A Framework for Adoption of Machine Learning in Industry for Software Defect Prediction

Rakesh Rana[1], Miroslaw Staron[1], Jörgen Hansson[1], Martin Nilsson[2] and Wilhelm Meding[3]
[1]*Computer Science & Engineering, Chalmers, University of Gothenburg, Gothenburg, Sweden*
[2]*Volvo Car Group, Gothenburg, Sweden*
[3]*Ericsson, Gothenburg, Sweden*

Keywords: Machine Learning, Software Defect Prediction, Technology Acceptance, Adoption, Software Quality Acronyms Used — ML: Machine Learning, SDP: Software Defect Prediction, TAM: Technology Acceptance Model.

Abstract: Machine learning algorithms are increasingly being used in a variety of application domains including software engineering. While their practical value have been outlined, demonstrated and highlighted in number of existing studies, their adoption in industry is still not widespread. The evaluations of machine learning algorithms in literature seem to focus on few attributes and mainly on predictive accuracy. On the other hand the decision space for adoption or acceptance of machine learning algorithms in industry encompasses much more factors. Companies looking to adopt such techniques want to know where such algorithms are most useful, if the new methods are reliable and cost effective. Further questions such as how much would it cost to setup, run and maintain systems based on such techniques are currently not fully investigated in the industry or in academia leading to difficulties in assessing the business case for adoption of these techniques in industry. In this paper we argue for the need of framework for adoption of machine learning in industry. We develop a framework for factors and attributes that contribute towards the decision of adoption of machine learning techniques in industry for the purpose of software defect predictions. The framework is developed in close collaboration within industry and thus provides useful insight for industry itself, academia and suppliers of tools and services.

1 INTRODUCTION

Testing is an essential activity in software engineering (Bertolino, 2007), but also one of the most expensive phase within software development life cycle with some estimates approximating it to consume about 50% of time and resources (Harrold, 2000). Software Defect Prediction (SDP) offers one possible way to make software testing more effective by making it possible to optimize test resource allocation, i.e. distributing more effort to parts (files/modules) that are predicted to be more prone to defects. The importance of such predictions is further substantiated by previous research suggesting applicability of 80:20 rule to software defects (that is approximately 20% of software files are responsible for 80% of errors and cost of rework) (Boehm, 1987) (Güneş Koru and Tian, 2003).

Different methods for defect prediction have been evaluated and used; these can broadly be classified as traditional (using expert opinions and regression based approaches) and those based on machine learning techniques. Methods based on machine learning offer addition advantage with their ability to improve their performance through experience (as more data is made available over time). Despite the importance of predicting defects in a software project and demonstrations that SDP using ML techniques is not too difficult to apply in practice (Menzies et al., 2003), their adoption and application by practitioners in industry has been limited which is apparent from the lack of published experience reports. Adoption of any complex method/technology is dependent on several dimensions (Legris et al., 2003), but most of the earlier studies in SDP have focused mainly on the aspect of predictive accuracy. In this paper we argue that our lack of understanding of other factors relevant to industrial practitioners is a major reason for low adoption of ML techniques for SDP in industry.

Rana R., Staron M., Hansson J., Nilsson M. and Meding W..
A Framework for Adoption of Machine Learning in Industry for Software Defect Prediction.
DOI: 10.5220/0005099303830392
In *Proceedings of the 9th International Conference on Software Engineering and Applications* (ICSOFT-EA-2014), pages 383-392
ISBN: 978-989-758-036-9

Based on the technology acceptance model (TAM) and technology adoption frameworks we develop a framework for explaining the adoption of ML for SDP in industry. TAM intends to explain why users' belief and their attitudes towards a technology affect their acceptance or rejection of the information-communication technology. While TAM is parsimonious and theoretically justified model to explain information technology adoption (Van der Heijden, 2003), to use this model for a specific technology requires identification of detailed attributes specific to the given technology and context which collectively explain the belief and attitude of uses towards the given technology. The research question we address in this paper is:

"How can we use the technology acceptance and adoption models for developing framework for ML adoption in industry and how to adapt it for software defect prediction?"

