a set of training documents $D = \{d_0, d_1, \ldots, d_n\}$, where d_i represents the feature vector for a document categorised as $c_i \in \{1, -1\}^n$ with $0 \leqslant i < n$, the support vector machines are based on solving the optimisation problem expressed by

$$\max_{\alpha_i} \quad \sum_{i=0}^{n-1} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j c_i c_j K(d_i, d_j), \tag{5}$$

with $\alpha_i \geqslant 0$ and K representing the kernel function. Although new kernels are being proposed by the research community, the most established instances include the linear kernel where $K(d_i, d_j) = d_i \cdot d_j$, the polynomial kernel where $K(d_i, d_j) = [d_i \cdot d_j + 1]^{\beta}$, the Gaussian radial basis function kernel where $K(d_i, d_j) = \exp(-\gamma \|d_i - d_j\|^2)$ for $\gamma > 0$, and the sigmoid kernel where $K(d_i, d_j) = \tanh(\kappa d_i \cdot d_j + c)$ with $\kappa > 0$ and $c < 0$.

2.3.4. Rocchio

Rocchio is a profile-based classification algorithm (Moschitti, 2003) adapted from the classical Vector Space Model with *TF/IDF* weighting and relevance feedback to the classification process. This

type of classifier uses a similarity measure between a representation (also called profile) p_i of each category c_i and the unseen document d_j to classify. This similarity is usually estimated as the cosine angle between the vector that represents c_i and the feature vector obtained from d_j. Therefore, a document to classify is considered to belong to a particular category when its related similarity estimation is greater than a certain threshold.

Rocchio needs a feature frequency function to be defined, such as

$$s(f,d) = \frac{r(f,d)\log(|D|/n_f)}{\sum_{i\in F} r(i,d)\log(|D|/n_i)}, \tag{6}$$

where F is the set of all existing features with $f \in F$, n_i expresses in how many documents f_i appears, and r is the function of relative relevance of multiple occurrences that can be defined by $r(f,d) = \max(0, \log(0, n_f))$.

The profile p_i of a category c_i is a vector of weights where one instance is calculated by

$$w(f,d) = \max\left(0, \frac{\beta}{|D_c|}\sum_{d\in D_c} s(f,d) - \frac{\gamma}{|\overline{D}_c|}\sum_{d\in \overline{D}_c} s(f,d)\right), \tag{7}$$

where D_c is the set of documents belonging to c and $\overline{D}_c$ the set of documents not belonging to c. The parameters β and γ control the relative impact of these positive and negative instances to the vector of weights, with standard values being $\beta = 16$ and $\gamma = 4$ (Moschitti, 2003).

2.4. Evaluation measures

Precision (π) and *recall* (ρ) are two common measures for assessing how successful a text categoriser is. Precision indicates the probability that a document assigned to a certain category by the classifier actually belongs to that category. On the contrary, recall estimates the probability that a document that actually belongs to a certain category will be correctly assigned to that category during the categorisation process. These two measures are defined by

$$\pi = \frac{TP_i}{TP_i + FP_i}, \quad \rho = \frac{TP_i}{TP_i + FN_i}, \tag{8}$$

where TP_i indicates the number of *true positives* or how many documents were correctly classified under category c_i. Similarly, FP_i indicates the number of *false positives* and FN_i corresponds to *false negatives*. Table 1 provides an overview of these measures.

Precision and recall are generally combined into a single measure called F_β, with $0 \leqslant \beta \leqslant \infty$. The parameter β, which is used to find the appropriate balance between the importance of π and ρ, is expressed by

$$F_\beta = \frac{(\beta^2+1)\pi\rho}{\beta^2\pi + \rho}. \tag{9}$$

Values close to 0 give more importance to π while those closer to ∞ provide more relevance to ρ. The most common applied value is 1, which procures the same importance for both π and ρ. Therefore, Eq. (9) is transformed into

$$F_1 = \frac{2\pi\rho}{\pi+\rho} = \frac{2TP_i}{2TP_i + FP_i + FN_i}. \tag{10}$$

Instead of using category-specific values of F_1, an averaged measure is usually preferred, concretely the macro- and micro-average, identified by F_1^M and F_1^μ respectively. Micro-averaging gives more emphasis on performance of frequent categories (i.e., there are more training documents for these categories) and is defined by

$$F_1^\mu = \frac{\sum_{i=1}^{|C|} 2TP_i}{\sum_{i=1}^{|C|}(2TP_i + FP_i + FN_i)}, \tag{11}$$

where $|C|$ indicates the number of categories.

By contrast, macro-averaging focuses on uncommon categories. Micro-averaged measures will almost always have higher scores than the macro-averaged ones. This can be expressed by

$$F_1^M = \frac{\sum_{i=1}^{|C|} F_{1_i}}{|C|}. \tag{12}$$

Finally, the overall *error* is measured through

$$error = \frac{\sum_{i=1}^{|C|}(FP_i + FN_i)}{\sum_{i=1}^{|C|}(TP_i + FP_i + FN_i + TN_i)}. \tag{13}$$

2.5. Cross validation

Cross validation consists of partitioning a sample of data into subsamples such that analysis is initially performed on a single subsample, while further subsamples are retained in order for subsequent use in confirming and validating the initial analysis. Cross validation is a concept borrowed from statistics by text classification where it is used to evaluate a categoriser by applying the measures described in Section 2.4. A learning algorithm is trained with a percentage of all the existing training set. When the training is finished, the data that was not used for training is then used to test the performance of the trained algorithm.

There are different types of cross validation, with the n-fold method (also called rotation estimation) possibly the most common (Kohavi, 1995). It consists of dividing the complete collection of documents D into n mutually exclusive subsets called folds D_0, $D_1, \ldots, D_{n-1}$. Each fold should have the same number of documents except for the last $n-1$ subset that may end having more documents than the other subsets. Then, for each partition, one of the n subsets D_t is used as the test set while the rest of the subsets $D \backslash D_t$ are used to create a training set. The accuracy estimation using cross validation corresponds to the number of correct categorisations divided by the total number of documents in the collection as expressed by

$$acc_{cv} = \frac{\sum_{\langle d_j, c_i\rangle \in D_k} \delta(C(D \setminus D_j, d_j), c_i)}{n} \tag{14}$$

with D_j being the test fold that incorporates the document d_j with category c_i, $\delta(i,j) = 1$ if $i = j$ or 0 otherwise, and C is the categorisation function that returns a category.

