A Semantic Relatedness Approach for Traceability Link Recovery

Anas Mahmoud[1] , Nan Niu[1], and Songhua Xu[2]
[1]Dept. of Computer Science and Engineering, Mississippi State University, Mississippi State, MS
[2]Oak Ridge National Laboratory, One Bethel Valley Road, Oak Ridge, TN, USA, 37830
amm560@msstate.edu, niu@cse.msstate.edu, xus1@ornl.gov

***Abstract*—Human analysts working with automated tracing tools need to directly vet candidate traceability links in order to determine the true traceability information. Currently, human intervention happens at the end of the traceability process, after candidate traceability links have already been generated. This often leads to a decline in the results' accuracy. In this paper, we propose an approach, based on semantic relatedness (SR), which brings human judgment to an earlier stage of the tracing process by integrating it into the underlying retrieval mechanism. SR tries to mimic human mental model of relevance by considering a broad range of semantic relations, hence producing more semantically meaningful results. We evaluated our approach using three datasets from different application domains, and assessed the tracing results via six different performance measures concerning both result quality and browsability. The empirical evaluation results show that our SR approach achieves a significantly better performance in recovering true links than a standard Vector Space Model (VSM) in all datasets. Our approach also achieves a significantly better precision than Latent Semantic Indexing (LSI) in two of our datasets.**

***Index Terms*—information search and retrieval, automated tracing, semantic relatedness, experimentation.**

I. Introduction

Traceability, according to IEEE, is defined as: (1) the degree to which a relationship can be established between two or more products of the development process, and (2) the degree to which each element in a software development process establishes its reason for existing [1]. This definition is strongly influenced by the originators of traceability in the requirements engineering community. In particular, Gotel and Finkelstein [2] defined requirements traceability as "*the ability to describe and follow the life of a requirement, in both a forward and backward direction*".

The availability of traceability information among various software artifacts (e.g. source code, requirements, test cases and design) significantly reduces the amount of time required by developers to comprehend the system [3]. This can be vital in various software engineering activities, such as verification and validation (V&V) and impact analysis, where developers spend a considerable amount of their time building a mental model of the system and acquiring a holistic understanding of the task in hand [4].

The automated tracing process consists of three steps: indexing, retrieval, and presentation. In the indexing step, input artifacts are converted into more compact forms that are compatible with their underlying information retrieval (IR) models. In the retrieval step, IR algorithms, such as Latent Semantic Indexing (LSI) and Vector Space Model (VSM) [4], are used to identify a set of candidate traceability links by matching a trace query with artifacts in the software repository. In the presentation step, retrieved candidate traceability links are presented to the human analyst for further validation.

Although IR-based tools (e.g. ADAMS [5], RETRO [6]) help automate traceability link generation to a large extent, they are still shy from attaining optimal accuracy. Direct human judgment of candidate traceability links is still required in order to produce the final traceability matrix (TM) [7]. The underlying assumption is that, the humans' inherent capability of judging relatedness of concepts gives them an upper hand over automated methods. Human analysts employ their prior knowledge of application domain and natural language skills to overcome the vocabulary mismatch and concept assignment problems associated with software artifacts [8], and leverage more semantic relations to support their decisions when vetting candidate TM.

In practical settings, human intervention comes at the end of the traceability process (after-the-fact), after candidate traceability links have already been retrieved and ranked [9]. This process imposes a great challenge on the analysts' performance, as their decisions can be influenced by the candidate TM generated by an IR-based tool [10]. In fact, studies of assisted requirements tracing - in which human analysts work with the automated tracing tool to verify traceability links - suggest that human analysts working with such tools usually reduce the accuracy of the generated results [9]. In an attempt to conquer this challenge, in this paper, we bring human judgment into an earlier phase of the automated tracing process, by integrating it into the underlying retrieval mechanism. To achieve the goal, we use semantic relatedness (SR), a retrieval technique that can, to a large extent, imitate the human mental model of relevance.

Researchers have successfully applied SR to several natural language processing (NLP) applications such as automated spelling correction [11], text retrieval [12], word sense disambiguation [13], question answering [14], and automatic speech recognition [15]. The success of SR in these related domains has motivated this work to utilize it as the core measure for establishing the semantic relevance between software artifacts, i.e. traceability recovery. The main research

question is: By integrating human judgment into the retrieval process, through the use of SR, can we improve the overall performance of automated tracing tools?

To answer our research question, we first describe a set of requirements for integrating SR into the automated tracing process, and then conduct an experimental evaluation to validate our claims. We implemented and executed our approach on three datasets from different application domains and assessed the results using six different performance measures for evaluating various aspects of automated tracing tools. The evaluation produced encouraging results on the benefit of integrating SR for traceability link recovery. The rest of the paper is organized as follows. Section II provides background information about the traceability problem and semantic relatedness. Section III describes our experimental approach and the implementation process. Section IV presents analysis results. Section V describes the threats to validity. Section VI discusses the implications, and finally, Section VII concludes the paper and suggests potential research directions.

II. BACKGROUND AND RELATED WORK

A. Automated Tracing

The automated tracing problem can be defined as the ability to establish traceability links between various artifacts in a software system automatically, with no operator intervention, at the rates of 100% accuracy (precision) and 100% coverage (recall). While several solutions for this problem have been proposed in the literature, IR-based methods seem to be superior [4, 5]. IR methods aim to match a query of keywords with a set of objects in the software repository, and rank the retrieved objects based on how relevant they are to the query using a predefined similarity measure. IR methods heavily investigated in the automated tracing literature include: Vector Space Model (VSM) [4], Latent Semantic Indexing (LSI) [16], and Probabilistic Network Model [17, 18].

Extensive empirical studies have been conducted to evaluate the effectiveness of different IR-based automated tracing techniques. Converging evidence indicates that all the exploited methods so far are almost equivalent in their capability to capture almost the same information [19, 20]. In most cases, a recall of 90 - 100% is achieved at precision between 5-30% [20]. In general, IR-based traceability tools still suffer on the precision side. Such tools cannot give a high recall without also recovering too many false positives, leading to higher classification efforts when analysts manually verify candidate TMs. Classification in this context refers to identifying correct links and discarding false positives [21].

Motivated by these findings, recent traceability research has started focusing on other factors that can impact the overall performance. In general, current research trends in automated tracing can be categorized into three main categories: new retrieval methods, performance enhancement techniques, and the analysis of human's role in the tracing process.

For the first category, researchers investigated unconventional methodologies for traceability links recovery. For example, Sulattnov and Hayes [22] applied Swarm Intelligence, an artificial intelligence technique, to trace textual requirements artifacts. The approach was evaluated using two datasets. Results showed that the swarm intelligence based methods slightly outperformed the classical VSM based method in terms of precision and recall, but achieved statistically significant results in terms of DiffAR - a measurement related to results contrast (cf. Section III). McMillan et al. [23] proposed a technique for recovering traceability links by combining textual and structural information of software artifacts. The technique is based on the assumption that related requirements share related source code elements. Preliminary results showed some performance improvement compared to stand-alone text-matching methods. Gibiec et al. [24] used a web-based query expansion algorithm to trace stubborn requirements. The proposed approach was evaluated using a dataset from the healthcare domain. The results showed a significant performance improvement in a portion of these hard-to-trace requirements. Other examples can be found in [25, 26, 27]. In general, while these new techniques help improve some aspects of the process, none of them provided a universally superior solution to the problem, as most of the studies showed inconsistent performance across different datasets.