2 BACKGROUND AND RELATED WORK

2.1 Software Defect Prediction Using Tradition Approaches

Traditional methods used for software defect prediction and risk assessment can be broadly categorized under:

- Expert Opinions
- Analogy Based Predictions
- Regression Based Approaches

Statistical approaches based on regression have also been used for the task of defect prediction. The dependent (or outcome) variable could be binary (defective or not defective) as in logistic regression or the model could be built to predict the number of expected defects as in case of multiple linear regression. Logistic regression has been applied in Khoshgoftaar and Allen (Khoshgoftaar and Allen, 1999) for classifying modules as fault-prone or not. Zimmermann, Premraj and Zeller (Zimmermann et al., 2007) also applied Logistic regression to classify file/packages in Eclipse project as defect prone (has defect Vs. not has defect) . Multiple linear regression is used to model software changes (Khoshgoftaar et al., 1993) as a function of a set of software complexity metrics. Linear regression was also used by Khoshgoftaar et al. (Khoshgoftaar et al., 1992) for predicting program faults in two subsystems of a general-purpose operating system, where they also evaluated different fitting criteria's (namely Least Squares, Least Absolute Value, Relative Least Squares and Minimum Relative Error).

2.2 Software Defect Prediction Using ML Techniques

Broad types of Machine Learning (ML) techniques used for software defect prediction:

- Decision Trees (DTs)
- Support Vector Machines (SVMs)
- Artificial Neural Networks (ANNs)
- Bayesian Belief Networks (BNNs)

Machine learning algorithms can also be used to model the software defect prediction as a classification problem as in case of DTs and SVMs where the class variable can take two values (defective or not defective). Or the problem can be modelled to predict expected number of defects in a software module/system using different code and change metrics. ML techniques for pattern recognition for e.g. ANNs and BNNs can be used to accomplish such tasks.

Number of various classification models including DTs and SVMs have been evaluated and compared in (Lessmann et al., 2008). Iker Gondra (Gondra, 2008) applied machine learning algorithms to predict the fault proneness and compared between the ANNs and SVMs and found that if fault proneness is modelled as classification task, SVMs performs better than the ANNs.

Table 1 provides an overview of some of the important ML techniques that can be applied for SDP and lists their main advantages and limitations. For details on ML techniques applicable in software engineering domain; readers are referred to work by Zhang and Tsai (Zhang and Tsai, 2003).

2.3 Technology Adoption Framework

According to Attewell (Attewell, 1992) adoption of complex technology is not an event, but resembles knowledge acquisition over time, the perspective is applicable where new innovation/technique is (Attewell, 1992):

- Abstract and have demanding scientific base,Fragile in sense of consistency, i.e. do not always perform as expected,
- Difficult to try in a meaningful way, and
- Unpackaged, i.e. adopters cannot pick a tool out of shelve and use it as a black box model, but instead need to acquire broad tacit knowledge and procedural know-how.

Table 1: Overview of ML techniques used for software defect prediction.

Algorithm Type	**DTs**
Domain Knowledge	Not Required
Training Data	Adequate data needed to avoid over-fitting.
Advantages	Robust to noisy data; Missing values tolerated; Capable of learning disjunctive expressions.
Disadvantages	Prone to over-fitting.
Algorithm Type	**SVMs**
Domain knowledge	Not Required
Training Data	Adequate data needed for training.
Advantages	Effective for high dimensional spaces, is memory efficient and is versatile as it can take different kernel functions as decision function
Disadvantages	SVMs are likely to give low performance if number of features is much higher than the number of samples
Algorithm Type	**ANNs**
Domain knowledge	Not Required
Training Data	Adequate data needed for training.
Advantages	Able to learn non-linear and complex functions; Robust to errors in training data.
Disadvantages	Slow training and convergent process; Prone to over-fitting; Results difficult to interpret.
Algorithm Type	**BNNs**
Domain Knowledge	Not Required
Training Data	Required for estimate the prior probabilities.
Advantages	Able to give probabilistic predictions; Useful for knowledge discovery; Can be used very early in the development lifecycle
Disadvantages	Requires estimation of many prior probabilities that can be very large for big models; computationally expensive; requires domain expertise for building the network.

Characteristics of ML based techniques fits well to most above point and thus can be classed as complex technology/techniques. Further according to the Theory of Reasoned Action (TRA) (Ajzen and Fishbein, 1980), the intention of adoption of behaviour or technology is based on the beliefs about the consequences of adoption. The theory have been used to build Technology Acceptance Model (TAM) by Davis (Davis Jr, 1986), an overview of model is presented in Figure 1. TAM postulates that a users' adoption intention and the actual usage of information technology is determined by two critical factors, the perceived usefulness and perceived ease of use. Perceived usefulness is defined as the degree to which a user believes that using a particular system would enhance his/her job performance, while perceived ease of use is the degree to which the user believes that using the system would be effort free (Van der Heijden, 2003).

