2015 IEEE/ACM 37th IEEE International Conference on Software Engineering

Poster: Software Development Risk Management: Using Machine Learning for Generating Risk Prompts

Harry Raymond Joseph
Department of Electrical Engineering
Indian Institute of Technology Madras
Chennai, India 600 036
Email: raymond.harry@tum.de

***Abstract*—Software risk management is a critical component of software development management. Due to the magnitude of potential losses, risk identification and mitigation early on become paramount. Lists containing hundreds of possible risk prompts are available both in academic literature as well as in practice. Given the large number of risks documented, scanning the lists for risks and pinning down relevant risks, though comprehensive, becomes impractical. In this work, a machine learning algorithm is developed to generate risk prompts, based on software project characteristics and other factors. The work also explores the utility of post-classification tagging of risks.**

***Index Terms*—Software risk, software management, machine learning, risk prompts.**

I. Introduction

This study aims to build a machine learning based risk prompting system that outputs the most relevant risk prompts based on specification, scenario and taxonomy tags for the software development project. The reduction in the number of prompts as against a 'keyword' search is as high as 50%.

Software risk management requires a multi-dimensional understanding of varied aspects ranging from critical process paths to macroeconomic market dynamics. Several lists are available that document varied risks that may arise out of a software development project. The lists usually comprise of a few hundreds of risk prompts. Examples that compile these lists include [1] and [2]. A comprehensive survey of the top 10 lists that compile software development risks is provided in [3]. A good starting point to looking at risks is to consider the 10 software project risk taxonomies introduced in [4]. More recent and commonly accepted risk 'dimensions' in literature, as in [5] are: team, organizational environment, requirements, planning - control, user and complexity.

Each of these risk dimensions have several risk prompts relating to them - for instance, the CMU-SEI 1993 list comprises of 200 questions based on 13 risk taxonomies [6]. While a less granular approach entails fewer prompts, there is most certainly a compromise on the value afforded by less granular lists. However, highly granular lists require a lot of time for analysis, thus ending up with little practical use.

Other less generic and more precise risk analysis approaches include scenario-based and specification-based approaches. These approaches are considered more time consuming and less scientific. However a risk anaysis approach that combines all 3 aspects - scenario, specification and taxonomy-based compilations is highly recommended [7]. The rest of this work is organized as follows: Section II presents the background for the study and the collation of data. Section III presents the machine learning classification algorithms employed, followed by section IV which makes a case for the utility of post-classification tagging in risk analysis. Section V presents the results of the algorithm and limitations of this work. Finally, Section VI concludes outlining the future scope of this work.

II. Background and Data Collation

The list of risk prompts were obtained by compiling unique risk prompts from several lists, including those in [3] along with the risk prompts in [7], and then granulating these prompts to reach higher levels of specificity. As pointed out earlier, about 433 risk prompts were compiled to make a highly granular list under 16 taxonomies.

40 advanced student volunteers in project management sorted the 433 risk prompts, and identified risks for 400 software projects successfully completed by Oracle. Note that the risk prompts were sorted under 16 taxonomies, independently. This approach, significantly reduces the task of risk identification for the volunteers, since a risk prompt under a certain taxonomy is considered independent of another risk prompt under another taxonomy.

Thanks to a well-organized Oracle product catalogue, available on [8], a web crawling program was able to provide specifications and possible scenarios in the development process, for these 400 software products. Among the specifications, 20 most representative, frequently occuring specifications were chosen as the list of possible specifications.

The output from the web crawler also yielded the product development scenarios of a larger deviating variety. With the constraint of reducing the list of possible scenarios for machine learning, 25 scenarios were chosen. Some scenarios, though less frequently occuring, were chosen in the interest of preserving representation across all the 400 Oracle products considered. Though this approach is less precise, the limitation

978-1-4799-1934-5/15 $31.00 © 2015 IEEE
DOI 10.1109/ICSE.2015.271

TABLE I
EXAMPLES OF TAXONOMIES, SPECIFICATIONS AND SCENARIOS

Taxonomy	Specification	Scenario
Planning	Network	Encryption
Complexity	Contractor	Prediction
Team	Device	Computation
User	Multi-User	Decision

TABLE II
RESULTS OF MACHINE LEARNING CLASSIFIER

Taxonomy	No. of ANNs	Risk Prompts	Keyword Search
1	6	40	83
2	5	39	103
3	5	46	100
4	3	23	45
5	7	56	132

is confined only to this specific study. Enhanced scenario data availability will ensure that the quality, precision and relevance of the risk prompts improve.

