Enhancing Software Search with Semantic Information from Wikipedia

Xiaoli Ma and Bo Yuan

Abstract Software is becoming ubiquitous, from desktop computers to smart phones, and has created significant impact on the quality of our everyday life. Sharing and reusing high-quality software can save tremendous amount of time and efforts that otherwise would need to be reinvented. The challenge is how to efficiently search through a potentially huge database of software and return the most relevant results. In this paper, we present a prototype of semantic software search engine that exploits the semantic information from Wikipedia, one of the largest online knowledge repositories as the result of collaborative intelligence. We propose a technique to replace the original concept space by an extended concept space extracted from Wikipedia to incorporate commonsense knowledge into software search. Experimental results show that this strategy can achieve better performance over traditional software search based on the original concept space.

1 Introduction

To facilitate the sharing of software applications across diverse disciplines, it is often desirable to develop a mechanism to automatically collect software distributed on the Web and make them easily accessible by users. Currently, the most widely used method is keyword matching. However, understanding precisely the intention of users based on search keywords still remains as a major challenge. Fortunately, recent studies in semantics show that semantics can help extend our understanding of the original content. For example, by using ontology in search, especially in organizing resources and understanding search words, search engines can better understand the meaning of various concepts and produce more targeted outputs [1].

X. Ma • B. Yuan (✉)
Intelligent Computing Lab, Division of Informatics, Graduate School at Shenzhen, Tsinghua University, Shenzhen 518055, People's Republic of China
e-mail: thumxl@gmail.com; yuanb@sz.tsinghua.edu.cn

Previous studies show that good correlation between computed relatedness scores and human judgments can be achieved by using vectors of Wikipedia[1]-based concepts to represent the original texts [2]. As concepts represent meaning units for constructing knowledge, the concept space can, at least to some extent, serve as the commonsense and domain-specific knowledge in search applications. Wikipedia is the largest online encyclopedia and has been successfully exploited to assist the research in knowledge engineering and machine learning. Recently, it has been predominately used as additional text features for knowledge discovery and visualization in topic modeling, text categorization and clustering, semantic analysis, and other text processing tasks [3–6].

In this paper, we employed similar ideas in software search by using Wikipedia-based concept vectors to specify software applications. A semantic index is also built using this concept space. With the help of this fusion strategy, information from multiple sources can be unified to achieve a more complete understanding towards a specific software application, compared to its original description information. During the search process, the relatedness between software applications (represented as concept vectors) is calculated to rank the search results.

The contribution of this paper is twofold. First, we presented extended concept space construction (ECSC), a novel approach to forming an extended semantic representation of the original concept based on Wikipedia. This method can be also applied to other tasks requiring semantic information without building a comprehensive semantic system. Second, we applied ECSC in the task of software search and achieved promising results compared to traditional software search engines in terms of the consistency with the search results produced by human users.

The rest part of this paper is organized as follows. The strategy for extending the concept space of software for indexing and semantic search is detailed in Sect. 2. The major experimental results are presented in Sect. 3, along with a description of the data collection and evaluation procedure. A number of existing software search platforms and related studies are discussed in Sect. 4 and this paper is concluded in Sect. 5 with some directions for future work.

2 Semantic Search Using ECSC

It is well known that useful semantic information can be extracted by analyzing the log files of search engines to provide better search services. However, in many cases, the log files are not made accessible to the public. Instead, a more practical approach is to use publically available knowledge resources such as Wikipedia to acquire the semantic information to improve the search experience. More specifically, given a software description, a concept space (a vector of concepts) is constructed to represent its essential attributes (e.g., what each term is about and what it is).

[1]http://www.wikipedia.org/

For example, the description of Weka[2] (Waikato Environment for Knowledge Analysis) may read as “A suite of machine learning software developed by the Machine Learning Group at University of Waikato, containing a collection of popular machine learning algorithms for data mining tasks.” Ideally, a user that is interested in a certain data mining algorithm should be able to retrieve this software record using the name of the algorithm or even its alias as the keyword, even if it does not explicitly appear in the original description. In other words, the software search engine should be able to achieve a more general understanding of the software on top of its existing description.