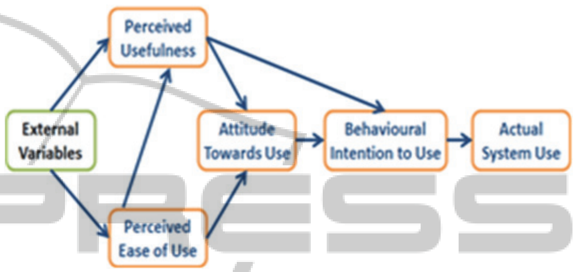

Figure 1: Overview of Original Technology Acceptance Model (Legris et al., 2003).

In this study we are focused on technology adoption decisions, thus the model we use for our framework is based on the revised version of original TAM model (Pijpers et al., 2001), the postulation of revised model is that potential users of a technology actively evaluate the usefulness and ease of use of given technology in their decision making process (Yang, 2005). Our position in this paper is similar:

We contend that applying technology adoption framework to ML techniques use in SDP is needed to better understand the needs of industry - which will help accelerate the technology transfer and adoption process of these techniques.

Technology adoption framework by Tornatzky et al. (Tornatzky et al., 1990) also provide a model of adoption that has been applied widely. According to the framework, there are three elements which influence the innovation adoption process:

1. The external environmental context,
2. The technological context, and
3. The organizational context.

Chau and Tam (Chau and Tam, 1997) used the framework to model the factors affecting adoption of open systems in the Information Science (IS). We adapt their framework in conjunction with the Technology Acceptance Model (TAM) to model the factors affecting adoption of ML in industry.

3 STUDY DESIGN

The research process for development and quantitative validation of adoption framework for ML techniques in industry is shown in Figure 2. The focus of this paper is Stage-1, where the center of attention has been to develop the general adoption framework for machine learning techniques and demonstrate how the model can be adapted for the specific case of software defect prediction (SDP).

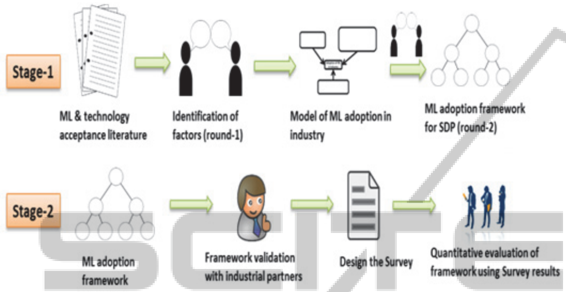

Figure 2: Research process overview.

Literature Review: To capture the factors that affect the adoption of ML techniques in industry we searched for likely factors mentioned in software engineering, machine learning and technology adoption literature. A list of factors deemed potentially relevant for industry was compiled which was used for discussions with the industrial practitioners. The application area we concentrated on is defect prediction in software system/projects.

Interviews: Semi-structured interviews were conducted with industrial practitioners to first evaluate which factors are relevant for ML adoption in industry. In the next round the same interviewees helped adapt this general model for the case of software defect prediction.

In total four managers from two large companies with significant focus on software development were interviewed consequently in two rounds. The companies included in the study are:

- **Volvo Car Group (VCG):** A company from the automotive domain, and
- **Ericsson:** A company from the telecom domain

The divisions we interacted with have one thing in common, they have not yet adopted machine learning as their main method/technique for predicting software defects, but they are evaluating it as a possible technique to compliment the current software defect measurement/prediction systems in place. The interviewees included,

• Manager at Volvo Cars Group within the department responsible for integrating software sourced from different teams and suppliers, the manager has more than 20 years of experience working with software development and testing. Ensuring safety and quality of software developed is a major responsibility in this job role.

• Team leader at Volvo Car Group responsible for collection, analysis and reporting of project status with regard to software defects and their predictions, the team leader has more than three decades of experience in various roles at the company.

• A senior quality manager at Ericsson whose experience with software (mainly within quality assurance) spans more than three decades, and

• Team leader of metrics team at Ericsson; metrics team is a unit at Ericsson that provides the measurement systems for various purposes including software defect measurement, monitoring and prediction systems within the organization.

The main focus in the first round of interviews is to identify the factors relevant with regard to technology adoption/acceptance decisions (to build a general framework of ML adoption in industry). While the second round of interviews were focused on identification of relevant attributes for each factor in the specific context of software defect prediction.

4 FRAMEWORK FOR ADOPTION OF ML TECHNIQUES IN INDUSTRY

It is important to note that for any organization at any given point in time, the trade-off analysis is not between adopting or not adopting a new technology/process (as in case of ML techniques); the trade-off is between adopting it now or deferring that decision until a later date. This distinction is important as the factors that affect the adoption are not only specifically related to direct advantages and limitation of given technology/process, but also organizational and environmental at a given point in time. In this context, nine important factors that affect the adoption of ML techniques were identified; these can be grouped into three categories according to the framework by Tornatzky (Tornatzky et al., 1990). The framework for adoption of ML in industry is presented in Fig 3.