For the second category of research, researchers try to utilize other factors, beyond the underlying retrieval mechanism, for improving accuracy of tracing tools. For example, De Lucia proposed an incremental approach, based on user feedback analysis, to improve the retrieval performances by incorporating feedback from user classification decisions with the underlying retrieval mechanism [21]. Cleland-Huang et al. [28] introduced three performance enhancement strategies for incorporating supporting information into a probabilistic retrieval algorithm. The strategies include hierarchical modeling, logical clustering of artifacts, and semi-automated pruning of the probabilistic network. Capobianco et al. [29] proposed an approach that considers the nouns contained in an artifact content to derive the semantics of an artifact. Other examples can be found in [30, 31, 32]. Again, in all of these studies, mixed results were reported over various datasets, confirming De Lucia's speculation that "*there is an upper bound to the precision that can be achieved by an IR-based traceability recovery tool on a given software system*" [21].

Prompted by these findings, researches have started looking at the problem from a different perspective – the human analysts. These studies are aimed at understanding the way human analysts interact with the candidate traceability links in a generated TM, and the process they follow to make classification decisions. The research group led by Hayes and Dekhtyar reported a series of studies of analysts performing tracing tasks [7, 9, 10]. These studies revealed several interesting observations about human's tracing behavior, including: a) analysts usually fail to recover the true TM b) in general, all analysts, regardless of their tracing experience, classification effort and comfort level with tracing, tend to converge their final TM's toward a hot spot in the recall-precision space, and c) the initial TM accuracy is the most important factor impacting final TM accuracy [9].

The above brief review of current trends in automated tracing research shows that, on the one hand, the success of requirements tracing, as measured by the final TM, hinges largely on how analysts subjectively evaluate the candidate traceability links provided by IR methods, and on the other

hand, the performance of human analysts is influenced by the quality of the initial candidate TM generated by the tool. In an attempt to break this loop, we introduce a new approach that combines both IR methods and human judgment into one step through the use of SR. Next is a brief description of SR, its measurements and applications.

B. Semantic Relatedness (SR)

SR tries to quantify the degree to which two concepts semantically relate to each other by exploiting different types of semantic links connecting them. The main intent is to mimic human mental model when computing the relatedness of words. Human brain establishes the semantic relatedness between words based on the internal structures of their meaning, or the implied meanings of words [33]. For example, both words <*cow, horse*> imply a mammal that has four legs, hence, they can be considered related. Also, the words <*horse, car*> both refer to a transportation means for humans, from which perspective they can be considered related. Another aspect the brain examines is the frequent association between words. Words that often appear together are likely to be related. For example, the words <*table, chair*> appear together frequently, giving the human brain an indication of relatedness.

It is also important to mention that the degree to which a human relates words depends on his/her previous experience and accumulated knowledge about these words under different contexts. Therefore, different people might perceive different judgment regarding word relatedness. For example, a person who has never seen or used a computer before will consider the word pair <*mouse, keyboard*> unrelated. In an attempt to replicate this process computationally, SR measures observe word usages by utilizing common sense knowledge.

A wide range of methods for measuring SR are discussed in the literature. These measures mainly estimate word semantic relatedness by exploiting massive amounts of lexical knowledge, and using statistical techniques to leverage all possible relations that contribute to the similarity of concepts. Such knowledge is usually available in external sources including, Linguistic Knowledge Bases (LKB) such as WordNet [34], collaborative knowledge bases (CKB) such as Wikipedia [35], or general web search results [36] such as Google search. These data sources are described as follows.

Linguistic Knowledge Bases (LKB): created by highly trained linguists following clearly defined guidelines, where semantic links among concepts are manually assigned. Dictionaries are good examples of LKBs where their content is typically of high quality. However, LKBs have limited coverage as their size and content are limited. They also usually lack domain-specific vocabulary and can quickly be out-dated due to high maintenance costs [37].

Most of the LKB-based SR measures in the literature utilize WordNet as their main source of knowledge [38]. WordNet, introduced and maintained by Cognitive Science Laboratory of Princeton University, is a large lexical database of English verbs, nouns, and adjectives grouped into sets of cognitive synonyms called synsets [34]. WordNet-based techniques view WordNet as a graph and identify relatedness as the path length between concepts; the shorter the path the more similar the concepts [38]. Thorough investigation of such techniques revealed that they achieve a 35% correlation with human judgments [11].

Web-based: SR Measures that rely on web search results treat the whole web as a corpus by utilizing general purpose search engines such as Google [39]. While this source of knowledge might be the most comprehensive, the web search results typically do not exhibit any clear structure. In addition to the high noise-to-signal ratio typical of web search results, different websites have different structures, making knowledge extraction a complicated process with even more computational overhead. Also, there is a practicality concern resulting from initiating multiple Web search requests or long search queries.

Several web-based SR measures have been proposed in the literature. These measures, in general, generate confidence scores based on word co-occurrence using counts collected over very large corpora or lexical patterns extracted from text snippets returned by search engines. Performance studies of different state-of-the-art Web-based SR measures show that they achieve an average of 50% correlation with human judgment [40]; some other studies reported higher correlation levels up to 88% [41]. Example of SR measures that use Web search results can be found in [39, 40].

Collaborative Knowledge Bases (CKB): usually available on the Web, are maintained by volunteer communities from diverse domains and expertise. They contain massive amounts of knowledge including domain-specific terms found in LKBs. CKBs are usually built following a well-defined structure with better concentrated knowledge that eliminates a high percentage of noise usually returned by search engines. In CBKs, semantic links are leveraged implicitly rather being explicitly defined such as in LKBs.

Most well-known CKB-based SR measures utilize Wikipedia as their source of knowledge [35]. In 2001, Wikipedia was released to the public as a free, massive and constantly evolving source of knowledge expressed in natural language, opening a new horizon for enhancing NLP research [37]. Since then, Wikipedia has been exploited in various semantic relatedness measures [35, 42, 43]. These measures generally estimate word semantic relatedness by representing documents as vectors in the Wikipedia article space. CKB-based measures have been reported to be able to achieve a correlation rate of up to 75% with human judgment [42, 44].

Finally, in SR, it is important to differentiate measurements of similarity and relatedness between two concepts. Similarity is usually defined by considering the lexical relations of synonymy, or equivalent words (e.g. <*sick, ill*>) and hyponymy, or the type-of relation (e.g. <*ambulance, vehicle*>). Relatedness, on the other hand, extends the definition of similarity by examining all types of semantic relations that connect two concepts. Such relations include, in addition to the aforementioned two similarity relations, antonym, which is the opposite meaning (e.g. <*male, female*>), meronymy, or the part-of relations (e.g. <*room, hotel*>), functional relations of frequent association (e.g. <*patient, hospital*>) and other non-classical relations. In other words, similarity can be viewed as a special case of relatedness [11].