In Wikipedia, each article is treated as a concept and each concept is represented by a vector of words that occur in the article. The strength of association between words and concepts can be computed by WLVM (Wikipedia Link Vector Model) [7] using the hyperlink structure of Wikipedia. The semantic relatedness of two Wikipedia articles is defined by the angle between the vectors of the links within them. A software package called WikipediaMiner[3] contains an implementation of WLVM, which was used in our work to compute the relatedness between concepts.

Using ECSC, each software application is annotated as a set of concepts (original concept space) and each concept is mapped into a weighted sequence of Wikipedia concepts ordered by their relevance (i.e., use Wikipedia concepts to augment the bag of concepts that represent the software) using WikipediaMiner. Concepts with low levels of relatedness are discarded. Finally, all sets of concepts are merged into a vector of concepts (extended concept space) as the new software description. When searching for software, the semantic relatedness between two software applications is calculated by comparing their vectors of concepts, for example, using the cosine metric. In the meantime, with the help of ECSC, for the same input keyword, it is now possible to retrieve software records that are closely related but otherwise would not have been possible to be retrieved. The overall process of ECSC is shown in Fig. 1.

To demonstrate how our approach works, the top 10 individual Wikipedia concepts most relevant to a given concept (e.g., SVM and Bayesian probability) is shown in Table 1. Table 2 shows the original concepts in the description of Weka as well as the extended new concepts. Table 3 shows the comparison of the list of concepts of Weka and Mlpy,[4] a Python open source machine learning library.

With the concepts of software applications, in our work, inverted index was built using Apache Lucene[5] along with other important attributes as the index content. In order to efficiently rank the search results, different weights were assigned to attributes. For instance, in Eq. (1), $v_{ij} = 1$ if the jth software record contained the ith

[2]http://www.cs.waikato.ac.nz/ml/weka/

[3]http://wikipedia-miner.cms.waikato.ac.nz/

[4]http://mlpy.sourceforge.net/

[5]http://lucene.apache.org/

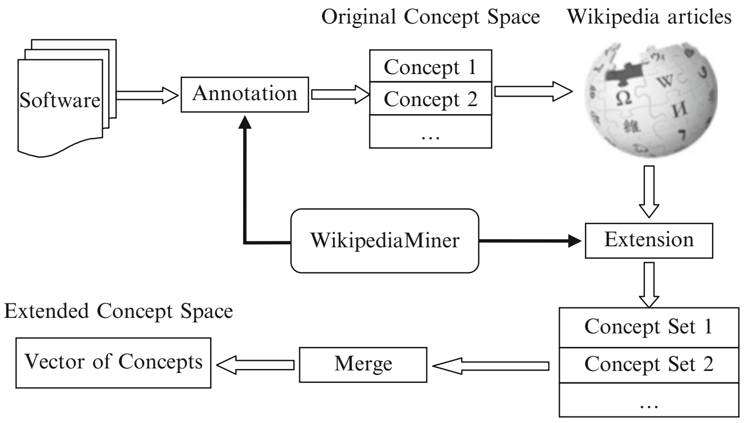

Fig. 1 A diagram of the overall process of ECSC

Table 1 The top 10 concepts most relevant to SVM and Bayesian probability

#	SVM	Bayesian probability
1	Statistical classification	Bayes' theorem
2	Decision tree learning	Frequency probability
3	Naïve Bayes classifier	Prior probability
4	Supervised learning	Decision theory
5	Perceptron	Principle of indifference
6	Linear classifier	Probability interpretations
7	Kernel methods	Statistical inference
8	Multiclass classification	Bayesian inference
9	Discriminative model	Frequentist inference
10	Document classification	Posterior probability

Table 2 The original and extended concepts most relevant to Weka

#	Original concepts	Extended concepts
1	Association rule learning	Apriori algorithm
2	Clustering	Principal component analysis
3	Feature selection	Cross-validation (statistics)
4	K-means clustering	K-medoids
5	Predictive modeling	Naïve Bayes
6	RapidMiner	Association rule learning
7	Boosting	Bootstrap aggregating
8	C4.5 algorithm	ID3 algorithm
9	Decision tree learning	Random forest
10	Statistical classification	Support vector machine

keyword and $v_{ij} = 0$ otherwise; for $W[P_{i,j}]$, its value was 0.5 for words in title and authors, 0.4 for words in the concept space, and 0.1 for other attributes.