III. MACHINE LEARNING ALGORITHM

Developing the machine learning algorithm making the risk prompts based on scenario, taxonomy and specifications data is less burdensome but requires a sophisticated set of distributed machine learning jobs. Since independence of risk prompts across taxonomies was assumed, analysis of each of the taxonomies may thus be treated independent of each other. In addition, it may also be assumed that for each of the risk-prompts considered, inclusion of a risk prompt is independednt of inclusion of another risk prompt even if both prompts belong to the same taxonomy. These simplifications reduce the problem significantly, and we have several multilabel classification problems, which are resolved by using multiple multilabel artificial neural networks (ANNs) described in [9] and [10]. Basic details of the process are:

- Each classifier is a single hidden layer neural network, which outputs a probability on the relevance of a certain risk prompt ranging from 0 to 1.
- Even though the problem seems to suffer from the curse of dimensionality, relevant attributes are 'screened using' correlation measures.
- On an average, for each risk prompt, every neural network outputs a degree of relevance and has as its input: 5-10 scenarios and almost the same number of specifications.
- Each of the neural networks corresponding to a risk prompt, operates independently from the other.
- This is good scaling - 400 instances and 20 attributes.

IV. POST-CLASSIFICATION TAGGING

Post-classification tagging is an essential utility that risk prompting affords. Post-classification tagging provides actionable insights to systematically mitigate software development project risks. However, it is necessary to tag risks before the classification stage - this isn't an advantage arising out of the learning setting, but rather a data collation advantage obtained after appropriate risk classification.

An example of the utility of such post-classification tagging is the tagging of risks as systemic and idiosyncratic. In this study, all the 433 risk prompts have been classified into either of these two categories. This allows for efficient software pricing - systemic risks are transferred by including the monetary risk index for these risks in the project cost.

Another particularly advantageous use of post-classification tagging is prioritization. Risks may be defined ranging from high to low priority risks, that can provide a timeline for project managers to act.

V. RESULTS AND LIMITATIONS

The trained neural networks were tested against 40 products, and the corresponding manual risk prompts as classified by volunteers. The results for randomly chosen 5 products are provided in Table II. The results are compared against the performance of the 'keyword' relevant search - which simply returns risk prompts, based on the matching keywords in the specifications and scenarios of the software product. We find that number of risk prompts returned have reduced by at least 50% as against the keyword based search.

Several limitations are in order, particularly:

- Computationally inefficient - requires the use of several ANNs. A solution could be to use hierarchical methods.
- False negatives pose a real problem. One solution would be to set decision points below 0.5 - for instance prompts with degree of relevance exceeding 0.4 should be notified.

VI. FURTHER CONSIDERATIONS

There is considerable scope for future work, and elaborating on this research - possible points to probe further are:

- Machine learning approach-find a comprehensive dataset.
- Is the level of granularity of compiled list sufficient?
- Time complexity-quicker, yet more relevant risk prompts.

REFERENCES

[1] H. Barki, S. Rivard, et al., "Toward an assessment of software development risk," Journal of Management Information Systems, vol. 10, pp. 203-225, 1993.

[2] F. W. McFarlan, "Portfolio approach to information systems," Harvard Business Review, vol. 59, pp. 142-150, 1981.

[3] T. Arnuphaptrairong, "Top ten lists of software project risks: Evidence from the literature survey," Proceedings of the International MultiConference of Engineers and Computer Scientists, vol. 1, pp. 1-6, 2011.

[4] B. W. Boehm, "Software risk management: principles and practices," IEEE Software, vol. 8, pp. 3241, 1991.

[5] L. Wallace, M. Keil, et al., "Understanding software project risk: a cluster analysis," Journal of Information and Management, vol. 42, pp. 115-125, 2004.

[6] M. J. Carr, S. L. Konda, et al., "Taxonomy-based risk identification," Software Engineering Institute, Carnegie Mellon University, 1993.

[7] J. McCaffrey, "Analyzing project exposure and risk using PERIL," MSDN Magazine, 2009.

[8] Oracle, Oracle Products, See: http://www.oracle.com/us/products/product-list/products-a-z/index.html. Retrieved on: 13 January 2014.

[9] R. Grodzicki, J. Mandziuk, et al., "Improved multilabel classification with neural networks," Lecture Notes in Computer Science, vol.5199, pp. 409-416, 2008.

[10] M. L. Zhang and Z. H. Zhou, "Multi-label neural networks with applications to functional genomics and text categorization," IEEE Transactions on Knowledge and Data Engineering, vol. 18, pp. 1338-1351, 2006.