III. APPROACH AND IMPLEMENTATION

In this section we describe requirements for integrating human judgment in the automated tracing process. We start by proposing an experimental framework to assess the potential effect of SR on traceability link recovery. Figure 1 depicts our experimental framework, which also describes the way the problem is formulated. The framework shows the suggested enhancement over the conventional traceability process, where human judgment is integrated into the underlying retrieval mechanism. This approach is not to be confused with the approach presented in [21] which uses feedback from actual experts. Instead, as mentioned earlier, human judgment in our approach is integrated through the use of SR.

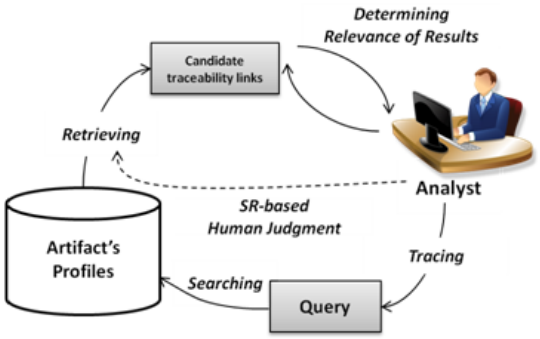

Fig. 1. Integrating SR into the tracing process.

This design entails several research questions, related to experimental settings, that need to be addressed prior to running the experiment. These questions include, what specific SR technique to exploit, what other IR methods should be used to compare with SR performance, what aspects of the performance to evaluate and the datasets to use. Next is a description of each of these design decisions.

A. Semantic Relatedness Measures

In this paper we use a Wikipedia-based semantic relatedness measure, namely Explicit Semantic Analysis (ESA), as our basic SR retrieval mechanism that simulates human judgment [44]. We adopt the ESA method due to two reasons: the nature of the task in hand (traceability) and the attributes of the different SR measures. In particular, we tease out the following requirements for our SR selection:

1) Correlation with Human Judgment: To integrate human common sense in the underlying retrieval mechanism, the selected SR measure should achieve high correlation with human judgment. An experimental comparison of equivalent semantic relatedness measures, using different knowledge sources, showed that Wikipedia-based measures significantly outperform WordNet in the same SR task, achieving a correlation coefficient of 0.75 with human judgments, higher than Google search result and WordNet based measures [35].

2) Text Comparison: The nature of traceability tasks requires comparing text fragments (profiles) to generate the candidate traceability links. WordNet-based methods and web-based methods are inherently limited to individual words matching. Getting these measures to compare longer text requires an extra level of sophistication [43]. However, ESA is a text matching technique, where fragments of text of arbitrary length can be compared in a similar way to single word comparison.

3) Performance: As mentioned earlier, LKBs are usually limited in coverage. While Web-search might be comprehensive, it usually includes a high noise-to-signal ratio. However, CKBs such as Wikipedia, which contains massive amounts of manually organized domain specific knowledge, achieve a balance between accuracy and coverage, thus motivating us to use a Wikipedia-based measure for SR estimation.

4) Practicality: In general, web search-based SR measures require initiating a web search request for each query [27]. This approach raises a major practicality issue because only a limited number of requests can be initiated in a certain amount of time, and usually there is a limit on the length of the query that can return sensible results. LKBs, such as, WordNet, do not suffer from the performance problem as they are usually available locally. Some CKB approaches suffer from that problem too, which however, is not the case for Wikipedia as its entire corpus can be downloaded[1] and treated as a local data source.

Overall, Wikipedia-based measures seem to be achieving significantly better results in relevant tasks than other knowledge sources. Wikipedia achieves a balance between accuracy and coverage, overcoming the limited coverage and scalability issue of LKBs, as well as the noise and practicality issues of web search-based SR measures.

Among the different Wikipedia-based measures, ESA [43] has been proven the most robust method, outperforming related measures such as WikiRelate [35] and WLM [44], in achieving high correlation with human judgment and with reasonable computation overhead. It also compares text fragment, making it a more fit approach for tackling traceability tasks. In addition, due to its flexibility, ESA has been extended to work in cross-lingual retrieval settings, which can be considered an extreme case of the vocabulary mismatch problem.

ESA represents the meaning of texts in a high-dimensional weighted vector of concepts derived from Wikipedia. In details, given a text fragment $T = \{t_1,\ldots, t_n\}$, and a space of Wikipedia articles C, initially, a weighted vector V is created for the text, where each entry of the vector v_i is the tf.idf weight of the term t_i in T. Using a centroid-based classifier [43], all Wikipedia articles in C are ranked according to their relevance to the text. Let $\langle k_j \rangle$ be the strength of association of term t_i with Wikipedia article c_j, $\{c_j \in c_1, c_2, \ldots, c_n\}$ (where N is the total number of Wikipedia articles). Then the semantic interpretation vector S for text T is a vector of length N, in which the weight of each concept c_j is defined as:

$$S_i = \sum_{w_i \in T} v_i . k_j \qquad (1)$$

Entries of this vector reflect the relevance of the corresponding articles to text T. Finally SR between two texts is calculated as the cosine between their corresponding vectors.

[1] http://en.wikipedia.org/wiki/Wikipedia:Database_download

B. Base Case Selection

To assess performance of ESA, a base case retrieval mechanism that is known to do well in traceability tasks is needed to put ESA performance in perspective. In our experiment, two IR methods, namely LSI and VSM, are used as base cases. As mentioned earlier, both methods are among the most utilized methods in the traceability literature, known to achieve a comparable performance on different datasets. Also, comparing three different methods with various levels of semantic support will help us better understand the role of semantics in recovering traceability information. Next is a detailed description of both methods.

Vector Space Model (VSM): In VSM, each document is composed by a set of terms $T = \{t_1 \ldots t_n\}$ and every term t_i is assigned a weight w_i using a certain weighting scheme. The terms in T are regarded as the coordinate axis in N-dimensional coordinate system and the terms weights $W = \{w_1 \ldots w_n\}$ are the corresponding values. Mathematically, if Q and D were two artifacts' represented in the vector space, then their similarity is measured as the cosine of the angle between them:

$$sim(Q, D) = \frac{\sum_{i=1}^{n} q_i \cdot d_i}{\sqrt{\sum_{i=1}^{n} q_i^2 \cdot \sum_{i=i}^{n} d_i^2}} \qquad (2)$$

In this study, we adopt tf.idf as our main weighting scheme, where $q_i = tf_i(q) \bullet idf_i$, $d_i = tf_i(d) \bullet idf_i$. $tf_i(d)$ and $tf_i(q)$ are term frequency of $term_i$ in *Q* and *D* respectively. idf_i is the inverse document frequency, and is computed as $idf_i = log_2(t/df_i)$, where t is the total number of profiles in the corpus and df_i is the number of profiles in which $term_i$ occurs.