In Fig 3 (+) and (-) signs denote the possibility of positive/negative relationship with medium strength between a given factor and probability of adoption of ML. A double (++/--) indicate a strong relationship; the strength of relationship can be tested by setting a stricter significance level during quantitative evaluation (for e.g. alpha value of 0.1 for +/- and 0.05 for ++/--). Accordingly hypotheses for each factor can be formulated which can be tested quantitatively from a survey. We provide a couple of examples of null hypothesis that can be quantitatively tested:

H1: *Higher levels of perceived benefits of adopting ML techniques will strongly (and positively) affect the likelihood of their adoption.*

H2: *Higher levels of perceived barriers of adopting ML techniques will strongly (and negatively) affect the likelihood of their adoption.*

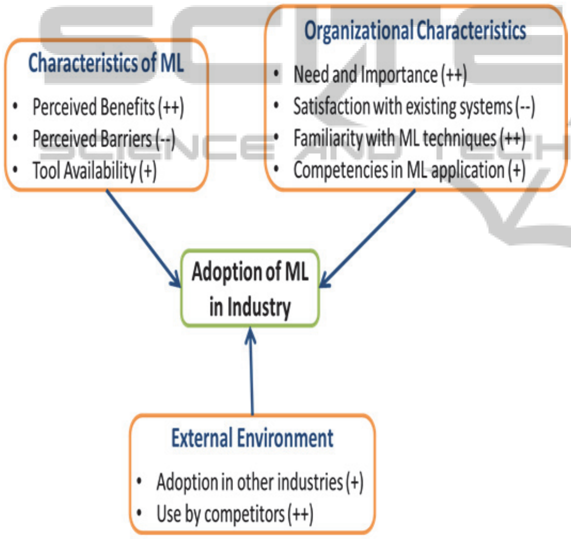

Figure 3: A Model for ML adoption in Industry.

5 ADAPTATION OF ML ADOPTION FRAMEWORK FOR SDP

We adapt the general framework for ML adoption in industry (Fig 3) to the specific problem of software defect prediction.

5.1 Characteristics of Machine Learning

Adoption of any new technology or process change is heavily dependent upon the characteristics of technology/innovation. Factors affecting cost-benefit trade-off of adoption are some of the critical factors in decisions of adoption. The relevant attributes that affect the acceptance of ML for software defect predictions are presented in Figure 4.

Perceived Benefits: one of the most critical factors in adopting ML techniques in industry are the perceived benefits of these techniques for a given organizations specific context. The keywords here are perceived and context. While the actual benefits, an organization can achieve by adopting a new innovation/technology is important in long run, at a given point in time what affects an organizations decision to adopt a new specific technology/innovation is its perception.

When it comes to SDP, the perceived benefits of using ML approaches as expressed in previous studies evaluating ML techniques for SDP and opinions expressed by the interviewees of this study are ability of ML based algorithms to:

- Provide higher prediction accuracy (high probability of detection and low probability of false alarm) (Gondra, 2008).
- Be highly automated, i.e. most aspects of system including data collection to visualization of results can be done using smart algorithms mining and analyzing data autonomously from the multiple local databases (Zhang and Zaki, 2006) with minimal human intervention.
- It is perceived that ML techniques can handle large data; in fact ML methods are expected to improve their performance as more data is made available over time (Zhang and Tsai, 2003).
- Another important expectation with techniques applied to predicting software defects is that these techniques are capable of identifying new patterns in data thus providing new insights from the data itself. This offers possibility to use large historical data to discover regularities and use them to improve future decisions (Mitchell, 1999). New insights can be generated using large data by employing specific ML techniques such as causal modeling for example by using Bayesian Networks to model causal networks and deduct probabilistic relationships.
- Given the self-adaptive nature, using ML techniques is also perceived to be low on maintenance activities.

Perceived Barriers: On the other hand perceived barriers negatively affect the adoption/acceptance of ML techniques. For software defect predictions, some of the common perceived barriers are:

- Steep learning curve – According to Edmondson et al. (Edmondson et al., 2003), users of new innovation/technology need to understand it well before they can put it into productive use. Their study also suggests that when tacit knowledge is needed, new technologies may fail in market even when their advantages have been proven.

  For example in case of SDP, when using classification or pattern recognition, selecting the set of attributes (inputs) that give optimal results is very much based on domain experience and experience of using ML based techniques which is difficult to document/codify explicitly for new users.

- Lack of trust – stakeholders in software projects who are used to traditional approaches of predicting defects (such as expert opinions) do not generally trust the algorithms to outperform expert based predictions.