$$\text{Rank}(j) = \sum_{i=1,j=1}^{i,j} v_{i,j} \times W\left[P_{i,j}\right] \quad (1)$$

Table 3 Comparison of the concepts of Weka and Mlpy

#	Weka	Mlpy
1	RapidMiner	Support vector machine
2	Predictive modeling	Least squares
3	Decision tree learning	Ridge regressions
4	C4.5 algorithm	Linear discriminant analysis
5	Feature selection	Perceptron
6	Clustering	Logistic regression
7	Association rule learning	Hierarchical clustering
8	Boosting	Partial least squares
9	Statistical classification	K-means
10	K-means clustering	Principal component analysis
11	ID3 algorithm	Fisher discriminant
12	Apriori algorithm	Regression
13	Bootstrap aggregating	Python
14	Random forest	Classification
15	K-medoids	Clustering
16	Support vector machine	Dimensionality reduction

3 Empirical Evaluation

We implemented our ECSC approach using an English Wikipedia dump[6] dated 12 March 2012. After parsing the Wikipedia XML dump, we obtained 16.0 GB of text articles. Upon removing narrow and overly specific concepts (those having fewer than 100 words and fewer than 5 in-links or out-links), 3,636,122 articles related to software were left. We processed the text of these articles by removing stop words and rare words, and stemming the remaining words, which yielded 39,524 distinct terms to be used for representing Wikipedia concepts as attribute vectors. The concept space contained over 20,000 concepts and 40,000 relatedness scores.

Totally over 3,000 software applications were collected from the Web, including more than 2,000 applications retrieved from software repositories and communities such as SourceForge[7] and Mloss[8] and around 1,000 applications crawled from various Web sites such as universities, research institutes, and personal blogs. Figure 2 shows the metadata of software crawled from the Web.

For evaluation purpose [8], we compared the results based on ECSC to results produced by human users. We used 40 software applications as the test samples and identified 20 most related software applications for each of them using three methods. The motivation was to investigate that whether the new concept space

[6]http://dumps.wikimedia.org/

[7]http://sourceforge.net/

[8]http://mloss.org/software/

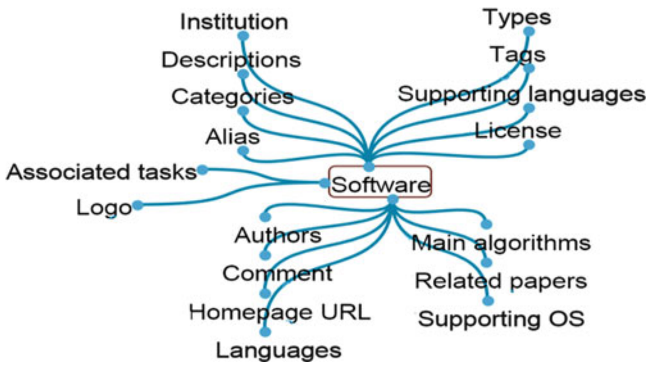

Fig. 2 The schema of software profile

based on ECSC can bring measurable benefits to the quality of search results. In other words, we were interested in finding out whether ECSC can help the search engine better understand the scope of software applications. With the lack of a well-accepted quantitative performance metric, we chose to use human results as the benchmark against which to evaluate different methods.

Let S1 be the set of software applications selected using the original software description while S2 be the set of software applications selected using ECSC. Furthermore, we asked a group of research students in our lab as volunteers to manually select a set of 20 software applications that they believed were most related to each test sample, referred to as S3.

Ideally, S1 and S2 should be similar to S3 (containing a large portion of shared software items), and in practice it would be interesting to compare the size of the intersection between S1 and S3 (the performance of the original method) with the size of the intersection between S2 and S3 (the performance of ECSC). Figure 3 shows that, over the 40 test samples, ECSC (solid line) produced consistently better results than the original method (dashed line), which confirms that the extended concept space can bring measurable benefit to software search services.