Table 1
Category-specific contingency table of classification. It shows the expert decision about a document belonging to a category, in relation to the classifier prediction.

Expert decision	Classifier prediction	Result
Yes	Yes	*TP*
Yes	No	*FN*
No	Yes	*FP*
No	No	*TN*

3. Medical systematic reviews

A systematic review consists of synthesising the relevant published literature representing the high-quality research evidence that answers a specific research question (Sackett, Rosenberg, Gray, Haynes, & Richardson, 1996). Systematic reviews provide the foundation for Evidence-based Medicine (Greenhalgh, 2010), which is based on the premise that medical knowledge, based on

the accumulation of results from multiple scientific studies, is more reliable than expert opinion. Although evidence-based research and practice originated in medicine, they are currently used in other areas including nursing, economics, software engineering (Kitchenham et al., 2009), and social sciences (Petticrew & Roberts, 2006).

Fig. 1 depicts the general process of producing a systematic review. It starts by proposing a health-related question, which is followed by the pertinent retrieval of candidate scientific studies from the literature. The corresponding search task should attempt to cover all the existing literature without bias (Dubben & Beck-Bornholdt, 2005) through a number of queries and filters that may include a combination of controlled vocabulary and natural language expressions.

Once all potential studies have been retrieved, the screening phase consists of two or more reviewers reading each abstract in order to determine its eligibility for a full-text review (The Cochrane Collaboration, 2011). This screening process is typically performed over collections of 2000–5000 bibliographic references, and its main goal is to include all relevant articles (University of York, 2008). At the end of the screening process, the number of relevant articles detected would typically be around 200–400. Based on our own experience, the time required for one person to proceed with screening 10,000 instances is about 500 hours (i.e., 20 instances per hour). In other words, this is a very time-consuming task.

The last step consists of retrieving the full text of these relevant articles in order to systematically evaluate each one of them, eventually selecting about 10–50 of those that are both pertinent and with a high standard of quality.

It is during the screening where automatic text classification can be useful as Fig. 1 illustrates. We identify two possible approaches to take advantage of the text classifier: (i) a *conservative* one where the classifier is used as a reassurance tool by reviewers, who would pay especial attention to those articles in disagreement, and (ii) an *aggressive* approach where one or more reviewers are 'replaced' by the text classifier.

4. Related work

One of the first attempts of testing a similar hypothesis was reported by Aphinyanaphongs, Tsamardinos, Statnikov, and Hardin (2005). They applied naïve bayes and SVM text classifiers to a corpus of internal medicine articles from the ACP Journal Club to discover that SVM offered the best performance in terms of sensitivity, specificity, and precision.

More recently, Wallace, Trikalinos, Lau, Brodley, and Schmid (2010) used a SVM classifier to three different collections of article abstracts plus titles, determining that the system was useful to reduce the number of citations to manually screen by almost 50%. The authors proposed a semi-automatic 'aggressive' approach with reviewers trusting the text classifier by not screening those articles it labelled as irrelevant.

Frunza et al. (2011) applied a naïve bayes classifier to a collection of 47,274 article abstracts previously manually labelled, using 20,000 abstracts for training and the rest for testing. They obtained results that included very high recall values (up to 99%) at the expense of a moderate precision of 63%.

The most recent work (Bekhuis & Demner-Fushman, 2012) is also the most complete so far. It consisted of applying *k*NN, naïve bayes, and SVM classifiers, combined with Information Gain as their method for feature selection, to either article titles or article title plus abstracts. The experiments also showed positive results where the initial set of documents to manually screen was reduced by up to 46%.

The conclusion after looking at these previous instances is that there is a disparity about the type of classifiers being used and the text content selected from articles. Also, they did not analyse the

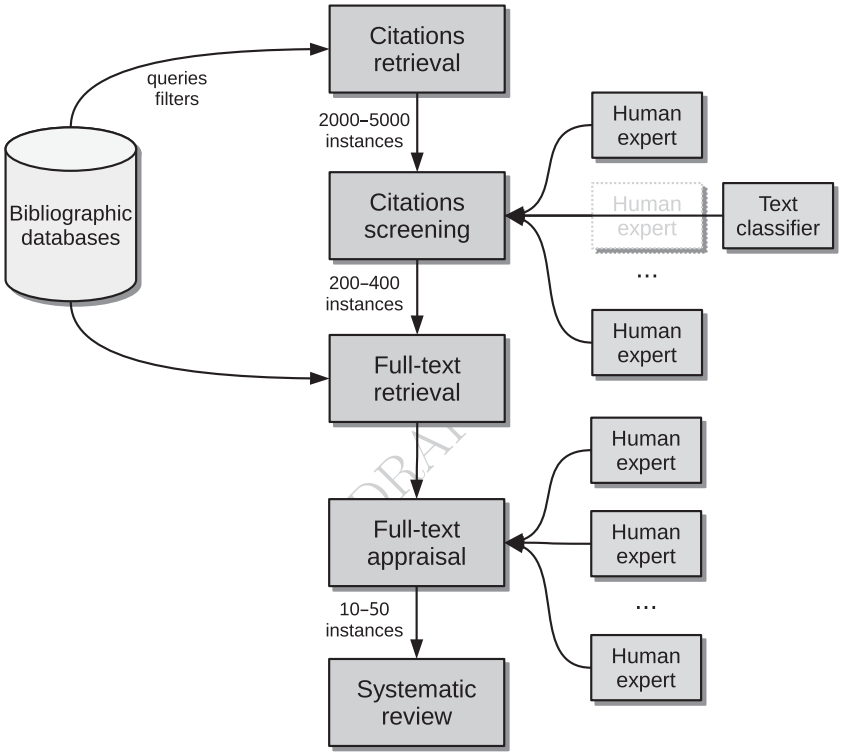


Fig. 1. Overview of the traditional process to produce a systematic review modified by the inclusion of automatic text classification to the citation screening phase. The text classifier can either replace or complement the input by a human expert depending on the acceptable level of risk.

impact of different types of feature selection methods and number of selected features (i.e., dimensionality of the VSM) as part of the classification process.