Latent Semantic Indexing (LSI): LSI is a dimension reduction technique based on Singular Value Decomposition (SVD) [45]. It tries to find new coordinates of query and document vectors in a reduced *k*-dimensional space, then match them. LSI starts by constructing a term-document matrix (A) for terms and documents in the corpus. This matrix is usually huge and sparse. In a simpler design this matrix can contain basic word counts; however, in traceability research it is common to use tf.idf or log-entropy weights, such that, the final matrix contains the weighted vectors of all the documents in the corpus. SVD is then applied to decompose A into three new matrices $A = USV^T$ where T stands for transpose. Dimensionality reduction is then performed to produce reduced approximations of $<U, S, V^T>$ by keeping the top k eigenvalues of these matrixes. These reduced matrixes can be described as $<U_k, S_k, V^T_K>$. The best value of k can be obtained experimentally; however, a value in the range of [100 - 300] is frequently used. From the new reduced space we can derive the equation $V = A^TUS^{-1}$. Now assuming A is a matrix with n > 1 documents, for a given document vector d in A, d can be expressed as $d = d^TUS^{-1}$. In LSI, the query is also treated as a document, which is the case in traceability, where the query itself is a usecase or a piece of code. The query q can be expressed in the new coordinates of the reduced space as $q = q^TUS^{-1}$. Finally, in the k-reduced space q and d can be represented as $d = d^TU_kS_k^{-1}$ and $q = q^TU_kS_k^{-1}$ respectively. The similarity of q and d can then be calculated as the cosine measure:

$$sim(q, d) = sim(q^TU_kS_k^{-1}, d^TU_kS_k^{-1}) \qquad (3)$$

C. Evaluation Metrics

Sundaram *et al.* identified a number of primary and secondary measures to assess the performance of different tracing tools and techniques [47]. These measures can be categorized, based on their operation, into two groups as follows:

Quality: Precision (P) and Recall (R) are the standard IR metrics to assess the quality of the different traceability tools and techniques. Recall measures coverage and is defined as the percentage of correct links that are retrieved. Precision measures accuracy and is defined as the percentage of retrieved candidate links that are correct. F-measure is the harmonic mean of recall and precision. Based on the fact that, automated tracing methods emphasize recall over precision [4], the $F_2 = 5 \cdot (P \cdot R) / (4 \cdot P + R)$ measure which weights recall twice as much as precision is usually used.

Browsability: Browsability is the extent to which a presentation eases the effort for the analyst to navigate the candidate traceability links. For a tracing tool or a method that uses a ranked list to present the results, it is important to not only retrieve the correct links but also to present them properly to ensure an effective and efficient comprehension process. Being set-based measures, precision, recall and F_2 measure do not give any information about the list browsability. To reflect such information other metrics are usually used. Assuming h and d belong to sets of system artifacts $H = \{h_1 \ldots h_n\}$ and $D = \{d_1, \ldots, d_m\}$; $L = \{(d, h) \mid sim(d,h)\}$ is a set of candidate traceability links generated by the tool. LT is the subset of true positives (correct links) in L of true links, a link in this subset is described as (d,h). LF is the subset of false positives in L, a link in this set is described using the notion (d',h'). Secondary metrics can be described as:

- *Mean Average Precision (MAP)*: is a measure of quality across recall levels. It can be described as the mean precision scores after each relevant link retrieved. Equation 4 describes MAP. A method or tool that produces a higher MAP is superior.

$$\text{MAP} = \frac{1}{|H|} \sum_{j=1}^{|H|} \frac{1}{m_j} \sum_{k=1}^{m_j} \text{Precision}(L_{jT}) \qquad (4)$$

- *DiffAR*: measures the contrast of the list, it can be described as the difference between the average similarity of true positives and false positives in a ranked list. A list with higher DiffAR has a clearer distinction between its correct and incorrect links, hence, is considered superior. Equation 5 describes DiffAR.

$$\text{DiffAr} = \frac{\sum_{(h,d) \in L_t} \text{sim}(d,h)}{|L_t|} - \frac{\sum_{(h',d') \in L_f} \text{sim}(d',h')}{|L_f|} \qquad (5)$$

- *Lag*: can be described as the average of the number of false positives with higher similarity score that precede each true positive in the ranked list, in other words, the average number of incorrect links that appears before each correct link in the list. Equation 6 describes Lag.

$$Lag = \frac{\sum_{(h,d)\in L} \text{Lag}(d,h)}{|L|} \quad (6)$$

D. Datasets

Three datasets are used to conduct the experiment in this paper including: CM-1, eTour, and iTrust. Next, is a description of these datasets and their application domains:

- *iTrust*: a medical application, developed by software engineering students at North Carolina State University (USA). It provides patients with a means to keep up with their medical history and records and to communicate with their doctors. The dataset contains 314 requirements-source code links[2].
- *eTour*: an electronic tourist guide application developed by final year students at the University of Salerno (Italy). eTour was selected as experimental object in this experiment because its source code contains a combination of English and Italian words, which is considered an extreme case of vocabulary mismatch. The dataset contains 394 requirements-source code links[3].
- *CM-1*: consists of a complete requirements (high-level) document and a complete design (low-level) document for a NASA scientific instrument. The project source code was written in C with approximately 20K lines of code. It has 235 high-level requirements and 220 design elements. The traceability matrix contains 361 actual requirement-requirement traces[4].

Table I shows the characteristics of each dataset. The table shows the size of the system in terms of lines of source code (LOC), lines of comments (COM), source and target of traceability links – use cases (UC) or requirements (Req.), and finally, the number of correct traceability links.

TABLE I. EXPERIMENT DATASETS

	General Information		Traceability Information		
	LOC	**COM**	**Source**	**Target**	**Links**
iTrust	18.3K	6.3K	UC	SC	314
eTour	17.5K	7.5K	UC	SC	394
CM1	20K	N/A	Req.	Req.	361

E. Implementation

We extended our tool *TraCter*, previously introduced in [48] with LSI and ESA components to conduct the experiment described in this paper. ESA implementation was guided through several online resources[5]. These resources include tools for parsing Wikipedia dumps (e.g. *WikiPrep*[6]) and carrying out ESA analysis. Wikipedia 2009 dumps were used in our implementation. Also, using the indexing component provided in the tool, all three datasets were indexed and stored in the system database, along with their answer sets for evaluation purposes. A detailed description of the indexing process can be found in our previous work [31]. The tool is also provided with an evaluation component to compute and record the different performance measures under different execution settings.

IV. RESULTS AND ANALYSIS

Tables II and Figures 2-6 show the data collected during the experiment. Values of a certain measure (*R*, *P*, F_2, MAP, DiffAR, and Lag) are calculated after applying a certain retrieval method (VSM, LSI and ESA) to trace all the usecases in a particular dataset, averaged over a certain threshold level (0.2, 0.4, 0.6, 0.8 and 1 - cutting off values from the top of the retrieved list). To simplify the statistical analysis, each dataset is analyzed separately. Analysis of variance is carried out to draw general conclusions about the performance. We used the 0.05 alpha level (α=0.05) to test the significance. Analysis results are shown in Tables II - IV. This section describes and discusses these results.