- For software projects, in general and in particular for safety and business critical software products, the penalty for mis-prediction is an important barrier. The severity of mis-prediction is correlated to importance of information need and actions it can trigger. For example a prediction model that falsely predicts 20% of software modules as defect prone (compared to actual 10%) may lead to review of 10% modules which was unnecessary and results in resource allocation which is not optimal.

  As traditional methods have been used for comparatively longer time, their levels of (un)certainty are known – which is not the case with ML techniques. To overcome this barrier we recommend that in the initial phase of adoption of machine learning techniques, these should be using alongside the traditional methods to validate their usefulness and predictive accuracy in practice. This provides the comparisons industrial practitioners want to see before trust in new techniques begins to build up over such trial periods.

- Given that most practical aspects can be affected by wide range of factors; techniques based on ML approaches usually do not take into account all of these. Human factors such as differences in productivity, people getting sick or motivation level of employees are hard to measure and account for in algorithmic models for SDP and thus a source of error in such techniques.

- Uncertainty regarding generalizability of ML over projects. The perception is that while ML techniques (used for classification and pattern recognition) work well in recognizing existing patterns in the data, but their performance degrades for patterns that are unseen before.

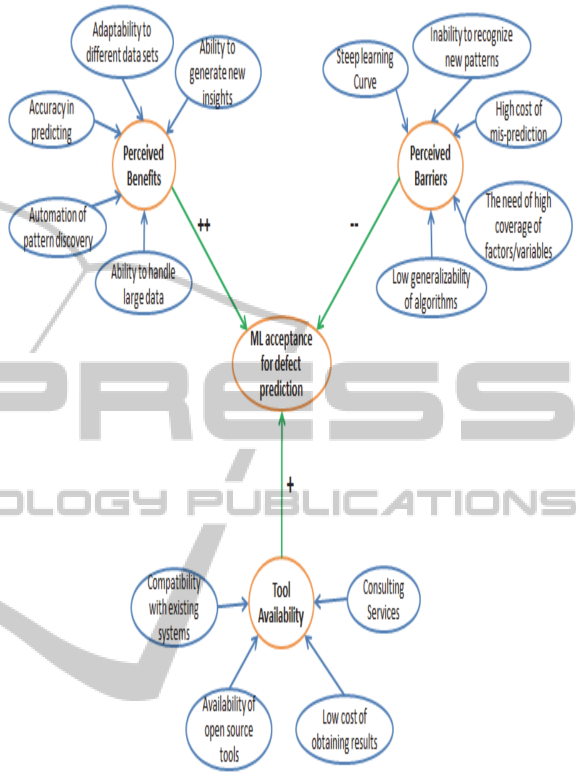

Figure 4: Overview of attributes relevant to ML characteristics that affects its acceptance for SDP.

Availability of Tool and Support is expected to increase the acceptance of ML in industry (Sonnenburg et al., 2007). Some of the attributes related to this factor are - if the available tools are open source or proprietary, how much support is available and how much they cost. Others include if the given tool is compatible with existing measurement systems and in-house competences with respect to its usage. Consulting services can also help specific companies to get started with new approaches that they do not have enough experience with - thus helping acceptance of new techniques and tools in industry.

A number of packages implementing ML algorithms are available for e.g. Netlab, Spider and BNT for Matlab; Nodelib, Torch for C++; and CREST for python. Commercial (e.g. Ayasdi, NeuroSolutions etc.) and open source tools (e.g. Weka, KNIME etc.) are also available with GUI. While availability of such tools is likely to increase the adoption of ML in industry, other attributes such as support and consulting services is also important in determining the level and speed with which ML is adopted in the industry.

One possible way of enhancing adoption through tool and support availability is by making available problem specific customized solutions for highly relevant industrial problems such as SDP. Other activities that can potentially accelerate the adoption process is integration of ML based algorithms in existing software packages widely used within industry, for e.g. Microsoft Neural Network algorithm available for SQL Server 2012.

5.2 Organizational Characteristics

Need and Importance: The higher the need and importance of given information is in an organization, the higher is the likelihood for adopting new techniques to satisfy this information need.

To improve on the accuracy and reliability for such measures, new approaches that offer higher accuracy and reliability are more likely to be adopted. Zhang and Tsai (Zhang and Tsai, 2003) provides a good overview of applications of ML in software engineering domain which outlines different information needs within this domain. Examples of information need specific to software defect predictions are:

- Predicting software quality (identification of high-risk, or fault-prone components)
- Predicting software reliability
- Predicating expected number of defects
- Predicting maintenance task effort
- Predicting software release timings

Factors such as how satisfied a company is with its existing defect prediction systems, their familiarity with machine learning techniques and in-house competences are also important for explaining acceptance and adoption of ML for SDP within a company. A model of attributes that contribute to these factors is presented in Figure 5.

