UNIVERSITY OF GOTHENBURG

Machine Learning for Reducing the Effort of Conducting Systematic Reviews in SE

Bachelor of Science Thesis in the Program Software Engineering&Management

Chuan Su

University of Gothenburg
Chalmers University of Technology
Department of Computer Science and Engineering
Göteborg, Sweden, January 2014

Machine Learning for Reducing the Effort of Conducting Systematic Reviews in SE

Chuan Su

Examiner: Morgan Ericsson

University of Gothenburg
Chalmers University of Technology
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone +46 (0)31-772 1000

Cover Picture taken from : http://www.uq.edu.au/uqabroad/university-of-gothenburg

Department of Computer Science and Engineering
Göteborg, Sweden, January 2014

Machine Learning for Reducing the Effort of Conducting Systematic Reviews in SE

Chuan Su*
Department of Computer Science and Engineering
IT University of Gothenburg / Chalmers University of Technology, Gothenburg, Sweden

ABSTRACT **Objective :** To investigate whether machine learning and text-based data mining can be used to support the primary studies selection process and decrease the needed efforts in systematic reviews conducted in the context of SE.

Research Design : A test collection was built from 3 systematic reviews used in previous work in the context of SE. The proposed probabilistic classifier based on *Bayes' Theorem* was constructed to predict and classify each article as containing high-quality evidence to warrant inclusion in study selection process or not. Feature engineering techniques were applied to the abstract-based features. Cross-validation experiments were performed to evaluate the efficiency of the document classifier. Three metrics - *precision*, *recall* and *specificity* were used together to measure the classification performance. We assume that a recall rate of 0.9 or higher is required for the classifier to identify an sufficient quantity of relevant papers. As long as recall is at least 0.9, the *Precision* and *Specificity* should be as high as possible,.

Results : From the hold-out cross validation experiment, the precision achieved with the classifier for two systematic review topics, was 93%, while 79% for another systematic review topic. The results of leave-one-out cross validation experiment were presented in three Confusion Matrix, which in detail indicated that the precision achieved with the classifier for the three systematic review topics was promising in terms of predicting relevant abstracts while relatively poor in terms of excluding irrelevant articles.

Conclusion : The classifier based on *Bayes' Theorem* has strong potential for performing the systematic review classification tasks in software engineering. The approach presented in this paper could be considered as a possible technique for assisting labor-intensive primary studies' selection process in an SLR.

Keywords : Machine learning, Systematic review, Naive Bays classifier, Text classification, Software engineering, Metrics, Recall

1. INTRODUCTION

The systematic literature review (SLR) has been considered as one of the key components of the application of evidence-based paradigm in software engineering (Kitchenham and Charters, 2007). SLRs were first introduced in the software engineering (SE) field in 2004 (Kitchenham, 2004) and have since been growing in popularity among software engineering researchers. The informal literature reviews frequently seen in literatures do not explicitly define the search process or the data extraction process (Kitchenham and Charters, 2007); hence it may be vulnerable to bias in both conduct and outcome, so that providing readers with a distorted view about the state of knowledge regarding the area at the focus of

* Corresponding author.
Email address : gussuchs@student.gu.se.

the review (Felizardo et al., 2011). In contrast, SLRs employ a well-defined methodical process of identifying, analyzing and interpreting all available evidence related to a specific research question, which makes it less likely that the results of the study that is performed are biased. The research papers summarized in an SLR are referred to *primary* studies, while the SLR itself is a form of *secondary* study (Brereton et al., 2007). "The aim of an SLR is not just to aggregate all existing evidence on a research question; it is also intended to support the development of evidence-based guidelines for practitioners. " (Kitchenham et al., 2009).

However, Kitchenham and Chaters (2007) pointed out that the major disadvantage of an SLR is that they require considerably more efforts than traditional literature reviews. Most of activities involved in an SLR are conducted manually, and their undertaking tends to be both labor-intensive and time consuming. In particular, the selection of primary studies according to predefined criteria in an SLR is challenging, especially when a large volume of 'irrelevant' results are returned by search methods. Moreover, with the selection of primary studies is performed by two or more reviewers, uncertainties about any primary studies sources for which agreement cannot be reached should be further investigated through sensitivity analysis, which implies additional efforts to re-read the studies classified by reviewers (Kitchenham and Charters, 2007). The mass of papers to be read, analyzed, and possibly re-read, as a result, makes it more challenging for researchers to synthesize the state of the art of a particular topic of interest. Thus, there is a significant demand for tools that will facilitate the study selection activity in systematic reviews.

In recent years, the application of Machine learning based approaches to text classification problems, according to which a general inductive process automatically builds a classifier by learning from the knowledge of the predefined categories and of a set of training instances belongs to them, has witnessed a booming interests (Fabrizio, 2002). In ML approaches the pre-classified documents are the key resources for the automated classification of text documents (Fabrizio, 2002). This seems to be particular beneficial with regard to the systematic discovery of relevant primary studies in SLRs. When conducting an SLR, reviewers usually keep detailed records of their search strategies, in particular, the articles for which they have reviewed the abstracts and read full text, and finally, which articles include sufficient high-quality evidence to warrant inclusion in study selection process (Cohen, 2006). Meanwhile reviewers also maintain a record of those candidate primary studies that are excluded as a result of more detailed inclusion / exclusion criteria (Kitchenham and Chaters, 2007). This process motivated our interest in using these key resource - the pre-classified documents to train a machine-learning based document classification system that would have the ability to predict which candidate articles that have not been reviewed were most likely to include evidence warranting inclusion in the selection of primary studies process. And the classification system could decrease the amount of documents that require manual review and therefore reduce the workload of selecting primary studies in a systematic review.

Within this study, we investigate whether machine learning and text-based data mining technologies can be used to support the primary studies selection process and decrease the needed efforts in systematic reviews conducted in the context of EBSE. More specifically, this study addresses the following questions :

I. Whether a machine learning-based classifier can reduce the reviewer's workload by excluding irrelevant documents in SE ?

II. How efficient the classifier is at assisting the primary studies selection process in Systematic Review in SE ?

As a result, this paper presents the investigators' application of a machine learning-based approach to reduce the labor required in performing a systematic review in the context of software engineering. The remainder of this paper is organized as following : The next section, Section 2, refers

to existing research related to this study. In Section 3, this study's design is described. Section 4 presents the study results followed by the discussion of our study results, limitations and future work in Section 5. Finally, we conclude out work in Section 6.