A. Primary Measures Analysis

Recall: Figure 2 shows the recall data. Analysis of variance over the results (Table II) shows that ESA significantly outperforms VSM on all three datasets. However, it shows a mixed performance comparing to LSI. Only on iTrust dataset ESA significantly outperforms LSI. The results also show that even though LSI starts slowly with significantly poorer performance at lower thresholds, it usually catches up to ESA at higher levels. However, unlike LSI, which changed dramatically after a certain threshold, ESA shows more stable performance increasing gradually with the increase of the threshold level in all three datasets. In general, analysis of recall shows that, ESA was successful in reducing the omission error by capturing more correct links than a standard VSM, and showing a more stable performance than LSI.

Precision: Figure 3 shows the precision for each dataset. Analysis of variance of precision results (Table II) shows that VSM significantly outperforms ESA in iTrust and eTour datasets. However, the performance difference is not significant in CM1. The results also show that ESA significantly outperforms LSI in terms of precision in both CM1 and iTrust datasets. ESA also slightly outperforms LSI in eTour, however, improvement in the performance was not statistically significant. As shown in the recall analysis, at higher threshold levels, LSI was able to catch up with ESA in terms of recall, which was reflected in the precision as a result of the negative correlation between precision and recall. Again, CM1 has shown a somewhat different behavior. While in the eTour and iTrust datasets VSM outperforms ESA significantly, on CM1 the difference between VSM and ESA was not statistically significant.

[2] http://agile.csc.ncsu.edu/iTrust/wiki/doku.php.

[3] http://www.cs.wm.edu/semeru/tefse2011/.

[4] http://mdp.ivv.nasa.gov/mdp_glossary.html#CM1, 2005.

[5] http://www.cs.technion.ac.il/~gabr/resources/code/esa/esa.html

[6] http://www.cs.technion.ac.il/~gabr/resources/code/wikiprep/

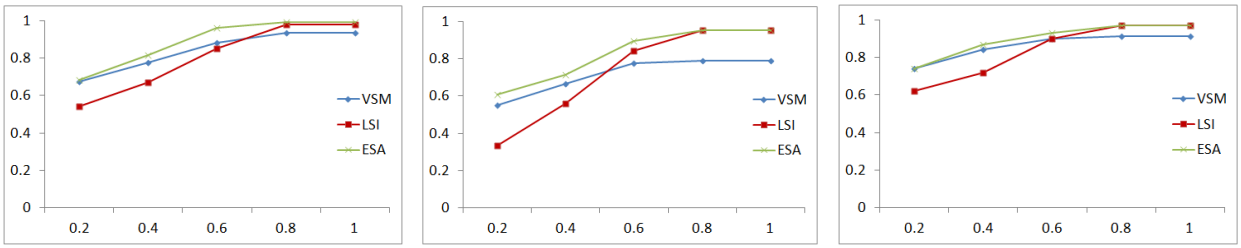

Fig. 2. Recall: iTrust, eTour and CM1 (x-axis: threshold level, y-axis: recall value)

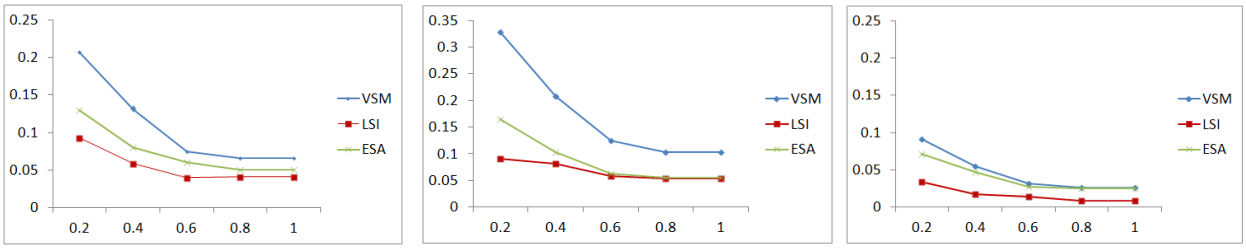

Fig. 3. Precision: iTrust, eTour and CM1(x-axis: threshold level, y-axis: precision value)

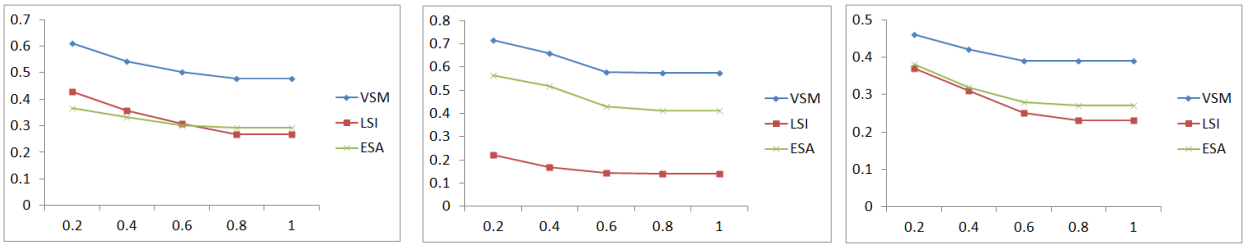

Fig. 4. MAP: iTrust, eTour and CM1(x-axis: threshold level, y-axis: MAP value)

TABLE II. QUALITY MEASURES' STATISTICAL ANALYSIS RESULTS

	Recall						**Precision**						**F_2**					
	ESA X VSM		ESA X LSI		VSM X LSI		ESA X VSM		ESA X LSI		VSM X LSI		ESA X VSM		ESA X LSI		VSM X LSI	
	F	*p*	*F*	*p*	*F*	*P*	*F*	*P*	*F*	*p*	*F*	*p*	*F*	*p*	*F*	*p*	*F*	*p*
ITrust	14.69	*.019*	9.78	*.035*	.79	.423	10.08	*.034*	13.33	*.02*	11.44	*.028*	12.250	*.019*	21.806	*.035*	16.807	*.015*
eTour	34.86	*.004*	3.43	.137	.023	.881	15.577	*.017*	1.761	.255	8.94	*.040*	87.11	*.001*	3.432	.138	24.299	*.008*
CM1	10.28	*.033*	3.63	.129	.35	.587	2.25	.21	42.25	*.003*	16.0	*.016*	1.882	.242	38.368	*.003*	18.843	*.012*

TABLE III. BROWSABILITY MEASURES' STATISTICAL ANALYSIS RESULTS

	MAP						**DiffAR**						**Lag**					
	ESA X VSM		ESA X LSI		VSM X LSI		ESA X VSM		ESA X LSI		VSM X LSI		ESA X VSM		ESA X LSI		VSM X LSI	
	F	*p*	*F*	*p*	*F*	*p*	*F*	*p*	*F*	*p*	*F*	*p*	*F*	*p*	*F*	*p*	*F*	*p*
ITrust	493.442	*.00*	.615	.447	818.0	*.00*	250.01	*.00*	112.66	*.00*	306.0	*.00*	28.930	*.006*	15.251	*.017*	24.369	*.008*
eTour	1125.0	*.00*	278.679	*.00*	1060.89	*.00*	140.167	*.00*	81	*.001*	361	*.00*	14.224	*.02*	34.565	*.004*	22.5	*.009*
CM1	200.643	*.00*	14.069	*.019*	89.814	*.001*	216	*.00*	13.50	*.021*	182	*.00*	19.231	*.012*	9.112	*.039*	10.530	*.032*

In fact at higher threshold levels, ESA was able to match VSM precision levels, which is a good sign that ESA, even though it achieved significantly higher recall rates at higher threshold levels, it still managed to keep the precision under control.