Satisfaction with Existing Systems: the motivation for change (adoption of new approaches) is strongly connected to given organizations satisfaction with its current measurement/analysis systems. If a company is well satisfied with accuracy and efficiency of existing methods it is unlikely to invest significant amount of cost, resources and learning on new approaches. In case of software defect prediction, attributes relevant to satisfaction with existing systems are:

- If or not the existing system satisfies the information need of stakeholders involved in the project.
- Does existing system allow stakeholders to effectively and efficiently visualize the trend over time and let them compare current projects with similar historical projects data.

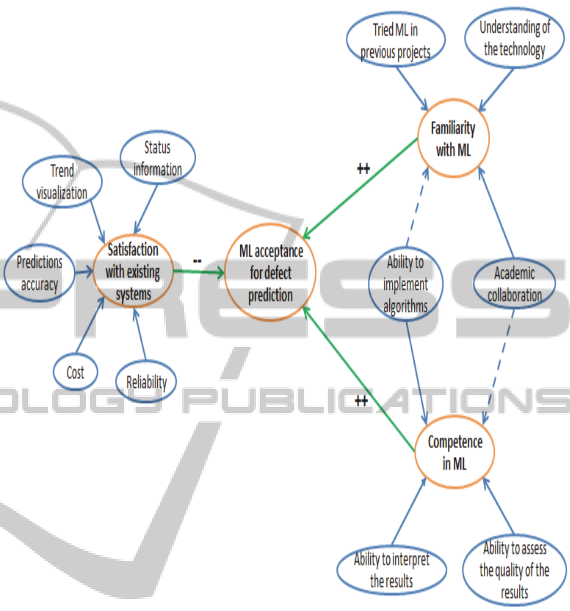

Figure 5: Overview of attributes relevant to organizational characteristics that affects its acceptance for SDP.

The reliability and cost also plays important role in determining the level of satisfaction with existing defect management and prediction systems within software development organizations.

Familiarity and Competence with ML Techniques: organizations familiar with approaches of machine learning though their workforce or collaborations with academia will have better understanding of advantages and limitations of such approaches. These organizations will also be more informed about practical applicability of these techniques and thus in a position where they can identify and assess areas where the benefits of using ML techniques outweigh the barriers – therefore organizations that are familiar with such methods are strongly likely to adopt these methods.

Attewell (Attewell, 1992) proposes that "*firms delay in-house adoption of complex technology until they obtain sufficient technical know-how to implement and operate is successfully*"

Almost all mature organizations engaged in developing software generally collect, store and analyze their product and process related data. Given that such data is available in large quantities (within the organizations), an organization with good competences/skills in machine learning are more likely to try ML techniques on their data and eventually adopt it on larger scales.

The main challenge in this context is unavailability of structured data. Much of the data generated within an organization is in form of unstructured text (e.g. software requirements, defect reports, customer feedback written in textual form). On the other hand most ML algorithms require inputs in numeric or categorical form which presents challenge in using such data in practice. Developments in field of Natural Language Processing (NLP) are already addressing these challenges and advances in such areas are likely to increase the adoption of ML based techniques for SDP.

5.3 External Environment

ML techniques, if adopted in different industries signals their applicability in practice, although this is not expected to be a strong factor deriving adoption in other industries – it is likely to affect positively the probability of adoption.

A similar but stronger factor for adoption of new technology/approaches such as ML in a given company is likely to be the information whether or not any of the competing companies are using such techniques. The motivation behind this factor is simple - every organization in a given domain intends to be at the forefront of technology or process knowledge. The adoption of a particular technique/process by a competitor is a strong signal that given technique could have potential benefits; this can potentially motivate the need for evaluation of such methods within the given organization.

6 HOW TO USE THE FRAMEWORK

Over the years companies have begun capturing huge volumes of data about their products, consumers and operations (Mitchell, 1999). ML offers new tools that can use this data to recognize patterns and provide useful insights hidden within these huge volumes of data.

6.1 Setting the Research Direction

The research in software defect predictions has been mainly focused on evaluating and highlighting the predictive accuracy of ML techniques and in some cases comparing it to traditional methods. On the other hand the adoption framework indicates that not only predictive accuracy, but attributes such as cost, reliability and generalizability are also important for adoption decisions.

Therefore the technology adoption framework, such as one proposed here, can be useful to guide future research directions by helping to identify which factors are relevant for industrial adoption, but currently unaddressed in terms of their scientific evaluation.