2. BACKGROUNDS AND RELATED WORK

2.1 Use of SLRs in Software Engineering

The SLR provides methodologically rigorous review of research results based on three clearly defined phases: (i) planning; (ii) conducting; and (iii) reporting the review (Kitchenham, 2004). *Fig.1* illustrates the overall 10-stage review process (Brereton et al., 2007). During the planning phase, the need for a systematic review is confirmed and a review protocol is established, which aims to minimize bias in the study by defining in advance how the systematic review is to be conducted (Kitchenham and Charter, 2007). When conducting the review process, all the relevant potential articles need to be located using an interactively refined search of many electronic sources such as IEEExplore, ACM Digital library and Google scholar. Selection of primary studies is then performed on all potential relevant studies by applying inclusion and exclusion criteria designed to screen out non-relevant studies (Kitchenham and Charter, 2007). Next, significant information is extracted and synthesized from the selected studies. Usually, the selection of primary studies requires at least two reviewers to review each abstract by applying predefined inclusion and exclusion criteria to access their relevance (Kitchenham and Charter, 2007). Primary studies which provide direct evidence about the research questions should be identified in this step. Abstracts that meet the inclusion criteria are subjected to the next stage - study quality assessment where reviewers read the article abstracts and the complete articles and develop a series of quality instruments to evaluate each studies with the goals of achieving 100% precision (Kitchenham and Charter, 2007). In the final phase, reviewers report the results of the review and circulate the results to potentially interested parties (Kitchenham and Charters, 2007).

Three previous studies (Kitchenham et al., 2009, 2010; da Silva et al., 2010) have been performed with the goal of assessing the use of SLRs in software engineering research. The first study developed by Kitchenham et al. (2009) suggested that the spread of software engineering topics addressed by SLRs was fairly limited and the main stream software engineering topics were not well represented. Due to the limitations of the first study that the search was manual and performed on a restricted set of source, Kitchenham et al (2010) extended their study in the year of 2010 by undertaking a broader automated search for other SLRs. The results indicated that the number of SLRs was increasing as well as the overall quality of the studies (Kitchenham et al., 2010). However, the authors also emphasized the issue that only a very small portion of SLRs evaluated the quality of primary studies (Kitchenham et al, 2010). Da silva et al. (2010)'s assessment study showed that the major limitation with the use of SLRs in SE is that a large number of SLRs failed to assess the quality of their primary studies and the number of SLRs providing guidelines to practitioners is still small, which confirmed the previous finding of Kitchenham et al. (Kitchenham et al., 2009, 2010).

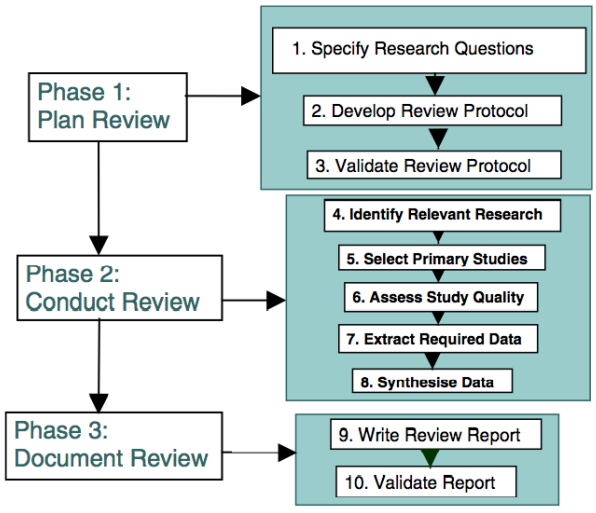

Fig. 1. SLR process (Brereton et al., 2007)

With the increasing number of SLRs performed on various topics within SE discipline, many empirical studies have also been carried out to report experiences of researchers and possible challenges when conducting SLRs in SE. Riaz et al. (2010) made a summary of the potential challenging aspects of the process to be : (i) formulation of research questions (Brereton et al., 2007; Staples and Niazi, 2008); (ii) conducting searches (Dyba et al., 2007); (iii) selection of primary studies (Dyba and Dingsoyr, 2008); and (iv) primary studies's quality assessment (Dyba et al., 2007, Staples and Niazi, 2008), from previous studies reported by various researchers. The problems faced by reviewers while conducting SLRs and the approaches adopted to address these problems vary across studies. One particular issue involves the selection of primary studies. Brereton et al (2007) research identified a problem with the quality of software engineering abstracts and pointed out that the standard of IT and software engineering abstracts is too poor to rely on when selecting primary studies. Other studies also provide evidence that the unstructured and poorly written abstracts can complicate the study selection process (Dybå et al., 2007; Kitchenham et al., 2008; Dybå and Dingsøyr, 2008). One solution to minimize the difficulties encountered in the primary study selection process in SE is to advocate the use of structured abstracts (Kitchenham et al., 2008).

2.2. Machine Learning and the SLR Process

Several studies have investigated the potential benefits of text mining and machine learning techniques in supporting SLRs process. Ananiadou et al. (2009, cited in Felizardo, 2010) employed text mining and machine learning techniques to support three different activities involved in an SLR : (i) search, (ii) study selection, and (iii) syntheses of the data; however their research concentrated on the field of social science. Without testing it is difficult to determine whether their findings could be successfully apply to creating systematic reviews in SE, particularly given the relative immaturity of study reporting in this field (Kitchenham et al., 2008).

The use of reference lists as a part of search strategy to locate and identify relevant primary studies for SLRs has been proposed and used by practitioners for several years (Skoglund and Runeson, 2009) and it was also suggested in Kitchenham et al. guideline (Kitchenham, 2004) for performing SLRs in SE. Felizardo et al. (2010) work suggested a use of meta-data analysis of documents via graphical representations such as citation maps - visualization of citation relationships among papers in supporting primary studies' selection and selection review activities. Their study has investigated the use of VTM (visual text mining) techniques to help with the selection of primary studies in the process of SLR within the context of EBSE. Felizardo et al. (2010) work also presented us one strategy to classify primary studies which is to identify the regions (clusters) of documents with similar content in terms of their titles, abstracts and keywords by applying *k-means* clustering algorithm. Using this technique, clusters are created automatically followed by the formation of their associated topics (Felizardo et al. , 2010) However, it is remarkable that the approach they used for the automatic classification of primary studies is quite different from the ones discussed here. Aside from (i) the automatic assignment of documents to a predefined set of categories, which is the *machine learning approac*h to text classification problem; their approach was subjected to (ii) the automatic identification of such a set of categories *and* the grouping of document under them, a task usually called *text clustering* (Fabrizio, 2002).