5. Validation

We considered appropriate to empirically evaluate our hypothesis through a validation exercise. It is important to differentiate verification and validation. The first consists of merely confirming that the implemented system works sensibly and fulfils expectations, while the latter requires an accurate analysis of experimental results with in relation to an existing set of actual results (Mihram, 1972).

For this purpose, Section 5.1 provides details on the characteristics of the collection of articles under study. Section 5.2 describes the selected configurations for machine learning algorithms and related methods such as feature selection. The implementation of these experiments is briefly described in Section 5.3, before Section 5.4 provides and discusses the results.

5.1. Corpus

The corpus selected for this work is called Internet-Based Randomised Control Trial (IBRCT) mapping. It was created with the purpose of identifying those Randomised Controlled Trials (RCT) that used the Internet as an intrinsic component to their clinical trial process, including the phases such as design, information retrieval, statistical analysis, or reporting regardless of whether the article describing the study explicitly established this fact.

In order to be included as relevant, a study had to fulfil the following criteria: (i) it had to be a RCT in the broad area of public health, including educational and behavioural applications; (ii) it used Internet, including the Web, to support its trial process; and (iii) it might utilise mobile Internet as a supplementary technology. Studies falling within the following categories were considered irrelevant: (i) they only used mobile technology with no Internet access; and (ii) they deal with social or educational care but not from a purely health research focus.

The corpus was prepared by querying multiple medical data bases through very specific search approaches. The latter was based on search expressions that included synonyms, natural language, and keywords in order to identify the appropriate concepts to find. Moreover, these queries were designed in order to favour high recall (see Section 2.4 for more details) at the expense of significant noise (i.e., low precision as defined also in Section 2.4). The data bases selected were Medline, Embase, Cinahl, Central, LILACS, PsycInfo, ERIC, Pedro, OT Seeker, in addition to general Web search engines such as Google, Yahoo, Copernic, and manual searches in the journals Journal of the American Medical Informatics Association (JAMIA) and Journal of Medical Internet Research (JMIR). Each of the retrieved articles were completely read and evaluated by the committee of experts in order to determine its suitability.

The resulting corpus consists of 1941 articles divided into 510 relevant and 1431 irrelevant instances. Each article includes its title, author list, journal, keywords, and abstract. The total number of terms is 631,435 terms in total, with 61,752 of them unique. This corresponds to an average of 325 per article, where the shortest article has 48 terms and the longest 1121.

5.2. Experiments

Three types of documents were prepared based on what text was selected from the citations: (i) only titles, (ii) only abstracts, or (iii) titles plus abstracts.

These documents were selected in an arbitrary order to be preprocessed by following the steps covered by Section 2.1. We performed a preliminary removal of both numeric characters and stop-words to the articles. Afterwards, we applied a Porter stemmer (Porter, 1980) to the remaining terms in order to remove the most common morphological and inflectional endings. In consequence, the initial dictionary was reduced by 40%, thus leaving 380,384 terms, with 16,580 of them being unique. This preprocessing configuration was chosen as the experimental baseline after several having performed preliminary trials that included combining term n-grams (Tan, Wang, & Lee, 2002), no stemming, and no stop-words with inferior results.

The machine learning algorithms selected were the 4 different instances explained in Section 2.3: Naïve bayes, *k*NN, SVM, and Rocchio. We parametrised *k*NN by chosing the value $k = 35$, which is known to perform well (Yang & Liu, 1999). For SVM, a linear kernel was preferred due to its good balance between execution time and accuracy of results. In the case of the Rocchio classifier, it was configured with the standard parameter values $\alpha = 0.25$, $\beta = 16$, $\gamma = 4$, and a threshold of 0.7 (Moschitti, 2003).

Feature selection was performed using the 7 different methods explained in Section 2.2: TF, DF, IDF, χ^2, local TF, local DF, and local IDF. The number of features to build the vectors representing the articles (i.e., the number of dimensions for the VSM) included: 5, 10, 25, 50, 100, 250, 500, 1000, 2000, 4000, 8000, and 15000. Feature vectors were built using the function *TF*|*IDF* as described in Section 2.2 by Eq. (1).

The set of documents were divided into training and testing subsets. Because we used cross validation (see Section 2.5) with 10 folds, the number of documents for training was 1747 while leaving 194 documents for testing in each of the 10 iterations.

Table 2 offers a summary of all the experimental combinations of machine learning algorithms, feature selection methods, and number of features. Collectively, they amount to a total number of 336 classification processes for each cross-validation fold, thus increasing the number of these to 3360 for each of the three document configuration. In consequence, the total number of classification processes was 10,080.