F_2 Measure: Analysis of variance of F_2 data (Table IV) shows that VSM is still dominating ESA and LSI, achieving statistically significant performance over LSI in all datasets. In CM1, ESA was able to almost match VSM performance, as no statistically significant difference was detected. Overall, F_2 analysis shows very similar patterns to the precision analysis however, the difference in the performance is more obvious due to the fact that the recall is also integrated in F_2, making the difference more obvious.

B. Secondary Measures Analysis

MAP: Figure 4 shows the MAP values of the three datasets. Analysis of variance (Table III) shows that VSM significantly outperforms LSI and ESA in all three datasets. It also shows that ESA significantly outperforms LSI in eTour and CM1, however, no significant difference in the performance is detected in iTrust dataset.

TABLE IV. F_2 VALUES

	iTrust			eTour			CM1		
	VSM	LSI	ESA	VSM	LSI	ESA	VSM	LSI	ESA
.2	.38	.19	.28	.44	.18	.31	.22	.11	.19
.4	.29	.14	.20	.38	.19	.23	.14	.06	.13
.6	.19	.11	.16	.27	.15	.17	.08	.04	.08
.8	.17	.11	.14	.24	.14	.15	.07	.03	.07
1	.17	.11	.14	.24	.14	.15	.07	.03	.07

Overall, ESA achieves a mediocre performance in terms of browsability, while it was more successful than LSI, it still could not beat VSM, due to the fact that more links were retrieved. However, it is able to achieve stable patterns at different threshold levels.

DiffAR: DiffAR data is shown in Figure 5. Such superior performance of VSM on this particular measure was actually expected due to the nature of its operation. VSM makes clear cut decisions when deciding whether two words are similar or not. However, in ESA, things are less obvious, due to the fact that a score is given to each of the two terms even though if they do not match lexically. Among the three methods, LSI produced the least DiffAR value. This can be explained based on the mathematical nature of LSI which produces really small similarity scores, as several multiplication processes are carried out over small numbers in the range [0 - 1].

Lag: Lag data is shown in Figure 6. VSM still outperform the other two methods significantly in all three datasets, but again, less links were retrieved using VSM, so such behavior is expected. The results also show that ESA is achieving significantly better performance than LSI in all datasets. This difference becomes obvious at higher thresholds, which shows that LSI tends to scatter the links all over the list, with high separation levels between the correct links, while the results are more centered in VSM and ESA. Also, it shows that LSI gives really low relevance scores to true positives which can be considered as a sign of random behavior.

V. THREATS TO VALIDITY

Several factors can affect the validity of our study. Construct validity is the degree to which the variables accurately measure the concepts they purport to measure [49]. In our experiment, there were minimal threats to construct validity as standard IR measures (recall, precision, and F_2), which have been used extensively in requirements traceability research, were used to assess the different methods applied. These measures were also complemented by another set of secondary measures (MAP, DiffAR, and Lag) that are used to provide more insights into the results, in particular, the browsability of the generated lists. We believe that these two sets of measures sufficiently capture and quantify the different aspects of tracing methods evaluated in this study.

Threats to external validity impact the generalizability of results. In particular, the results of this study might not generalize beyond the underlying experimental settings [49]. A major threat to the external validity comes from the datasets used in this experiment. In particular, two of these datasets were developed by students and may not be representative of a program written by industrial professionals.

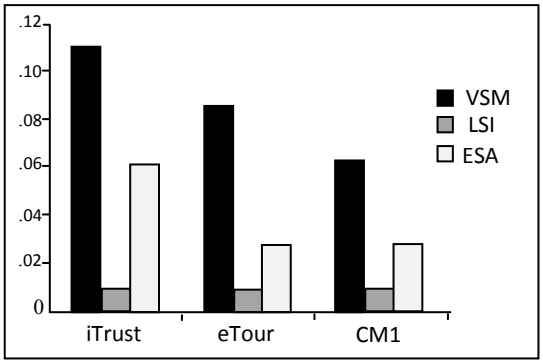

Fig. 5. DiffAR values for three datasets.

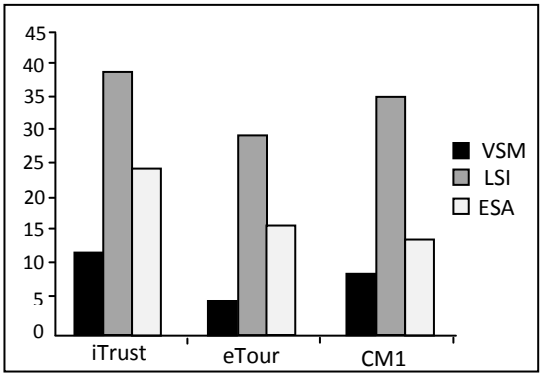

Fig. 6. Lag values for three datasets.

Also, all three of our datasets are limited in size which raises some scalability questions. However, we believe the use of three datasets, from different application domains, including requirements-to-source-code and requirement-to-requirement traceability tasks, helps mitigate related threats. Finally, specific design decisions and heuristics used during the implementation can also limit the results applicability. Such decisions include, using Wikipedia 2009 in ESA, using tf.idf weights and the heuristic value of k= 100 to calibrate LSI.

VI. DISCUSSION AND IMPLICATIONS

The research in this paper raises several questions about the nature of the automated traceability problem and its potential solutions. The first part of our definition of the problem states a 100% recall. This is based on the consensus among traceability researchers that an error of commission (false positive) is easier to deal with than an error of omission (false negative). Analysis of recall shows that ESA was indeed successful in achieving a better performance over VSM, reaching almost a 100% recall rate in all three datasets. LSI was also able to hit a maximum recall at higher threshold levels; however, it showed significantly poorer performance at lower thresholds. Based on these results, we argue that integrating human judgment, through the use of SR techniques, will help to leverage more advanced matching schemes, hence, achieving better recall. This argument also opens the door for several other questions related to using semantics. For example: how much semantics are needed? What effect does semantics have on the precision? And which semantically enhanced technique has more potential in achieving the desired 100% precision level stated early in our definition?

ESA can be considered as a more intelligent extension of LSI that looks beyond the classical semantic relations of synonyms and polynyms. Our analysis shows that LSI was able to match ESA recall at higher threshold levels. However, that came with a significant amount of noise that took the precision to significantly lower levels. To understand this behavior more, we refer to the secondary measures analysis, which shows that LSI was the least successful in terms of browsability, failing to distinguish between true and false positives, and also, scattering true links all over the list.

To gain a better insight into these three methods' internal operations, we pick two sample sub vectors (q_1, d_1) from one of our datasets, where ($q_1 \in Q_1$) and ($d_1 \in D_1$). q_1 = *<user, credential>* and d_1 = *<authenticate, email, password>*. q_1 represent a trace query, and d_1 is one of its true links. We observe the similarity scores given by each one of the exploited IR methods to each pair of terms in $q_1 \times d_1$. Results are shown in Table V.