The main steps involved in the task of text classification using ML approach are (i) document preprocessing, (ii) feature extraction, (iii) model selection, (iv) training and testing the classifier. *Fig.2* illustrates the main steps in machined learning-based approach to the text classification problem (Mita and Mukesh, 2011). *Document pre-processing* significantly reduces the size of the input text documents and usually involves the activities such as stop-word elimination (Kim et al, 2006; Zhang et al, 2007; Hao et al, 2008) and stemming (Porter, 1980). *Feature extraction* first transformed the input data into sets of features and then extract the relevant information from the feature sets,

which is accomplished using methods like TF-IDF,(Jones, 1972), LSI (Deerwester et al., 1990), multi-word (Zhang et al., 2007 and Church et al., 1990) etc. In the context of text classification, features or attributes are usually represented in the form of significant words, multi-words or frequently occurring phrases indicative of the text category (Mita and Mukesh, 2011). After feature extraction, an appropriate machine learning algorithm is applied to train the text classifier using a set of pre-classified documents which are presented as feature sets (document vectors) (Mita and Mukesh, 2011). The classifier is then evaluated and tested on a test set of documents. If the classification accuracy of the trained classifier is proved to be acceptable for the test set, then this model could be utilized to classify new instances of text documents (Mita and Mukesh, 2011).

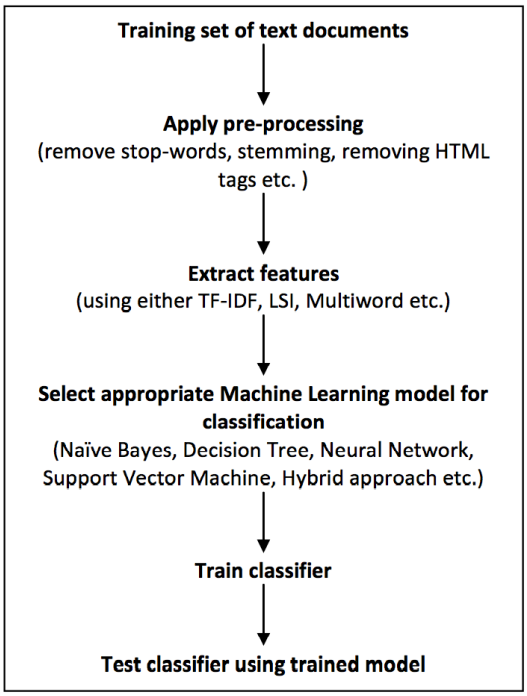

Fig. 2. Main steps in machine-learning approach to text classification (Mita and Mukesh, 2011)

Various machine learning algorithms such as Naive Bayes (Kim,et al., 2006 and Meena, et al., 2009), Neural Networks (Wang, et al. 2006), Support Vector Machine (Wang, et al. 2006 and Zhang, et al., 2008), and Decision Tree (Quinlan, 1986) have been proposed by researchers for the automatic text classification. SVM classifier is found to be very effective for 2-class text classification problems (Mita and Mukesh, 2011), which could also be used to judge the relevance of a given document to a particular category. Recent experimental research (Su et al., 2008, Rennie et al., 2003 and Mccallum et al., 1998) reveals that the modified version of the classical naive Bayes classifier yield a better performance than that of SVM classification approach such as *multinomial naive Bayes (MNB)* and *complement naive Bayes (CNB)*. But none of these studies targets to the problems of SE primary studies classification.

Decisions of Bayesian classifier are presented in terms of frequency of occurrence of words within a given text document (Stan et al., 2010). Since the Naive Bayesian approach is purely statistical, the classification by Bayesian classifier are easily understandable and its implementation is straightforward (Fabrizio, 2002). While *naive Bayes* is quite effective in various data mining tasks, we want to further investigate whether a simple probabilistic classifier could be extended to the challenging and practically important context of systematic review classification of SE.

3. RESEARCH DESIGN

In order to achieve the goals of this paper, we decided to implement a classifier which relies upon *naive Bayes* Theory, once trained, will classify previously un-reviewed articles as either relevant or non-relevant to the topic of the systematic review in SE, with sufficient precision in excluding the non-relevant articles. Although human efforts will still be required, there will be significant labor savings.

We built and evaluated the document classification system for the primary studies selection in SLRs in four phases : (i) building text collections for each of 3 SE review topics; (ii) preprocessing the text collection and applying feature engineering; (iii) text classification; (iv) classifier precision and performance evaluation. Below we discuss each of the four phases.

3.1 Text Collection

To validate our research results, we used

static and publicly available data - previously published SLR reports within SE discipline. The text collection we built contained 3 groups of articles for which systematic reviews were build targeting three SE classes (software process improvement, software architecture, research methods) in the field of SE. These three SLR reports were mapped into a public domain electronic library (also referred to digital library) that is, the abstracts of paper evaluated and triaged by the SE reviewers were fetched from that collection (e.g. IEEExplore, ACM Digital library and Google scholar and so on). In our research, we obtained the abstracts from their summery table instead of repeating all the SLR process steps they used to obtain the candidates papers (see *Appendix I* for details).

Table 1 gives information about the 3 SE classes for which systematic review were built. It is worth noting that we only utilize a limited number of pre-classified documents as the initial input for our data extraction process rather than the whole set of data in the text collection, as our main purpose is to demonstrate the applicability and effectiveness of the ML technology in the selection of primary studies process in an SLR. Moreover, it is reasonable to assume that the less articles the classifier used to achieve sufficient classification proficiency, the more efficiency the classifier is and the more manual work saved.

Table 1 Text collection description

SE class review	No. of abstract Reviewed by SE researchers	% judged relevant	No.of Relevant Abstract Included in Test Collection	No. of Non-relevant Abstract Included in Test Collection
Measurement of SPI	10817	1.4	15	15
Software Architecture	3036	1.9	15	15
SLR in SE	2506	0.7	13	13

We listed in this table the number of abstracts in each class reviewed by SE researchers, the percentage of the relevant abstracts among all the abstracts, as well as thenumber of relevant and non-relevant abstracts included in our test collection for each review.

We can observe from Table 1 that the pre-selected data (publications) involved in SLRs are highly imbalanced, with many more abstracts are judged non-relevant by SE reviews than are found relevant. Successful training a classifier to identify low-probability classes such as the articles included in theses reviews can be challenging (Cohen, 2006). In our research design, we decided to balance the amount of abstract for each class (here, the relevant class and the non-relevant class) in the text collection for our further processing.