5.3. Implementation

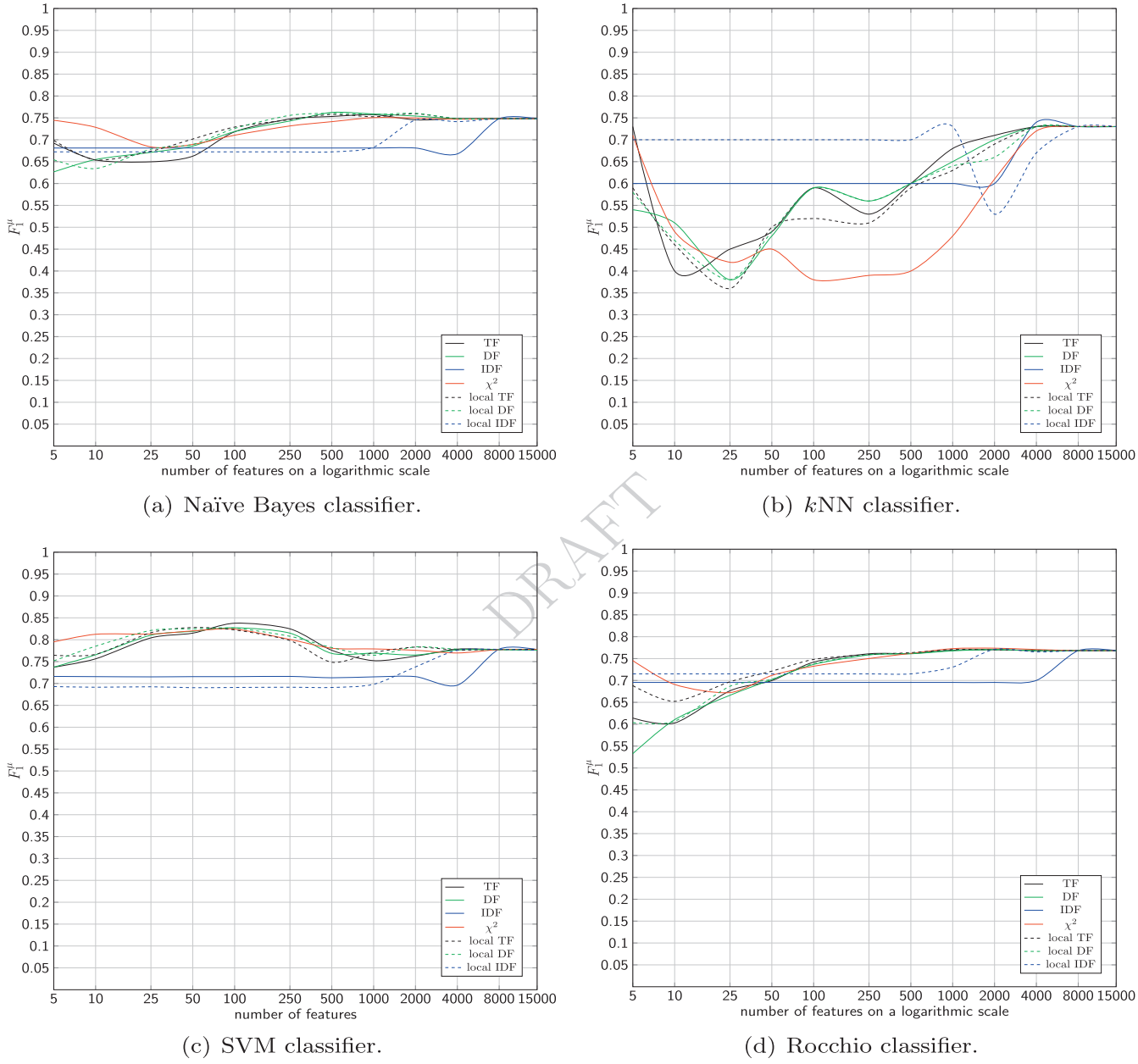
These experiments were implemented as a text-mining application based on the general text-mining application framework called Pimiento (García Adeva & Calvo, 2006). This software was written using Java Standard Edition (J2SE) and aimed at providing developers with the primary benefits of OOAF, such as modularity, reusability, extensibility, and inversion of control (Fayad & Schmidt, 1997).

Pimiento offers numerous features to suit both a production environment where performance is crucial and a research context in which highly configurable experiments must be executed. It can be used in production systems due to its high scalability based on a cache system that allows for precise control of the amount of memory allocated, and its performance efficiency thanks to a carefully tuned-up code-base. These features are offered as a collection of software components that cover all the functionalities that it tackles including categorisation, language identification, clustering, summarisation, and similarity analysis.

Because the text classification functionality offered by Pimiento offers the learning algorithms described in Section 2.3, we did not have to implement any new algorithm. This functionality supports preprocessing of documents in English, German, French, Spanish, and Basque. However, only the English preprocessor was needed for this work. There is also complete evaluation of results using the evaluation measures described in Section 2.4 such as the category-specific measures TP_i, FP_i, FN_i, π_i, ρ_i, F_{1_i} and the averaged measures π^μ, ρ^μ, F_1^μ, π^M, ρ^M, F_1^M, as well as partitioning of the testing space using *n*-fold cross-validation as explained in Section 2.5.

Table 2
Summary of classification experiments.

Classification algorithms	Feature selection methods	Number of features	Number of documents	Document sections
naïve bayes , *k*NN , SVM, Rocchio	TF, DF, IDF, χ^2, local TF, local DF, local IDF	5, 10, 25, 50, 100, 250, 500, 1000, 2000, 4000, 8000, 15000	1747 Training, 194 testing, 10-fold cross validation	Title, abstract, title and abstract



(a) Naïve Bayes classifier.

(b) kNN classifier.

(c) SVM classifier.

(d) Rocchio classifier.

Fig. 2. F_1^μ scores of classifying the articles based on their titles.

The experiments were executed in a computer server based on a 4-core Intel Xeon processor, 16 GB of RAM and the Linux operating system. Execution time varied significantly among classifiers, depending mostly on the machine learning algorithm in question. Naïve bayes and Rocchio were always the fastest, taking a few seconds to complete for a 10-fold cross validation task. For the same work *k*NN took about 40 min, while SVM was unpredictable due to its stochastic nature, but ranging from a few minutes to several hours. It is interesting to note that the number of selected features only contributed marginally to the increase in execution time.

5.4. Results

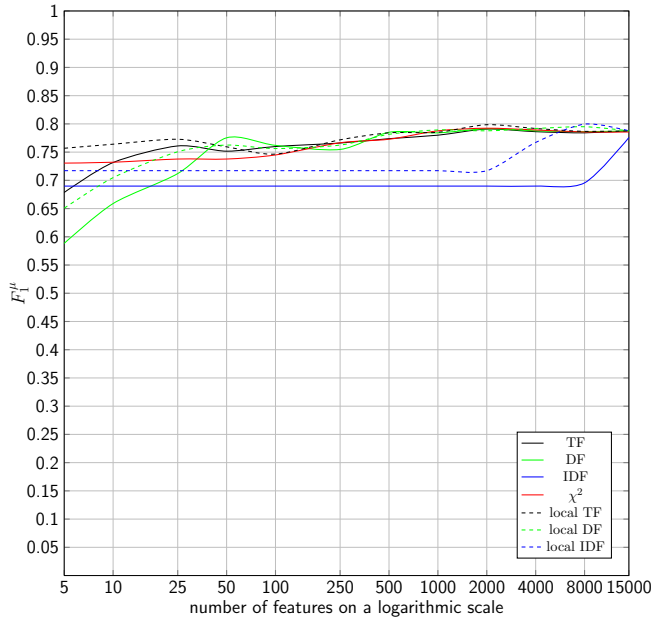
We chose the micro-averaged measure F_1^μ, explained in Section 2.4, in order to provide a general measure of performance by the classifiers.