More specifically, ECSC demonstrated notable improvement in the correlation between computed software relatedness and human judgment, compared to the origin system: the average proportion of common software items using ECSC was 16.8/20 = 84%, while the average proportion of common software items using the origin method was 12.4/20 = 62%. Note that the performance had certain level of fluctuation over the set of test samples, which may be due to the insufficient amount of related information available in Wikipedia articles for some software applications. Also, different software applications may have quite different concept space sizes (both original and extended concepts), which may have some impact on the advantage of ECSC over the original method.

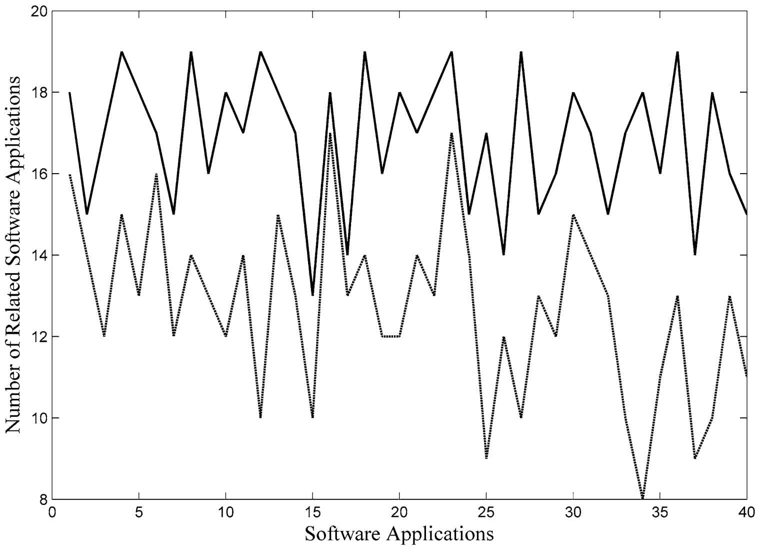

Fig. 3 The comparison between ECSC (*solid line*) and the original method (*dashed line*) over 40 software applications. The vertical axis shows the number of software applications returned that are shared with the results produced by human users

4 Related Work

SourceForge is a popular repository mainly for free or open source software, providing free services for controlling and managing software projects and limited search functions including keyword search for titles, category, license, programming languages, and operating systems. Github[9] is a platform similar to SourceForge with revision control systems for both paid private repositories and free open source projects. It also provides keyword search for titles and contents of code as well as category selection such as programming languages and developers.

Mloss (machine learning open source software) is dedicated to building a comprehensive open source machine learning environment and provides keyword search for titles with optional filters such as author, submitter, tag, license, programming language, and operating system. The major motivation of Mloss is that, given the large number of machine learning algorithms available in the literature, their implementations are often not open to the research community, resulting in low usability and weak interoperability. It is believed that publishing existing and freshly developed algorithm toolboxes along with short articles can be highly valuable to the development of machine learning and the general scientific communities.

[9]https://github.com/

Table 4 The category information of the concept of machine learning in Wikipedia

Machine learning	
• Applied machine learning	• Inductive logic programming
• Artificial intelligence conferences	• Machine learning algorithms
• Bayesian networks	• Markov models
• Classification algorithms	• Neural networks
• Cluster analysis	• Machine learning researchers
• Computational learning theory	• Support vector machines
• Data mining and machine learning software	• Kernel methods for machine learning
• Decision trees	• Latent variable models
• Dimension reduction	• Learning in computer vision
• Ensemble learning	• Log-linear models
• Evolutionary algorithms	• Loss functions

Understanding and incorporating the semantic information of software is critical for the effective search of software. For this purpose, Wikipedia has been widely used as the knowledge base to provide semantic information [9, 10]. For example, the titles of articles in Wikipedia are often treated as concepts and topics. Selected Wikipedia concepts can be augmented as extra text in clustering short texts to improve the accuracy [4]. Concepts derived from Wikipedia can be also used as a high-dimensional concept space for representing the semantic meaning of text [2].