3.2 Preparing the data

Each article in a text collection was first transformed into a feature vector which included all the single words from the title, abstract and keywords. In addition to the features of single word, it is also essential to utilize words collocation - a sequence of words that have certain tendency to be appeared together as significant features for the statistical analysis of texts. We applied regular expression pattern to each vectors of words to convert the most frequently appeared phases to their abbreviations. For instances, the phases *"software process improvement (s)*" and *"software process (es)*" in each vectors of words in the text collection of *SPI measurements* were replaced with "*SPI*" and "*SP*" respectively. And the phase "software architecture" in Software architecture text collection was replaced with " SA".

In order to reduce the size of input data and the impacts of different variants of the same "words" on classification performs, we filtered the text with a stop list of 300 most common English words and applied the Porter stemming algorithms to each single word features. Table 2 shows the number of significant features for each SE class review.

The first column gives the number of statistically significant features found in the training data for that review. The last three columns break the total number of features down to three distinct categories: number of

abstract-based features, number of title-based features and number of keyword-based features.

Table 2 Number of Significant Features

SE class review	Total No. of significant Features	Abstract-based Features	Title-based Features	Keywords-based Features
Measurement of SPI	3859	3435	242	182
Software Architecture	3786	3491	230	65
SLR in SE	3537	3293	218	26

We used the bag-of-words (BOW) model to represent each text collection. Each feature was thus treated as a natural number, present the (frequency of) occurrence of each single words (incl. multiwords phases abbreviation) appeared in an article, resulting in a feature vector consisting of entirely zeros or positive numbers in ***N***-dimensional space where ***N*** is the total number of words extracted from the text collection. The whole text collection is then presented as a ***N*** x ***J*** matrix where ***J*** is the number of articles in the text collection. As a note, the feature vector for each article ***j*** in a text collection has the same number of elements, ***N***.

3.3 Text Classification

We used *naive Bayes* classifier as a classification algorithm. It is a well-known and practical probabilistic classifier and has been employed in many applications (Kim et al., 2006).

In the context of text classification, the probability that a document represented by a ***n-dimensional*** vector $\vec{d_j} = (\omega_1, \omega_2, ...\omega_n)$ of features belongs to a class c_i is calculated by the application of *Bayes' theorem* as follows :

$$p(c_i \mid \vec{d_j}) = \frac{p(\vec{d_j} \mid c_i)p(c_i)}{p(\vec{d_j})}$$

In the equation above, $p(\vec{d_j})$ is the probability that a randomly picked document has vector $\vec{d_j}$ as its representation (because, $p(\vec{d_j})$ does not depend on c_i and the values of the features ω_i are given, so that it is effectively constant.). and $p(c_i)$ is the probability that a randomly picked document belongs to the category c_i (Fabrizio, 2002).

Within this framework, the input text document is treated by the *naive Bayes* classification model as an ordered sequence of word occurrence ,with each word occurrence as an independent trial (Kim, 2006). Thus it is common to make the assumption that any two coordinates of the document vector are, when viewed as random variables (features), statistically independent of each other (Le Zhang et al. , 2004); This *independence assumption* is encoded by the equation :

$$p(d \mid c_i) = \prod_{k=1}^{|T|} p(w_k \mid c_i)$$

Pseudocode for calculating the conditional probability of each features would like this :

Count the number of documents in each class
for every training document
for each class:
if a token appears in the document →
increment the count for that token
increment the count for tokens
for each class:
for each token:
divide the token count by the total token
to get conditional probabilities
return conditional probabilities for each class

However, when we calculate the product of $p(\omega_1 \mid c_i)p(\omega_2 \mid c_i)...p(\omega_N \mid c_i)$,we will get underflow as many of these numbers are very small(the result eventually rounds off to zero when multiplying many small numbers in Python). To avoid underflow and round-off error problem, we take the natural logarithm of this product rather than the number itself. The formula (1) may be written as

$$\log(c_i \mid \overrightarrow{d_j}) = \log p(c_i) + \sum_{k=1}^{n} \log p(w_k \mid c_i) - \log p(\overrightarrow{d_j})$$

As our intention is to judge whether or not an un-reviewed documents is relevant for the primary studies selection, the document space in our research was thus partitioned into two categories: $\overline{c_i}$ *(relevant)* and its complement $p(\overline{c_i}) = 1 - p(c_i)$ *(non-relevant)*.We may further obtain that :

$$\log(\overline{c_i} \mid \overrightarrow{d_j}) = \log(1 - p(c_i)) + \sum_{k=1}^{n} \log p(w_k \mid \overline{c_i}) - \log p(\overrightarrow{d_j})$$

And a document was classified as relevant if $\log p(c_i \mid \overrightarrow{d_j}) > \log p(\overline{c_i} \mid \overrightarrow{d_j})$, otherwise non-relevant.

3.4 Evaluation of classification efficiency in systematic review

To evaluate how our classifier approach perform on identifying new or un-reviewed articles for inclusion, a *hold-out cross validation* approach was used where the dataset was randomly split into training and validation (test) data. For each such split, the text classifier learned from the training data, and predictive accuracy of the trained classifier is assessed using the validation data. The final results are the average over the splits.

Hold-out cross validation method avoids the overlap between training data and test data, yielding a more accurate estimate for the general performance of the machine-learning algorithm (Payam, 2008). However, the downside of this method is that the available data resource can not be fully utilized for the evaluation of classifiers and the results are highly dependent on the choice of the *training/ test* split. The way that iterating the procedure of hold-out cross validation multiple times and averaging the results over iterations may reduce the negative effects of this problem, but unless this iteration is performed in a systematic manner, some data may have always been in the test set while others are not tested at all, or conversely some data may fall into the test set multiple times and have never been able to contribute to the learning phase (Payam,2008). To deal with these challenges and utilize the available data to the max, we further decided to conduct another cross-validation approach - *leave-one-out-cross-validation* (LOOCV).

In LOOCV process, a single observation of the document data set is retained as validation data and the remaining observations are used for training the document classifier. This evaluation procedure is repeated, with each observation in the data set is used once as the validation data. An accuracy estimate obtained using LOOCV is known to be almost unbiased and it is widely used where the available data are limited (Payam, 2008), which is also another reason for us to select LOOCV as our second evaluation method.