Three sets of results were prepared, based on what parts of an article were selected to form the documents used by the different classifiers, as described in previous Section 5.2. Thus, Fig. 2 provides an overview of F_1^μ values for each of the 4 classifiers combined with the 7 feature selection methods over a number of features when only selecting the titles of the articles. Likewise, Fig. 3 does so when only the abstracts were taken into account, while Fig. 4 corresponds to both titles and abstracts. Each data point in these graphs corresponds to the F_1^μ value for the average of the 10 classification tasks as produced by the 10-fold cross validation process.

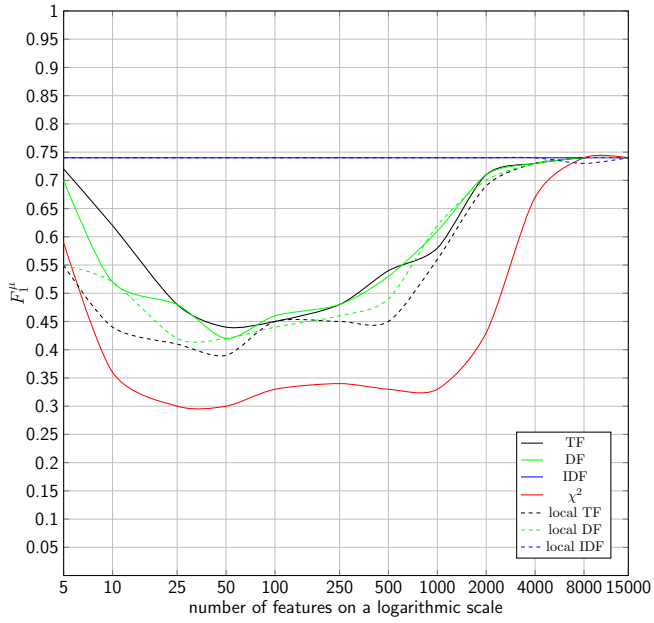
When using only titles, Fig. 2(c) shows that SVM was the classifier with better performance in terms of F_1^μ, with values frequently between 0.8 and 0.85. Naïve bayes and Rocchio, represented by Fig. 2(a) and (d) respectively, were both quite similar while slightly inferior to SVM. Fig. 2b made it clear that *k*NN was the worst performing classifier, which in some cases provided significant poor F_1^μ values — even below 0.4.

SVM also performed better than the rest when both abstracts and titles plus abstracts were selected from the articles to classify, as showed by Fig. 2 and Fig. 4(c) respectively.

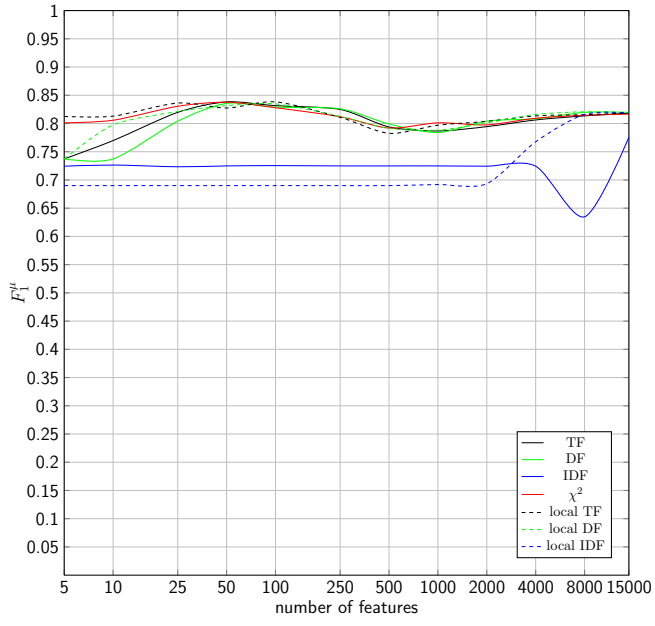
Both naïve bayes and Rocchio performed slightly better when using only abstracts (Fig. 3(a) and (d) respectively) instead of only titles (Fig. 2(a) and 3(a) respectively). However, for these two classification algorithms there was no remarkable difference when using only abstracts as opposed to using abstracts plus titles, as it can be observed by comparing Fig. 3(a) to Fig. 4(a) for naïve bayes and Fig. 3(a) to Fig. 4(a) for Rocchio.



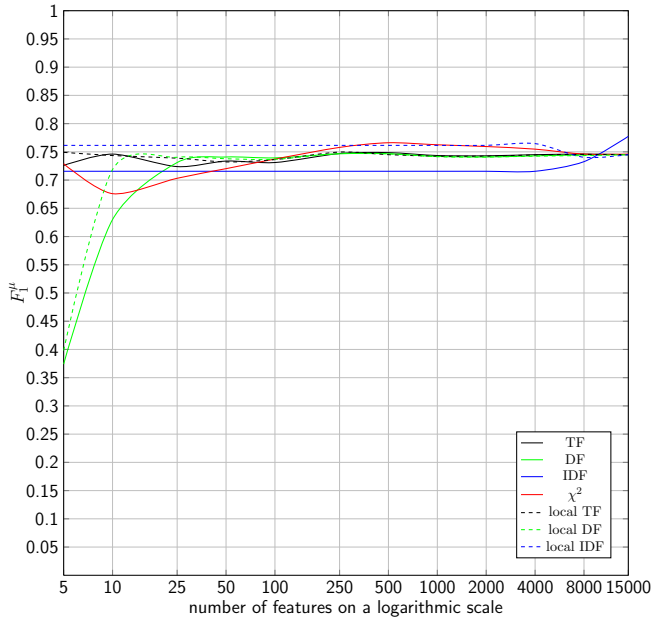
(a) Naïve Bayes classifier.



(b) *k*NN classifier.