6.2 Evaluating Specific ML Techniques by a given Company

Technology acceptance/adoption frameworks enhance our understanding of which factors affect the end users decision to adopt a given technology/innovation. Although these factors do play a role to varying degree when companies evaluate their decision to adopt or delay the adoption of such techniques, the lack of a framework can lead to sub-optimal decisions. Without a guiding framework there is high probability that effect of some detailed attributes that affect the overall usefulness is missed. The severity of problem is greater when comparisons are made between two or more techniques or tools where it is likely that evaluation would focus only on small set of attributes which does not provide the full picture.

In such cases, the adoption framework can be used as a guide so that all important factors and associated attributes are covered when considering adoption of new techniques or tools or even as a checklist to make such assessment and comparison between two or more techniques/tools using Likert-type scale for evaluation. To provide an example, Table 2 shows a checklist to compare a ML based technique against existing system for SDP and Table 3 show potential use of similar checklist for comparison of two competing tools. Industrial practitioners can use such checklists to make informed decision with regard to adoption of these techniques and for effective comparison between tools.

The technology adoption framework also help companies to reflect upon their strengths with respect to given technology and areas of potential improvement. Such analysis is useful to identify areas where training and competence build-up would be advantageous. For example in SDP, if a company identifies that the in-house competence for implementing and maintaining ML based system would benefit a specific business unit within the organization, necessary training and or recruitment targeting those specific skills could be quickly arranged, thus improvising the long term competitiveness of the company.

Table 2: Example of how comparative checklist can be used to evaluate new technique for SDP.

Attribute	Existing Method	New ML based technique
Predictive Accuracy	Good	Very Good
Auto data acquisition	Yes	Yes
Report generation	Yes, word document	Yes, web based
Can handle multiple projects	No	Yes
Generate causal maps	No	Yes
Running time (typical project)	15min	30min
Cost of license (tool)	None	$ 20000/ license
Maintenance cost (estimate)	$ 2000 pa	$ 7000 pa
…		

6.3 Improvising the Tool and Services by Vendors

Technology adoption framework is also useful for tool vendors who can use the information in multiple ways, to:

- Prioritize feature introduction, and
- Effective marketing of their tools and services

Tools based on emerging technologies/techniques usually provide new functionality not available in old well established tools, but at the same time they are not mature and need to constantly evolve to engage and acquire new customers. Understanding clearly which attributes are key for adoption decision help these tool vendors to prioritize the features they implement and deliver to their customers. For example, a vendor with Tool X for SDP which at a given time do not outperform existing tools on predictive accuracy; finds out that running and maintenance costs are important attributes in adoption decisions - may use this information to strategically decide to develop a light version of tool which demands low running and maintenance costs.

Table 3: Example of how adoption framework can be used to compare between two new tools/services.

Attribute	Tool A	Tool B
Predictive Accuracy	85%	82%
Auto data acquisition	Yes	Yes
Report generation	Yes, web based	Yes, multiple format
Can handle multiple projects	Yes	Yes
Generate causal maps	Yes, Non-Interactive	Yes, Interactive
Running time (typical project)	30min	40min
Cost of license (tool)	$ 20000/ license	$ 35000/ license
Maintenance cost (estimate)	$ 7000 pa	$ 9000 pa
…		

Understanding of which attributes play a key role in adoption decisions also help tool and service vendors to make their marketing more effective. Vendors may choose to highlight how they provide value to their customers on the key attributes industry is looking for when considering adopting a new technology based product or services. This accelerates the adoption and acceptance of new techniques within the industry.

7 CONCLUSIONS AND FUTURE WORK

Large and constantly growing amount of data is now available within organizations that can be used for gaining useful insights to improvise process, products and services. Machine learning techniques have high potential to aid companies in this purpose. Despite demonstration of usefulness of such techniques in academia and availability of tools, the adoption of these techniques in industry currently is far from optimal. Our position in this paper has been that for accelerating the adoption of ML based techniques in industry, we need to enhance our understanding of information needs of industry in this respect. Technology acceptance model offer cost effective approach to meet this purpose.

In this paper we developed a framework for the adoption of ML techniques in industry. The framework is developed with its basis on previous research on technology adoption and technology acceptance models. We also adapted the framework to the specific problem of software defect predictions and highlighted that while adoption decisions are multi-dimensional, current research studies have mainly focused on few of these attributes. We contend that elevating our understanding of factors and attributes relevant for industrial practitioners will help companies, researchers and tool vendors to meet the specific information needs.