The most common metric for evaluating the precision of a machine-learning based document classifier in predicting relevant articles is *precision*. *Precision* is defined as :

$$P = \frac{Number of Relevant Documents Correctly Classified}{Total Number of Documents Classified as Relevant}$$

The use of *Precision* as the metric could indicated and evaluated the efficiency of a document classifier, but it does not take the achieved recall and the amount of excluded irrelevant document into account. The efficiency of a systematic review ML classifier is also reflected in the other two aspects : (1) to reduce the number of missed articles containing high quality evidence by minimizing the number of relevant documents excluded by the classifier *(Recall)* ; (2) to reduce reviewers' workload by excluding maximized amount of irrelevant documents (Stan, 2010). The two metrics for measuring the recall of a machine learning based classifier (how completely the classifier identify the relevant articles) and the ability of excluding irrelevant articles, are *Recall*, and *Specificity Recall* (R) and *Specificity* (S) are defined as

$$R = \frac{Number of Relevant Documents Correctly Classified}{Total Number of Relevant Documents in Text Collection}$$

$$S = \frac{Number Of Irrelevant Documents Correctly Classified}{Total Number of Irrelevant Documents In Text Collection}$$

As in LOOCV each single observation is tested through each iteration on the training data, *the number of relevant document correctly classified*, *the total number of documents classified as relevant* and *the number of irrelevant document*s correctly classified can be easily acquired, so that the value of these three metrics could be achieved.

For our research, the three metrics - *Precision*, *Recall*, *Specificity* were used together to evaluate and measure the performance (efficiency) of the document classifier. We assume that a recall rate of 0.9 or higher is required for the classifier to identify an sufficient quantity of relevant papers. As long as recall is at least 0.9, the *Precision* and *Specificity* should be as high as possible.

4. RESULTS

We first performed the hold-out cross validation experiment, to evaluate the general performance of our classifier. For each text collection, we randomly selected a portion of our data for training set (20 out of 30 total articles) and a portion for test set (10 out of 30 total articles). We then iterate through the test set and classify each article in the test set. If an article is not classified correctly, the error count is incremented, and finally the total percentage error is reported. To get a good estimate of the error rate, we repeated this procedure ten times, and averaged the results in the end of test. We also statistically recorded the experiment data for each iteration. Figure 2 presented the error percentage of each test iteration and the final result of average error

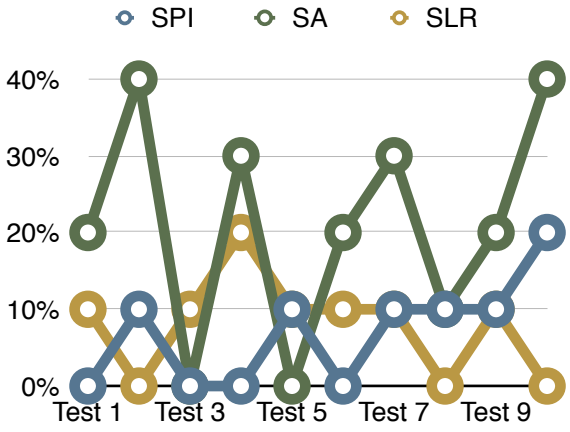

Fig2.Results of 10-times hold out cross-validation

rate for the three text categories was presented in Figure 3.

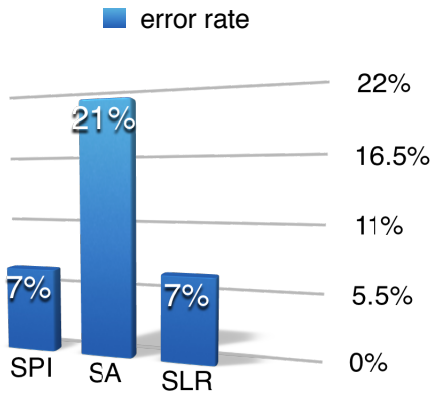

Fig.3. Final results of hold-out cross validation

We then performed the LOOCV experiment, to evaluate the actual precision of our classifier for inclusion and exclusion. LOOCV was performed 30 times for SPI , SA text collection and 26 times for SLR text collection. The number of positive documents and negative documents correctly classified and mis-classified during the experiment was recorded respectively. We used confusion matrix, which is a specific table layout that provide information about actual and predicted classifications achieved by a document classifier, to present the LOOCV results of each of three text collections *(see Confusion Matrix 1, Confusion Matrix 2 and Confusion Matrix 3 below)*. The columns in the first row, *Non-relevant* and *Relevant (**Predicted Category**)* in each of the three matrix, represents the number of irrelevant documents (in ***Actual Category***) that were correctly classified and incorrectly classified (*classified as relevant)* respectively, while the columns of second row represents the number of relevant documents that were correctly classified and mis-classified *(classified as non-relevant)* respectively.

Based on the information presented in these three confusion matrix, the precision (***P***), recall (***R***) and specificity (***S***) achieved by the classification system for each of the three SE text collections were calculated in the end of experiment (*See Figure 4*)

Confusion Matrix 1. LOOCV Results

SPI Text Collection		Predicted Category	
		Non-relevant	Relevent
Actual Category	Non-relevant (15)	13	2
	Relevant (15)	0	15

Confusion Matrix 2. LOOCV Results

SA Text Collection		Predicted Category	
		Non-relevant	Relevent
Actual Category	Non-relevant (15)	10	5
	Relevant (15)	2	13

Confusion Matrix 3. LOOCV Results

SLR Text Collection		Predicted Category	
		Non-relevant	Relevent
Actual Category	Non-relevant (13)	12	1
	Relevant (13)	2	11

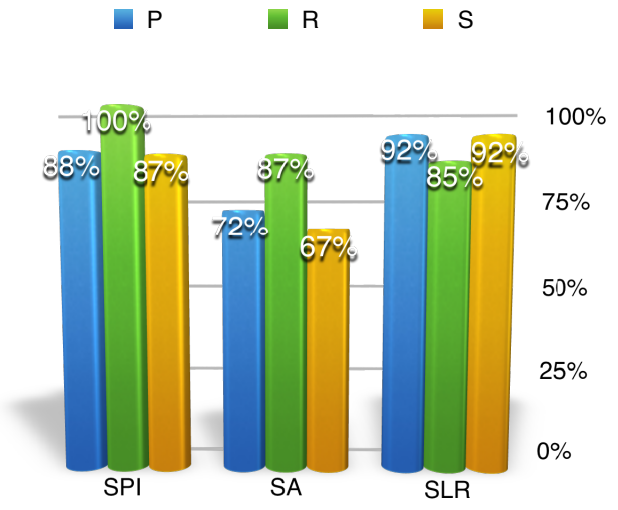

Fig.4. Results of leave-one-out-cross validation

5. DISCUSSION

Our research represented a means of training an automated document classifier to select articles with the highest likelihood of containing evidence warranting the inclusion for the primary study selection process in an SLR.