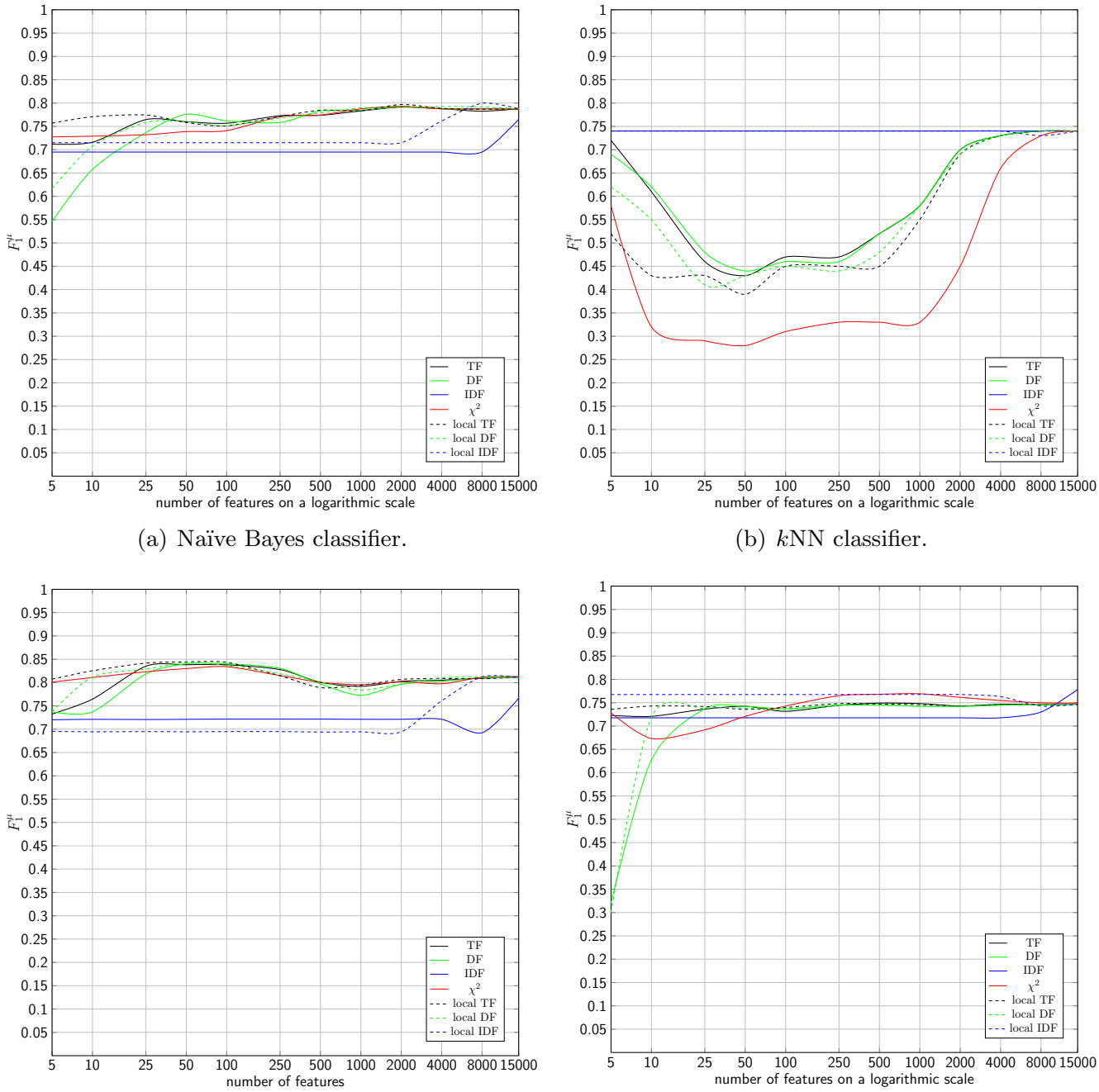


(c) SVM classifier.



(d) Rocchio classifier.

Fig. 3. F_1^μ scores of classifying the articles based on their abstracts.



(a) Naïve Bayes classifier.

(b) kNN classifier.

(c) SVM classifier.

(d) Rocchio classifier.

Fig. 4. F_1^μ scores of classifying the articles based on their titles and abstracts.

By contrast, using SVM did not seem to make a major difference regardless of the article contents selected, except for an insignificant difference in the averaged *error* that can probably be ignored. It only made some difference in the number of features required, as could be observed by comparing Figs. 2(c), 3(c), and 4(c). In other words, when only titles were selected SVM required a larger number of features to perform as well as when more article content was selected with fewer features. An explanation for this is that the more the contents selected from articles, the higher the diversity of features.

In general, SVM clearly showed superiority over the rest of classifiers, not only in classification performance but in the number of required features to perform well. This conforms with established research that describes how SVM tends to work well with few dimensions (Sindhwani & Keerthi, 2006). This situation might indicate that the number of 'quality' features in the corpus was small, based on the size of documents and the number of them. Another outcome is that the selection feature methods did not provide a dramatic difference in performance. It might seem as if SVM worked better when using term frequencies (either global or local), while the other algorithms preferred some type of document frequency (either direct, inverse, global, or local).

Table 3 provides a detailed account about the classifier configuration that produced the best results for each combination of classification algorithm and article type. By looking at these measurements, a positive outcome was that the values of F_1^μ were

Table 3
Best results by classification algorithm, feature selection method, number of features, and article type.