Furthermore, each article in Wikipedia belongs to at least one category and there is structured information in Wikipedia such as namespace and category trees that can be used as taxonomy[10] (see Table 4 for an example). Category itself is also a reliable and useful source for topic discovery and relatedness computing [11]. For example, semantic distances can be defined as a function of the number of edges in the taxonomy along the path between conceptual nodes [12].

5 Conclusion

In this paper, we proposed a novel approach called ECSC to computing the semantic relatedness of software applications using extended concept space. With the aid of Wikipedia articles, which contain many concepts and interlinks of concepts, we calculated the relatedness of concepts using WLVM and selected a vector of concepts closely related to a given concept, effectively extending the original concept space of software applications.

Compared to other methods based on taxonomy and semantic Web technologies, our approach does not require building a domain knowledge repository by human experts, which can be expensive and difficult to maintain. Instead, our approach can

[10]http://en.wikipedia.org/wiki/Wikipedia:Categorization

take the advantage of the comprehensive and authoritative knowledge in Wikipedia accumulated through collaborative intellectual work.

Empirical evaluation on 40 test samples confirms that ECSC can consistently lead to substantial improvement in accuracy when searching for a set of software applications closely related to a given software application. Compared to the traditional method based on the original concept space (description of software), the search results using ECSC can be better correlated with human judgments. Furthermore, due to the explicit use of domain concepts, the search results are also easy to be understood by human users. In the future, we will consider incorporating other semantic techniques into software search as well as applying ECSC to other application domains.

Acknowledgments This work was supported by the National Natural Science Foundation of China (No. 60905030). The authors are also grateful to Prof. Juanzi Li for her kind help.

References

1. Waitelonis, J., Sack, H., Hercher, J., Kramer, Z.: Semantically enabled exploratory video search. In: 3rd International Semantic Search Workshop, Article No. 8 (2010)
2. Gabrilovich, E., Markovitch, S.: Computing semantic relatedness using wikipedia-based explicit semantic analysis. In: 20th International Joint Conference on Artificial Intelligence, pp. 1606–1611 (2007)
3. Coursey, K., Mihalcea, R.: Topic identification using wikipedia graph centrality. In: 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers, pp. 117–120 (2009)
4. Banerjee, S., Ramanathan, K., Gupta, A.: Clustering short texts using wikipedia. In: 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 787–788 (2007)
5. Yang, J., Han, J., Oh, I., Kwak, M.: Using wikipedia technology for topic maps design. In: 45th Annual Southeast Regional Conference, pp. 106–110 (2007)
6. Medelyan, O., Witten, I.H., Milne, D.: Topic indexing with wikipedia. In: AAAI Workshop on Wikipedia and Artificial Intelligence: An Evolving Synergy, pp. 19–24 (2008)
7. Milne, D.: Computing semantic relatedness using wikipedia link structure. In: New Zealand Computer Science Research Student Conference (2007)
8. Tumer, D., Shah, M.A., Bitirim, Y.: An empirical evaluation on semantic search performance of keyword-based and semantic search engines: Google, Yahoo, Msn and Hakia. In: Fourth International Conference on Internet Monitoring and Protection, pp. 51–55 (2009)
9. Völkel, M., Krötzsch, M., Vrandecic, D., Haller, H., Studer, R.: Semantic wikipedia. In: 15th International Conference on World Wide Web, pp. 585–594 (2006)
10. Kaptein, R., Serdyukov, P., De Vries, A., Kamps, J.: Entity ranking using wikipedia as a pivot. In: 19th ACM International Conference on Information and Knowledge Management, pp. 69–78 (2010)
11. Chernov, S., Iofciu, T., Nejdl, W., Zhou, X.: Extracting semantic relationships between wikipedia categories. In: First Workshop on Semantic Wikis – From Wiki to Semantics (2006)
12. Strube, M., Ponzetto, S.P.: WikiRelate! Computing semantic relatedness using wikipedia. In: 21st National Conference on Artificial Intelligence, pp. 1419–1424 (2006)