The cross validation methods should provide an accurate estimate of the classification performance and the results have demonstrated that *Naive Bayes* classifier has strong potential for assisting the labor-intensive systematic reviews process in SE. From *Figure 3* we found that the classification system performed very well on SPI and SLR review topics but not on SA class with over 20% error rate. *Confusion Matrix 2* and *Figure 4* further demonstrated in detail that the precision the classifier achieved on SA class is comparatively poor -7 out of 30 articles were misclassified, where 5 out of 15 irrelevant documents were classified into positive class. Even though its recall rate (87%) is quite closed to 90% - the required recall rate (indicating that the classifier has the ability to achieved adequate fraction of relevant articles), the lower accuracy rate of excluding irrelevant papers (S 67%) would imply additional efforts of review compared to the other text collection. For the SPI group the classifier achieved a recall rate of 100% for including the relevant articles but still two irrelevant documents were mis-classified into positive class (S 87%). As decisions of Bayesian classifiers are presented in terms of frequency of occurrence of words within a given abstract (Stan et al. ,2010), we therefore conducted an analysis looking at the most significant features for each of the three tasks to investigate the problems of lower accuracy for excluding irrelevant documents. We observed that the presence of many statistically significant features among the relevant abstracts tend to be highly consistent. For instance, the features *"SPI","measurements","metrics"* appeared many times in each relevant articles in SPI class, *"SA" , "evaluability", "quality"* presented frequently in each relevant article in SA group and it is also the same case with SLR group - *"systematic", "review", "search", "criteria", "results"*. These significant

features were then treated by classifier systems as the most strongly predictive dimensions with which to separate positives from negatives.

In contrast that the inclusion for each review study are more specific, most irrelevant abstracts returned in search strategy varied widely as they may target at different topic area, so that many significant features appeared in one abstract were rarely presented in other irrelevant papers. The difference in topic-specificity and incongruence of the significant features among each excluded paper contributed to the lower number of strongly predictive features, meaning that the classifier system had fewer dimensions with which to triage the negative documents. In order to improve the precision for excluding irrelevant papers, further work is necessary to enhance the topic-specificity of the articles in negative group before performing document classification on the SE review work process.

We may observe that the accuracy rate achieved for both predicting relevant papers (P) and excluding irrelevant papers (S) in *SLR* text collection is surprisingly higher compared to others. We further investigated the observations and found that the samples of excluded papers in SLR text collection were selected from hundreds of non-relevant documents by authors. These data shared some consistent characteristics - *survey or informal literature review,* which tend to be more topic-specific (*see Appendix I*). When going though the feature set, the word features, "survey", "literature", "experience" occurred many times in each excluded papers. Even more important, the dominated features for inclusion (positive features) - " systematic", "criteria", "review", "statistical" rarely appeared in negative groups. Thus, the strong uniqueness of significant features for both positive and negative categories promote the classifier to adequately model the triage process. In contrast, the uniqueness of positive features in the other two text collections tend to be relatively weak to perform the tasks of exclusion. Examining the abstracts that were mis-classified into the category of relevance, we observed that some strong predictive features for inclusion (positive features), such as " SPI", "improvements" in the text collection of SPI measurements, "SA", "architecture", "quality", "evaluation" in the text collection of software architecture were presented frequently in their mis-classified abstracts. Even though these mis-classified articles targeted at the same research fields - software process improvement and software architecture as the included documents, they were not relevant to the specific research problems - *measurement of software process improvements*, *evaluability of software architecture* under these research area. The frequency of these word features might be used by the classifier as the main dimensions with which to triage the documents that are relevant to its research field, however there may not be enough strong classification concepts for the classifier to exclude the abstracts which are irrelevant to specific research questions under such fields. In this research design, the use of abbreviations of SE phases as single features is our attempt to enhance the uniqueness of word features for classification tasks. In order to investigate impact of the uniqueness of word features on the classification performance, we conducted the experiments on the original feature sets instead of using abbreviations. However, the differences between the results of these two tests were not apparent. Despite this, we still believe further exploration to enhance the typicality of word features is essential for achieving sufficient accuracy rate of classification. In future work, we plan to apply weight engineering which refers to modifying the existing training data by weighting some of the attributes more than others, to the frequency features that are more targeted and specific to the research questions. And it makes sense to give some feature weights bigger than other features, if this is likely to improve the performance of a given learning algorithm (Stan et al. 2010). For instance, the features "metrics", "assessment", "measure" tend to be more specific to the research questions but the probability of these features were not as higher as "SPI", "improvements" from our feature set given the positive category.

Based on our analysis, we believe that the lower number of strongly predictive features for exclusion and the weaker typicality of

positive features are the main reasons why the performance on excluding irrelevant papers were relatively poor.

In practice, when selecting primary studies it is quite difficult to screen out 100% of irrelevant articles by simply applying inclusion and exclusion criteria on title and abstracts in the initial selection stage. As a rule the potential selected articles need to be assessed for their actual relevance in the "Study Quality Assessment" stage. From this point of view, we believe our result is still of practical importance as it demonstrated the classifier achieved sufficient precision rate in terms of including relevant papers in the initial screening phase of primary studies' selection process.

Our current study has several limitations. One of them relates to the illustration of only three SE review topics, consequently, the utilization of machine-learning technology was investigated in a limited context. It would be interesting to further investigate which review topics or what types of topics in SE the machine learning-based text classifier will provide expected benefits for. Second, we only utilized a limited set of abstracts as the input data to the classification tasks, it is essential to employ a full version of SLR process for further evaluating the efficiency of a given document classifier. In addition, we also believe that additional work is also necessary to investigate, how many articles (training data) is adequate for a machine-learning based document classifier to provide sufficient accuracy of predicted classification. As previously mentioned above, we balanced the amount of abstracts in the positive and negative class for the text classification process. In general, the initial systematic review data returned by the search strategy contain a large majority of irrelevant abstracts (sometimes more than 99%) (Stan et al. 2010). Thus, it is practically important to deal with the impacts of highly imbalanced rate of the training data in class distribution on text classification tasks through improving the feature engineering process and classification algorithms. Furthermore, in our current approach we are only using the words from the title, keywords and abstract as potential classification features. Further work is necessary to explore the possible improvements of classification performance by employing full text classification methods. On the one hand, using full text classification methods may to some extent reduce the impacts of abstracts with poor quality on the training processes. On the another hand, Porter stemming algorithm may have a more consistent and beneficial effect on full text (Cohen et al, 2006). We have also experimented without applying porter stemming algorithms to the original text, and we have found that the classification performance of "stemming words" representation equals to that of representation with 'raw' (unnormalized) data.

6. CONCLUSIONS

Our research has demonstrated that a simple and efficient probabilistic classifier based on *Bayes' theorem* has strong potential for performing the systematic review classification tasks in software engineering. The approach presented here could therefore be considered as a possible technique for assisting labor-intensive primary studies' selection process in an SLR.