Article	Classifier	Category-specific measures								Averaged measures						
		Category	*TP*	*FP*	*FN*	*TN*	ρ	π	F_1	ρ^M	π^M	F_1^M	ρ^μ	π^μ	F_1^μ	Error
Title	Naïve bayes	Relevant	43	38	7	104	0.86	0.53	0.65	0.8	0.73	0.74	0.76	0.76	0.76	0.18
	DF, 500	Irrelevant	104	7	38	43	0.73	0.94	0.82							
	Rocchio	Relevant	13	6	37	136	0.27	0.68	0.38	0.61	0.73	0.62	0.77	0.77	0.77	0.16
	χ^2, 2000	Irrelevant	136	37	6	13	0.96	0.79	0.86							
	*k*NN	Relevant	3	5	47	138	0.06	0.39	0.11	0.51	0.57	0.47	0.73	0.73	0.73	0.18
	TF, 5	Irrelevant	138	47	5	3	0.96	0.74	0.84							
	SVM	Relevant	31	12	19	130	0.63	0.72	0.67	0.5	0.8	0.78	0.84	0.84	0.84	0.12
	TF, 100	Irrelevant	130	19	12	31	0.91	0.87	0.89							
Abstract	Naïve bayes	Relevant	43	31	7	111	0.86	0.58	0.69	0.82	0.76	0.77	0.8	0.8	0.8	0.15
	Local IDF, 8000	Irrelevant	111	7	31	43	0.78	0.94	0.85							
	Rocchio	Relevant	15	7	35	135	0.3	0.67	0.41	0.62	0.73	0.64	0.78	0.78	0.78	0.15
	IDF, 15000	Irrelevant	135	35	7	15	0.95	0.79	0.86							
	*k*NN	Relevant	2	5	49	137	0.04	0.31	0.07	0.5	0.52	0.45	0.72	0.72	0.72	0.19
	TF, 50	Irrelevant	137	49	5	2	0.96	0.74	0.84							
	SVM	Relevant	34	15	16	128	0.69	0.7	0.69	0.79	0.79	0.79	0.84	0.84	0.84	0.12
	TF, 10	Irrelevant	128	16	15	34	0.89	0.89	0.89							
Title and abstract	Naïve bayes	Relevant	43	31	7	111	0.86	0.58	0.69	0.82	0.76	0.77	0.8	0.8	0.8	0.15
	Local IDF, 8000	Irrelevant	111	7	31	43	0.78	0.94	0.853							
	Rocchio	Relevant	17	9	33	134	0.33	0.64	0.44	0.63	0.72	0.65	0.78	0.78	0.78	0.15
	IDF, 15000	Irrelevant	134	33	9	17	0.94	0.8	0.86							
	*k*NN	Relevant	2	6	48	136	0.05	0.31	0.08	0.5	0.52	0.46	0.72	0.72	0.72	0.19
	TF, 5	Irrelevant	136	48	6	2	0.96	74	0.83							
	SVM	Relevant	33	12	17	130	0.7	0.72	0.7	0.79	0.8	0.79	0.84	0.84	0.84	0.11
	Local TF, 50	Irrelevant	130	17	12	33	0.91	0.88	0.9							

produced by a generally well-balanced combination of precision and recall values, both when looking at category-specific measures (i.e., π and ρ) or averaged measures (i.e., π^μ, ρ^μ, π^M, and ρ^M).

Within the context of how the classifier was intended to be useful, it was sensible to assume that one of the key measures to pay attention to was the number of false negative predictions of relevant articles, which corresponded to the measure *FN* for category *relevant* (i.e., *FP* for category *irrelevant*). The reason is that these mistakes corresponded to relevant articles classified as irrelevant. This is especially important if the automatic classification was used *aggressively* as described in Section 3. If this is the case, the number of citations to be screened manually would be reduced by up to 57% in the case of naïve bayes and 77% for SVM, at the cost of missing relevant articles. In other words, irrespective of overall measures such as F_1^μ, π^μ, or ρ^μ, naïve bayes seemed to provide the best results in terms of false negatives, with only 7 articles mistakenly identified as such.

By contrast, other mistakes such as false positives were less important because they did not exclude relevant articles — instead, they merely increased the burden by including irrelevant articles as relevant. These are probably important aspects to consider carefully when applying a text classifier to support decision-making.

6. Conclusions and future work

We empirically evaluated the application of automatic text classification to the process of medical systematic reviews, in order to facilitate the manual process carried out by experts during the citation screening phase. The experiments involved multiple classification algorithms combined with several feature selection methods, and number of features, applied to different parts of the given articles. The analysis of these experiments showed overall positive results, especially when using the algorithms naïve bayes and SVM. Naïve bayes offered the lowest rate of mistakes in the form of *FN* for any type of article, whereas SVM performed as well when using only titles as then appending abstracts. The discussion of the results explained how the selected method for feature selection can make significant difference, contrary to common practice of ignoring this aspect of the text classification process. In this regard, the experiments also provided an interesting insight into how a reduced number of features can produce the best results. In summary, the presented results indicated that including automatic text classification to the manual screening process when performing systematic reviews, can substantially reduce the number of articles reviewers have to analyse.

Regarding future work, there are two aspects that could be explored further. One is attempting to improve classification performance by applying ensembles of classifiers (Valentini & Masulli, 2002). Another would consist of performing similar experiments with other collections of articles within other medical domains, in order to contrast how useful automatic text classification to the decision-making process.

Acknowledgements

The corpus used in this work was provided by Anne Brice from the Critical Appraisal Skills Programme in Oxford, UK.

This work was supported by funding received from the Department of Education, Universities and Research of the Basque Government (Grant No. BFI-09-270), the UPV/EHU [GIU08/27, INF10/58, GIU11/28 and UFI11/19], Gipuzkoa Regional Council [OF53/2011], the Department of Industry, Commerce and Tourism — Basque Government [S-PE09UN60 and S-PE11UN115], and the Spanish Ministry of Science and Innovation [TIN2009-14 159-C05-03].

References

Aggarwal, Charu C., & Zhai, ChengXiang (2012). A survey of text classification algorithms. In *Mining text data* (pp. 163–222).

Aphinyanaphongs, Yindalon, Tsamardinos, Ioannis, Statnikov, Alexander R., Hardin, Douglas P., et al. (2005). Text categorization models for high-quality retrieval in internal medicine. *JAMIA, 12*(2), 207–216.

Baeza-Yates, Ricardo, & Ribeiro-Neto, Berthier (1999). *Modern information retrieval*. Wokingham, UK: Addison-Wesley.

Bastian, Hilda, Glasziou, Paul, & Chalmers, Iain (2010). Seventy-five trials and eleven systematic reviews a day: How will we ever keep up? *PLoS Med* 7(9).

Bekhuis, Tanja, & Demner-Fushman, Dina (2012). Screening nonrandomized studies for medical systematic reviews: A comparative study of classifiers. *Artificial Intelligence in Medicine, 55*(3), 197–207.

Burges, Christopher J. C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery, 2*(2), 121–167.

Dubben, Hans-Hermann, & Beck-Bornholdt, Hans-Peter (2005). Systematic review of publication bias in studies on publication bias. *BMJ, 331*(7514), 433–434. http://dx.doi.org/10.1136/bmj.38478.497164.F7.

Fayad, Mohamed, & Schmidt, Douglas C. (1997). Object-oriented application frameworks. *Communications of the ACM, 40*(10), 32–38.