TABLE V. SIMILARITY COMPARISON VIA A SAMPLE TRACE

	VSM	LSI	ESA
<user, authenticate>	0.00	0.00	.022
<user, email>	0.00	.0025	.766
<user, password>	0.00	.0020	.766
<credential, authenticate>	0.00	.0025	.860
<credential, password>	0.00	.0030	.896
<credential, email>	0.00	.0030	.876
Similarity:	**0.00**	**.0027**	**.882**

VSM failed to identify any relations between the two vectors, due to the lack of overlapping text. LSI, however, was able to detect some relatedness between different terms, while some of these relations might be explained based on LSI operation, such as a polynym relation between *<credential, password>,* there is no clear indication that that was the case, or just a mathematical coincidence. Finally ESA, was successfully able to detect a very strong similarity among the different terms, correlating the most with our personal judgment.

Such poor performance of LSI can be explained based on the operation of LSI. LSI builds its underlying model from the knowledge available within the system pool of artifacts, which is usually limited in size and coverage. This leads to a random behavior when trying to leverage semantic relations. ESA extends the knowledge base beyond the corpus, to cover more concepts [37]. The consideration of such relations gives ESA more stability and accuracy in its performance. Based on that, we argue that the injection of more relation has a positive impact on the performance. Even though LSI can be considered as a measure of SR, ESA can make more reliable relatedness decisions as its operation is independent from the amount of knowledge available within the system. Finally, its important to point out that although the proposed approach integrates human judgment, it does not actually utilize the judgment of experts. Instead it uses a general collection of documents (Wikipedia) which can be modified by anyone with or without knowledge of a particular software system.

VII. CONCLUSIONS

In this paper we proposed an approach based on Semantic Relatedness (SR) for automatic traceability link recovery. We described a set of requirements for integrating SR into the automated tracing process and conducted an experimental evaluation to assess our approach.

The experiment was conducted using three datasets from different application domains. Analysis of primary results showed that SR achieve a balance between LSI and VSM. It significantly outperforms the recall of VSM and significantly outperforms the precision of LSI (over two of our datasets), showing more stable performance at different threshold levels. In terms of secondary measures, VSM has shown the best performance. ESA achieved a mediocre performance, outperforming LSI, which exhibits the worst performance in terms of browsability. The research in this paper raised several questions related to various aspects of the automated tracing problem including semantics and human involvement. Future work will be focused on conducting more experiments using industrial size datasets. Also, the practicality of the approach will be evaluated through proper usability studies.

ACKNOWLEDGEMENT

This work is supported in part by the NSF (U.S. National Science Foundation) Grant CCF1238336. Songhua Xu performed this research as a Eugene P. Wigner Fellow and staff member at the Oak Ridge National Laboratory, managed by UT-Battle, LLC, for the U.S. Department of Energy under Contract DE-AC05-00OR22725.

REFERENCES

[1] IEEE Standard Computer Dictionary: Compilation of IEEE Standard Computer Glossaries. Institute of Electrical and Electronics Engineers Inc., 1991.

[2] O. Gotel and A. Finkelstein, An Analysis of the Requirements Traceability Problem, In *International Requirements Engineering Conference* (*RE)*, pp. 94-101, 1994.

[3] A. De Lucia, M. Di Penta, R. Oliveto, and F. Zurolo, Improving Comprehensibility of Source Code Via Traceability Information: A Controlled Experiment, In *International Conference on Program Comprehension* (*ICPC)*, pp. 317 –326, 2006.

[4] J. H. Hayes, A. Dekhtyar, and S.K. Sundaram, Advancing Candidate Link Generation for Requirements Tracing: the Study of Methods, *IEEE Transactions on Software Engineering* (*TSE)*, 32(01): 4–19, 2006.

[5] A. De Lucia, F. Fasano, R. Oliveto, and G. Tortora, Recovering Traceability Links in Software Artifact Management Systems Using Information Retrieval Methods, *ACM Transactions on Software Engineering and Methodology* (*TOSEM)*, 16(4): 13-50, 2007.

[6] J. H. Hayes, A. Dekhtyar, S. K. Sundaram, E. A. Holbrook, S. Vadlamudi, and A. April, REquirements TRacing On target (RETRO): Improving Software Maintenance Through Traceability Recovery, *Innovations in Systems and Software Engineering* (*ISSE)*, 3(3): 193-202, 2007.

[7] J. H. Hayes and A. Dekhtyar, Humans in the Traceability Loop: Can't Live With 'Em, Can't Live Without 'Em, In *International Workshop on Traceability in Emerging Forms of Software Engineering* (*TEFSE)*, pp. 20-23, 2005.

[8] T. J. Biggerstaff, B. G. Mitbander, and D. E. Webster, Program Understanding and the Concept Assignment Problem, *Commun. ACM* , pp. 72-82, 1994.

[9] A. Dekhtyar, O. Dekhtyar, J. Holden, J.H. Hayes, D. Cuddeback, and W. Kong, On Human Analyst Performance in Assisted Requirements Tracing: Statistical Analysis, In *International Requirements Engineering Conference (RE)*, pp. 111 - 120, 2011.

[10] D. Cuddeback, A. Dekhtyar, and J. H. Hayes, Automated Requirements Traceability: The Study of Human Analysts, In *International Requirements Engineering Conference* (*RE)*, pp. 231 - 240, 2010.

[11] A. Budanitsky and G. Hirst, Evaluating WordNet-based Measures of Semantic Distance, *Computational Linguistics*, 32(1):13-47, 2006.

[12] L. E. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman, and E. Ruppin, Placing Search in Context: The Concept Revisited, *ACM Transactions on Information Systems* (*ToIS)*, 20(1):116-131, 2002.

[13] S. Patwardhan, S. Banerjee, and T. Pedersen, SenseRelate:: TargetWord A Generalized Framework for Word Sense Disambiguation, In *National Conference on Artificial Intelligence (AAAI)*, pp. 73–76, 2005.

[14] D. Ahn, V. Jijkoun, G. Mishne, K. Muller, M. de Rijke, and S. Schlobach, Using Wikipedia at the TREC QA Track, In *Text REtrieval Conference* (*TREC)*, 2004.

[15] M. Pucher, WordNet-based Semantic Relatedness Measures in Automatic Speech Recognition for Meetings, In *Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions* (*ACL),* pp. 129-132, 2007.

[16] A. Marcus and J. Maletic, Recovering Documentation-to-Source-Code Traceability Links Using Latent Semantic Indexing, In *International Conference on Software Engineering* (*ICSE)*, pp. 125-135, 2003.

[17] G. Antoniol, G. Canfora, G. Casazza, A.D. Lucia, and E. Merlo, Recovering Traceability Links Between Code and Documentation, *IEEE Transactions on Software Engineering* (*TSE)*, 28(10): 970–983, 2002.

[18] J. Cleland-Huang, B. Berenbach, S. Clark, R. Settimi and E. Romanova, Best Practices for Automated Traceability, *Computer*, 40(6): 27-35, 2007.