Future work will be focused on investigating the possible classification performance by applying a modified version of classical *naive Bayes* classifier such as *multi nominal naive Bayes* and *complement naive Bayes*. We also plan to explore other state-of-art machine-learning approaches and techniques to design an automated document classification system to address the classification tasks with specific data characteristics in SLRs, such as the highly imbalance rate for the available training data. In addition, we will also concentrate on investigating the possible feature engineering techniques to enhance the feature typicality for the systematic review classification tasks.

We encourage other software engineering researchers to investigate applying machine-learning based approaches to the primary studies selection and other activities involved in SLRs workflows in SE.

Acknowledgments
The author thanks **Morgan Ericsson**, the supervisor of this thesis work, for his guidance, encouragement, review and helpful suggestions through this research. The author also acknowledges the University of Gothenburg for providing the opportunity to pursue this bachelor's study.

References

Brereton, P., Kitchenham, B.A., Budgen, D., Turner, M., and Khalil, M. 2007. Lessons from applying the systematic literature review process within the software engineering domain. The Journal of Systems and Software, 80 (2007), 571-583.

Church K. W., and Hanks P. 1990. Word association norms, mutual information and lexicography. Computational Linguistics, Vol. 16, No. 1, pp. 22-29.

Cohen AM, Hersh WR, Peterson K, et al. Reducing workload in systematic review preparation using automated citation\classification. 2006. J Am Med Inform Assoc 2006;13:206e19.

da Silva, F.Q.B., Santos, A.L.M., Soares, S., França, A.C.C. and Monteiro, C.V.F., Six Years of Systematic Literature Reviews in Software Engineering: an Extended Tertiary Study, Proc. International Conference on Software (ICSE' 10), IEEE Computer Society, Cape Town, South Africa, 10 pages, 2010.

Dyba, T., Dingsoyr, T., and Hanssen, G.K. (2007) Applying systematic reviews to diverse study types: An experience report. Empirical Software Engineering and Measurement (ESEM 2007), Madrid, Spain, 20-21 Sept. 2007, pp. 225-234.

Dybå, T. and Dingsøyr, T. (2008) Strength of evidence in systematic reviews in software engineering. In Proceedings of the Second ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, Kaiserslautern, Germany, 9-10 Oct. 2008. ACM, New York, NY, pp. 178-187.

Deerwester S., Dumais S. T., Landauer T. K., Furnas G. W., and Harshman R.. 1990. Indexing by Latent Semantic Analysis. Journal of American Society of Information Science, 41(6), pp. 391-407.

Felizardo, K.R. ; Salleh, N. ; Martins, R.M. ; Mendes, E. ; MacDonell, S.G. ; Maldonado, J.C. , Using Visual Text Mining to Support the Study Selection Activity in Systematic Literature Reviews ., Empirical Software Engineering and Measurement (ESEM), 2011 International Symposium on., pages 77-86

Felizardo, K.R., Nakwgawa, E.Y, Feitosa, D., Minghim, R. and Maldonado, J.C., An Approach Based on Visual Text Mining to Support Categorization and Classification in the Systematic Mapping, Proc. International Conference on Evaluation and Assessment in Software Engineering (EASE '10) BCS- eWiC, Keele University, UK, 10 pages, 2010.

Fabrizio S., 2002. Machine learning in automated text categorization. ACM Computing Surveys, 34(1):1-47.

Hao Lili., and Hao Lizhu. 2008. Automatic identification of stopwords in Chinese text classification. In proceedings of the IEEE international conference on Computer Science and Software Engineering, pp. 718 – 722.

Jones K. S. 1972. A statistical interpretation of term specificity and its application in retrieval. Journal of Documentation, Vol. 28, No. 1, pp. 11-21.

Kitchenham, B.A., Pretorius, R., Budgen, D., Brereton, O.P., Turner, M., Niazi,.M, and Linkman, S., Systematic literature reviews in software engineering – A tertiary study, Information and Software Technology, vol. 52, no. 8, pp. 792–805, 2010.

Kitchenham, B.A., Brereton, O.P, Budgen, D., Turner, M., Bailey, J., and Linkman, S., Systematic literature reviews in software engineering – A systematic literature review, Information and Software Technology, vol. 51, no. 1, pp. 7–15, 2009.

Kitchenham, B.A., Brereton, O.P., Owen, S., Butcher, J. and Jefferies, C., Length and readability of structured software engineering abstracts, IET Software, vol. 2, no. 1, pp. 37–45, 2008.

Kitchenham, B.A., Charters, S., Guidelines for Performing Systematic Literature Reviews in Software Engineering . Tech. Rep. EBSE 2007-001, Keele University and Durham University Joint Report, 2007.

Kitchenham, B.A., 2004. In: Procedures for Undertaking Systematic Reviews. Joint Technical Report, Computer Science Depart- ment, Keele University (TR/SE-0401) and National ICT Australia Ltd (0400011T.1).

Kim S., Han K., Rim H., and Myaeng S. H. 2006. Some effective techniques for naïve bayes text classification. IEEE Transactions on Knowledge and Data Engineering, vol. 18, no. 11, pp. 1457-1466.

Le Zhang, Jingbo Zhu and Tianshun Yao. An evaluation of statistical spam filtering techniques. Journal ACM Transcations on Asian Language Information Processing (TALIP) Vol. 3 no.4, pp. 243-269. 2004.

Mita K. Dalal, and Mukesh A. Zaveri. 2011. Automatic Text Classification : A Technical Review. International journal of computer applications (0975-8887) volume28-No.2.

Meena M. J., and Chandran K. R. 2009. Naïve bayes text classification with positive features selected by statistical method. In proceedings of the IEEE international conference on Advanced Computing, pp. 28 – 33.

Mccallum Andrew, Nigam Kamal. A comparison of event models for Naïve Bayes text classification, 1998.

Payam Refaeilzadeh, Lei Tang, Hua Liu. 2008. Cross-Validation. Arizon State University.
Porter M. F. 1980. An algorithm for suffix stripping. Program, 14 (3), pp. 130-137.

Quinlan J. R. 1986. Induction of Decision Trees. Machine Learning, pp. 81-106

Riaz, M., Sulayman, N., Salleh, M. and Mendes, E., Experiences Conducting Systematic Reviews from Novices' Perspective, Proc. International Conference on Evaluation and Assessment in Software Engineering (EASE' 10), BCS-eWiC, Keele University, UK, 10 pages, 2010.

Rennie JD, Shih L, Teevan J, et al. Tackling the poor assumptions of Naïve Bayes text classifiers. In: Proc Int Conf on Machine Learning. 2003:616 - 23.