Frunza, Oana, Inkpen, Diana, Matwin, Stan, Klement, William, Peter O'Blenis (2011). Exploiting the systematic review protocol for classification of medical abstracts. *Artificial Intelligence in Medicine*, 51(1), 17–25, January 2011.

García Adeva, J. J., & Calvo, R. (2006). Mining text with pimiento. *IEEE Internet Computing, 10*(4), 27–35.

Greenhalgh, T. (2010). *How to read a paper: The basics of evidence-based medicine*. HOW – How To. Wiley. ISBN 9781444323184.

Hearst, Marti A. (1999). Untangling text data mining. In *Proceedings of the 37th conference on association for computational linguistics, College Park, Maryland* (pp. 3–10). Association for Computational Linguistics.

Ide, N. C., Loane, R. F., & Demner-Fushman, D. (2007). Essie: A concept-based search engine for structured biomedical text. *Journal of the American Medical Informatics Association, 14*(3), 253–263.

Iruetaguena, A., Garcíá Adeva, J. J., Pikatza, J. M., Segundo, U., Buenestado, D., & Barrena, R. (2013). Automatic retrieval of current evidence to support update of bibliography in clinical guidelines. *Expert Systems with Applications, 40*(6), 2081–2091. http://dx.doi.org/10.1016/j.eswa.2012.10.015.

Joachims, T. (2002). *Learning to classify text using support vector machines – Methods, theory, and algorithms*. Springer: Kluwer.

Kitchenham, B., Brereton, O. P., Budgen, D., Turner, M., Bailey, J., & Linkman, S. (2009). Systematic literature reviews in software engineering — a systematic literature review. *Information and Software Technology*, 51 (1), 7–15. ISSN 0950-5849.

Kohavi, Ron (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *IJCAI* (pp. 1137–1145).

Lewis, David D. (1998). Naive (bayes) at forty: The independence assumption in information retrieval. In *Claire Nédellec and Céline Rouveirol. Proceedings of ECML-98, 10th European Conference on Machine Learning, Chemnitz, DE* (pp. 4–15). Heidelberg, DE: SpringerVerlag.

Lind, James (1753). A treatise of the scurvy. In three parts. Containing an inquiry into the nature, causes and cure, of that disease. Together with a critical and chronological view of what has been published on the subject. Sands, Edinburgh.

Liu, B. (2011). *Web data mining: Exploring hyperlinks, contents, and usage data. Data-Centric Systems and Applications*. Springer. ISBN 9783642194597.

Lovins, B. Julie (1968). Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics,* 11, 22–31.

McCallum, Andrew & Nigam, Kamal (1998). A comparison of event models for naive bayes text classification. In *Proceedings of AAAI-98 workshop on learning for text categorization* (pp. 137–142). Madison, Wisconsin

Mihram, G. A. (1972). Some practical aspects of the verification and validation of simulation models. *Operational Research Quarterly, 23*(1).

Moschitti, A. (2003). A study on optimal parameter tuning for Rocchio Text Classifier. In *Proceedings of the 25th European conference on information retrieval research*.

Petticrew, Mark, & Roberts, Helen (2006). *Systematic reviews in the social sciences: A practical guide*. Blackwell Publishing.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program, 14*(3), 130–137.

Sackett, David L., Rosenberg, William M. C., Gray, J. A. Muir, Haynes, R. Brian, & Richardson, W. Scott (1996). Evidence based medicine: what it is and what it isn't. BMJ, 312 (7023): 71–72.

Salton, Gerard (1989). *Automatic text processing: The transformation, analysis, and retrieval of information by computer*. Addison-Wesley, Reading, Pennsylvania.

Salton, Gerard, & Buckley, Christopher (1988). Term-weighting approaches in automatic text retrieval. *Information Processing and Management, 24*(5), 513–523.

Sebastiani, Fabrizio (2002). Machine learning in automated text categorization. *ACM Computing Surveys, 34*(1), 1–47.

Shojania, Kaveh G., Sampson, Margaret, Ansari, Mohammed T., Ji, Jun, Doucette, Steve, & Moher, David (2007). How quickly do systematic reviews go out of date? A survival analysis. *Annals of Internal Medicine, 147*(4), 224–233.

Shonjania, Kaveh G., & Bero, Lisa A. (2001). Taking advantage of the explosion of systematic reviews: An efficient MEDLINE search strategy. *Effective Clinical Practice, 4*(4), 157–162.

Sindhwani, Vikas & Keerthi, S. Sathiya (2006). Large scale semi-supervised linear SVMs. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '06* (pp. 477–484). New York, NY, USA, ACM. ISBN 1-59593-369-7.

Swanson, D. R. (1986). Fish oil, Raynaud's syndrome and undiscovered public knowledge. *Perspectives in Biology and Medicine, 30*(1), 7–18.

Tan, Chade-Meng, Wang, Yuan-Fang, & Lee, Chan-Do (2002). The use of bigrams to enhance text categorization. *Information Processing Management, 38*(4), 529–546. ISSN 0306-4573.

The Cochrane Collaboration. (2011). Cochrane Handbook for Systematic Reviews of Interventions. Version 5.1.0 [updated March 2011].

Thomas, James, McNaught, John, & Ananiadou, Sophia (2011). Applications of text mining within systematic reviews. *Research Synthesis Methods, 2*(1), 1–14. ISSN 1759-2887.

University of York (2008). *Undertaking systematic reviews of research on effectiveness: CRD's guidance for those carrying out or commissioning reviews*. NHS Centre for Reviews and Dissemination, University of York.

Valentini, G., & Masulli, F. (2002). Ensembles of learning machines. *Neural Nets WIRN Vietri-02 Series Lecture Notes in Computer Sciences, 2486*, 3–19.

Vapnik, Vladimir N. (1995). *The nature of statistical learning theory*. Springer.

Wallace, Byron, Trikalinos, Thomas, Lau, Joseph, Brodley, Carla, & Schmid, Christopher (2010). Semi-automated screening of biomedical citations for systematic reviews. *BMC Bioinformatics, 11*(1), 55.

Yang, Y. & Liu, X. (1999). A re-examination of text categorization methods, In *22nd Annual international SIGIR* (pp. 42–49).

Yang, Yiming, & Chute, Christopher G. (1994). An example-based mapping method for text categorization and retrieval. *ACM Transactions on Information Systems, 12*(3), 252–277.