[19] R. Oliveto, M. Gethers D. Poshyvanyk, and A. De Lucia, On the Equivalence of Information Retrieval Methods for Automated Traceability Link Recovery, In *International Conference on Program Comprehension (ICPC)*, pp. 68-71, 2010.

[20] J. H. Hayes and A. Dekhtyar, A framework for Comparing Requirements Tracing Experiments, *International Journal of Software Engineering and Knowledge Engineering* (*IJSEKE)*, 15(5): 751-782, 2005.

[21] A. De Lucia , R. Oliveto , and P. Sgueglia, Incremental Approach and User Feedbacks: a Silver Bullet for Traceability Recovery, In *International Conference on Software Maintenance* (*ICSM)*, pp.299-309, 2006.

[22] H. Sultanov and J. H. Hayes, Application of Swarm Techniques to Requirements Engineering: Requirements Tracing, In *International Requirements Engineering Conference* (*RE)*, pp.211-220, 2010.

[23] C. McMillan, D. Poshyvanyk, and M. Revelle, Combining Textual and Structural Analysis of Software Artifacts for Traceability Link Recovery. In *International Workshop on Traceability in Emerging Forms of Software Engineering* (*TEFSE)*, pp. 41-48, 2009.

[24] M. Gibiec, A. Czauderna, and J. Cleland-Huang, Towards Mining Replacement Queries for Hard-to-Retrieve Traces, In *International Conference on Automated Software Engineering (ASE)*, pp. 245-254, 2010.

[25] M. Eaddy, A. V. Aho, G. Antoniol, and Yann-Ganeuc. CERBERUS: Tracing Requirements to Source Code Using Information Retrieval, Dynamic Analysis, and Program Analysis, In *International Conference on Program Comprehension (ICPC),* pp. 53-62, 2008.

[26] M. Gethers, R. Oliveto, D. Poshyvanyk, and A. De Lucia, On integrating orthogonal information retrieval methods to improve traceability recovery, In *International Conference on Software Maintenance* (*ICSM)*, pp. 133-142, 2011.

[27] J. Cleland-Huang, A. Czauderna, M. Gibiec, J. Emenecker, A Machine Learning Approach for Tracing Regulatory Codes to Product Specific Requirements, In *International Conference on Software Engineering* (*ICSE)*, pp. 155-164, 2010.

[28] J. Cleland-Huang, R. Settimi, C. Duan, and X. Zou, Utilizing Supporting Evidence to Improve Dynamic Requirements Traceability, In *International Requirements Engineering Conference (RE)*, pp.135-144, 2005.

[29] G. Capobianco, A. De Lucia, R. Oliveto, A. Panichella, and S. Panichella, On the Role of the Nouns in IR-based Traceability Recovery, In *International Conference on Program Comprehension (ICPC)*, pp. 148 – 157, 2009.

[30] X. Zou, R. Settimi, and J. Cleland-Huang, Improving Automated Requirements Trace Retrieval: A Study of Term-based Enhancement Methods, *Empirical Software Engineering* (*EMSE)*, 15(2):119-146, 2010.

[31] A. Mahmoud and N. Niu, Source Code Indexing for Automated Tracing, In *International Workshop on Traceability in Emerging Forms of Software Engineering* (*TEFSE)*, pp. 3-9, 2011.

[32] X. Chen and J. C. Grundy, Improving Automated Documentation to Code Traceability by Combining Retrieval Techniques, In *International Conference on Automated Software Engineering* (*ASE)*, pp. 223-232, 2011.

[33] M. Hata, F. Homae, and H. Hagiwara, Semantic Relatedness Between Words in each Individual Brain: An Event-related Potential Study, *Neurosci Lett,* 1501(2):72-7, 2011.

[34] C. Fellbaum, WordNet: An Electronic Lexical Database. Cambridge, Mass.: *MIT Press*. 1998.

[35] M. Strube and S. P. Ponzetto, WikiRelate! Computing Semantic Relatedness Using Wikipedia, In *National Conference on Artificial Intelligence* (*AAAI)*, pp.1419-1424, 2006.

[36] J. Fang and L. Guo, Calculation of Relatedness by Using Search Results, In *Intelligent Systems and Applications* (*ISA)*, pp. 1 - 4, 2011.

[37] C. Müller and I. Gurevych, Using Wikipedia and Wiktionary in Domain-Specific Information Retrieval, In *Cross-language Evaluation Forum Conference on Evaluating Systems for Multilingual and Multimodal Information Access* (*CLEF)*, pp. 219-226, 2008.

[38] T. Pedersen, S. Patwardhan, and J. Michelizzi WordNet::Similarity - Measuring the Relatedness of Concepts, In *National Conference on Artificial Intelligence* (*AAAI),* pp.1024-1025, 2004.

[39] J. Gracia and E. Mena, Web-Based Measure of Semantic Relatedness, In *International Conference on Web Information Systems Engineering (WISE)*, pp. 136-150, 2008.

[40] N. Yang, L. Guo, J. Fang, and X. Chen, Semantic Relatedness Based on Searching Engines, In *International Conference on Computer Science and Information Technology* (*ICCSIT),* pp. 292 - 296, 2010.

[41] G. Spanakis, G. Siolas, and A. Stafylopatis, A Hybrid Web-Based Measure for Computing Semantic Relatedness Between Words, In *International Conference on Tools with Artificial Intelligence* (*ICTAI)*, pp. 441-448, 2009.

[42] E. Yeh, D. Ramage, C. D. Manning, E. Agirre, and A. Soroa, WikiWalk: random walks on Wikipedia for semantic relatedness, In *Workshop on Graph-based Methods for Natural Language Processing*, pp. 41-49, 2009.

[43] E. Gabrilovich and S. Markovitch, Computing Semantic Relatedness using Wikipedia-based Explicit Semantic Analysis, In *International Joint Conference on Artificial Intelligence* (*IJCAI)*, pp.1606-1611, 2007.

[44] D. Milne and I.H. Witten. An Effective, Low-cost Measure of Semantic Relatedness Obtained from Wikipedia Links, In *AAAI Workshop on Wikipedia and Artificial Intelligence* (*WIKIAI)* , 2008.

[45] S. Deerwester, S. T. Dumais, G. W. Wurnas, T.K Landauer, and R. Harshman, Indexing by Latent Semantic Analysis, *Journal of the American Society for Information Science and Technology* (*JASIST)*, (41) 6: 391–407, 1990.

[46] C. W. Cleverdon, J. Mills, and E. M. Keen, An Inquiry into Testing of Information Retrieval Systems, 2 vols, 1966.

[47] S. Sundaram, J. Hayes, A. Dekhtyar, and E. Holbrook, Assessing Traceability of Software Engineering Artifacts, *Requirements Engineering (REJ)*,15(3): 313-335 , 2010.

[48] A. Mahmoud, and N. Niu, TraCter: A Tool For Candidate Traceability Link Clustering, In *International Requirements Engineering Conference (RE)*, pp. 335-336, 2011.

[49] A. Dean and D. Voss. Design and Analysis of Experiments. Springer, 1999.