In future work we plan to quantitatively evaluate the effect size of important attributes towards ML adoption decision using large scale survey of companies that have already adopted ML techniques and ones that are yet to embrace them. Research with regard to which factors are important for industry and evaluative studies of ML based techniques/tools on these factors can complement the existing and on-going work on establishing the characteristics of ML techniques and thus contribute toward their adoption in industry and society.

ACKNOWLEDGEMENTS

The research presented here is done under the VISEE project which is funded by Vinnova and Volvo Cars jointly under the FFI programme (VISEE, Project No: DIARIENR: 2011-04438).

REFERENCES

Ajzen, I. & Fishbein, M., 1980. Understanding attitudes and predicting social behaviour. Available at: http://www.citeulike.org/group/38/article/235626.

Attewell, P., 1992. Technology diffusion and organizational learning: The case of business computing. *Organization Science*, 3(1), pp.1–19.

Bertolino, A., 2007. Software testing research: Achievements, challenges, dreams. In *2007 Future of Software Engineering*. IEEE Computer Society, pp. 85–103.

Boehm, B., 1987. *Industrial software metrics top 10 list*, IEEE Computer Soc 10662 Los Vaqueros Circle, Po Box 3014, Los Alamitos, Ca 90720-1264.

Chau, P.Y. & Tam, K.Y., 1997. Factors Affecting the Adoption of Open Systems: An Exploratory Study. *Mis Quarterly*, 21(1).

Davis Jr, F.D., 1986. *A technology acceptance model for empirically testing new end-user information systems: Theory and results*. Massachusetts Institute of Technology.

Edmondson, A.C. et al., 2003. Learning how and learning what: Effects of tacit and codified knowledge on performance improvement following technology adoption. *Decision Sciences*, 34(2), pp.197–224.

Gondra, I., 2008. Applying machine learning to software fault-proneness prediction. *Journal of Systems and Software*, 81(2), pp.186–195.

Güneş Koru, A. & Tian, J., 2003. An empirical comparison and characterization of high defect and high complexity modules. *Journal of Systems and Software*, 67(3), pp.153–163.

Harrold, M.J., 2000. Testing: a roadmap. In *Proceedings of the Conference on the Future of Software Engineering*. ACM, pp. 61–72.

Van der Heijden, H., 2003. Factors influencing the usage of websites: the case of a generic portal in The Netherlands. *Information & Management*, 40(6), pp.541–549.

Khoshgoftaar, T.M. et al., 1992. Predictive modeling techniques of software quality from software measures. *Software Engineering, IEEE Transactions on*, 18(11), pp.979–987.

Khoshgoftaar, T.M. & Allen, E.B., 1999. Logistic regression modeling of software quality. *International Journal of Reliability, Quality and Safety Engineering*, 6(04), pp.303–317.

Khoshgoftaar, T.M., Munson, J.C. & Lanning, D.L., 1993. A comparative study of predictive models for program changes during system testing and maintenance. In *Software Maintenance, 1993. CSM-93, Proceedings., Conference on*. IEEE, pp. 72–79.

Legris, P., Ingham, J. & Collerette, P., 2003. Why do people use information technology? A critical review of the technology acceptance model. *Information & management*, 40(3), pp.191–204.

Lessmann, S. et al., 2008. Benchmarking Classification Models for Software Defect Prediction: A Proposed Framework and Novel Findings. *IEEE Transactions on Software Engineering*, 34(4), pp.485–496.

Menzies, T. et al., 2003. How simple is software defect detection. *Submitted to the Emprical Software Engineering Journal*.

Mitchell, T.M., 1999. Machine learning and data mining. *Communications of the ACM*, 42(11), pp.30–36.

Pijpers, G.G. et al., 2001. Senior executives' use of information technology. *Information and Software Technology*, 43(15), pp.959–971.

Sonnenburg, S. et al., 2007. The need for open source software in machine learning. *Journal of Machine Learning Research*, 8(10).

Tornatzky, L.G., Fleischer, M. & Chakrabarti, A.K., 1990. Processes of technological innovation.

Yang, K.C., 2005. Exploring factors affecting the adoption of mobile commerce in Singapore. *Telematics and informatics*, 22(3), pp.257–277.

Zhang, D. & Tsai, J.J., 2003. Machine learning and software engineering. *Software Quality Journal*, 11(2), pp.87–119.

Zhang, S. & Zaki, M.J., 2006. Mining multiple data sources: local pattern analysis. *Data Mining and Knowledge Discovery*, 12(2-3), pp.121–125.

Zimmermann, T., Premraj, R. & Zeller, A., 2007. Predicting defects for eclipse. In *Predictor Models in Software Engineering, 2007. PROMISE'07: ICSE Workshops 2007. International Workshop on*. IEEE, pp. 9–9.