Stan M., Alexandre K., Diana I, Oana F, Peter O., 2010. A new algorithm for reducing the workload of experts in performing systematic reviews. JAMIA 17(4): 446-453.

Skoglund, M., Runeson, P., 2009. Reference-based search strategies in systematic reviews. In: 13th International Conference on Evaluation and Assessment in Software Engineering. EASE' 09. BCS, Durham University, England, UK, p. 10 pages.

Staples, M. and Niazi, M. 2008. Systematic review of organizational motivation for adopting cmm-based software process improvement. Information and Software Technology, 50(7-8):605–620, 2008.

Su J, Zhang H, Ling C, et al. Discriminative parameter learning for bayesian networks. The 25th International Conference on Machine Learning, 2008, 1016-23.

Wang Z., Sun X., Zhang D., Li X. 2006. An optimal SVM- based text classification algorithm. In proceedings of the 5th IEEE international conference on Machine Learning and Cybernetics, pp. 1378 – 1381.

Wang Z., He Y., and Jiang M.. 2006. A comparison among three neural networks for text classification. In proceedings of the IEEE 8th international conference on Signal Processing.

Zhang M., and Zhang D.. 2008. Trained SVMs based rules extraction method for text classification. In proceedings of the IEEE international symposium on IT in medicine and Education, pp. 16 – 19.

Zhang W., Yoshida T., and Tang X. 2007. Text classification using multi-word features. In proceedings of the IEEE international conference on Systems, Man and Cybernetics, pp. 3519 – 3524.

Appendix I

Kitchenham et al. (2008) published their research paper "Systematic literature reviews in software engineering- A systematic literature review" ,where they listed 20 relevant studies and also a table described the candidate articles that not selected as a result of their systematic review process.

Systematic review studies.

ID	Author	Date	Topic type	Topic area	Article type	Refs.	Include practitioner guidelines	Num. primary studies
S1	Barcelos and Travassos [1]	2006	Technology evaluation	Software architecture evaluation methods	SLR	Guideline TR	No	54
S2	Dyba et al. [4]	2006	Research trends	Power in SE experiments	SLR	Guideline TR	No	103
S3	Galin and Avrahami [7,8]	2005 & 2006	Technology evaluation	CMM	MA	No	No	19
S4	Glass et al. [9]	2004	Research trends	Comparative trends in CS, IS and SE	SLR	No	No	1485
S5	Grimstad et al. [11]	2006	Technology evaluation	Cost estimation	SLR	Guideline TR	Yes	32
S6	Hannay et al. [12]	2007	Research trends	Theory in SE experiments	SLR	Guideline TR	No	103
S7	Jørgensen [15]	2004	Technology evaluation	Cost estimation	SLR	No	Yes	15
S8	Jørgensen [14]	2007	Technology evaluation	Cost estimation	SLR	No	Yes	16
S9	Jørgensen and Shepperd [16]	2007	Research trends	Cost estimation	SLR	GuidelineTR	No	304
S10	Juristo et al. [18,19]	2004 & 2006	Technology evaluation	Unit testing	SLR	EBSE paper	No	24
S11	Kitchenham et al. [20,21]	2006 & 2007	Technology evaluation	Cost estimation	SLR	Guideline TR	Yes	10
S12	Mair and Shepperd [27]	2005	Technology evaluation	Cost estimation	SLR	No	No	20
S13	Mendes [28]	2005	Research trends	Web research	SLR	Guideline TR	No	173
S14	Moløkken-Østvold et al. [31]	2005	Technology evaluation	Cost estimation	SLR	No	No	6
S15	Petersson et al. [32]	2004	Technology evaluation	Capture–recapture in inspections	SLR	No	No	29
S16	Ramesh et al. [34]	2004	Research trends	Computer science research	SLR	No	No	628
S17	Runeson et al.[35]	2006	Technology evaluation	Testing methods	SLR	EBSE paper	No [a]	12
S18	Torchiano and Morisio [38]	2004	Technology evaluation	COTS development	SLR	No	No	21
S19	Sjøberg et al. [36]	2005	Research trends	SE experiments	SLR	Guideline TR	No	103
S20	Zannier et al. [40]	2006	Research trends	Empirical studies in ICSE	SLR	No	No	63

Candidate articles not selected.

Source	Authors	Reference	Year	Title	Reason for rejection
TSE	T. Mens and T. Tourwé	30(2), pp 126–139	2004	A survey of software refactoring	Informal literature survey
TSE	S. Balsamo, A. Di Marco, P. Inverardi	30(5), pp. 295–309	2004	Model-based performance prediction in software development	Informal literature survey
IET Software	S. Mahmood, R. Lai and Y.S. Kim	1(2), pp 57–66	2007	Survey of component-based software development	Informal literature survey
IEEE Software	D.C. Gumm	23(5) pp. 45–51	2006	Distribution dimensions in software development	Literature survey referenced but not described in article
IEEE Software	M. Shaw and P Clements	23(2) pp. 31–39	2006	The golden age of software Architecture	Informal literature survey
IEEE Software	M. Aberdour	24(1), pp. 58–64	2007	Achieving quality in open source software	Informal literature survey
IEEE Software	D. Damian	24(2), pp. 21–27	2007	Stakeholders in global requirements engineering: lessons learnt from practice	Informal literature survey
JSS	E. Folmer and J. Bosch	70, pp. 61–78	2004	Architecting for usability: a survey	Informal literature survey
IST	Hochstein and Lindvall	47, pp. 643–656	2005	Combating architectural degeneration: a survey	Informal literature survey
IST	S. Mahmood, R. Lai, Y.S. Kim, J.H. Kim, S.C. Park, H.S. h	47, pp. 693–707	2005	A survey of component-based system quality assurance and assessment	Informal literature survey
TOSEM	J. Estublier, D. Leblang, A. van der Hoek, R. Conradi, G. Clemm, W. Tichy, D. Wiborg-Weber	pp. 383–430	2005	Impact of software engineering research on the practice of software configuration management	Informal literature survey
TOSEM	Barbara G. Ryder, Mary Lou Soffa, Margaret Burnett	pp. 431–477	2005	The impact of software engineering research on modern programming languages	Informal literature survey. No clear search criteria, no data extraction process.
ACM Surv	J. Ma and J. V. Nickerson	38(3), pp. 1–24	2006	Hands-on, simulated and remote laboratories: a comparative literature review	Not a software engineering topic
ISESE	S. Wagner		2006	A literature survey of the quality economics of defect-detection techniques	Informal literature survey although quantitative data tabulated for different testing techniques.