Literature Reviews on Applying Artificial Intelligence/Machine Learning to Software Engineering Research Problems: Preliminary

Pornsiri Muenchaisri
Department of Computer Engineering, Faculty of Engineering,
Chulalongkorn University, Bangkok, Thailand
Pornsiri.mu@chula.ac.th

Abstract

This paper is aimed to explore the application of Artificial Intelligence/Machine Learning (AI/ML) to software engineering research problems. Which activities of software engineering use AI/ML the most for solving research problems? The scope of the paper is to preliminary review research papers published in Asia-Pacific Software Engineering Conference 2018 (APSEC 2018) proceedings and researches conducted at the Department of Computer Engineering, Chulalongkorn University (CPCU). The author manually reviews papers with some keywords such as machine learning, neural network, and natural language processing. The result shows that machine learning is used in coding and software quality improvement activities more than other activities.

1 Introduction

Software Engineering (SE) is a step-by-step approach to gather and analyze requirements, design, develop, and test a software effectively and efficiently. Each activity can be accomplished with suitable techniques and methods. For example, software requirements can be obtained from interviewing and joint application development method which are performed manually. After requirements are gathered, requirements may be categorized. An automatic tool may be needed for requirement categorization. The research problem may involve finding an approach that can classify and categorize each requirement into appropriate group. Methods for creating functional models from requirements automatically may be needed. Design defect detection may be predicted before software is implemented. Automatically generating test cases from requirements is also possible.

Artificial Intelligence (AI) emphasizes the development of software which can perform tasks like human being, such as visual perception, speech recognition, decision-making, and translation between languages. Machine Learning, one type of Artificial Intelligence (AI), allows software to learn from data and make decisions. AI/ML are mainly applied to solving problems on optimization, classification, clustering, and prediction. Two types of AI/ML research problems are involving on software application and on theory discovery.

In: S. Chawla, B. Wadhwa, P. Muenchaisri (eds.): Proceedings of the 2nd Software Engineering Education Workshop (SEED 2019), Putrajaya, Malaysia, 02-Dec-2019, published at http://ceur-ws.org

Several research problems in software engineering particularly in requirements engineering, defect prediction, and coding are solved using AI/ML. Tahira Iqbal *et al.* present literature review of AI/ML for requirements engineering research problems [Iqb18]. Robert Feldt *et al.* present the review of AI in SE [Fel18]. Previous researches [San19], [Poo18], [Mek12], [Man11], [Sre16], [Kae19], [Phe19] at CPCU have applied AL/ML methods in RE, coding, software quality improvement and maintenance. In this paper, the author intends to investigate which SE activities that are often used AI/ML methods to solve research problems. The most AI/ML used activity will be summarized. Research problems of this paper include

RQ1: What is the current state of the art in Software Engineering activities problems which are resolved with AI/ML methods in APSEC 2018?

RQ2: What is the current state of the art in Software Engineering activities problems which are resolved with AI/ML methods at CPCU?

The scope of this research is to extract information from APSEC 2018 proceedings and Software Engineering group at Chulalongkorn University, Thailand. The results of the study may be considered to possibly update AI/ML contents of some courses of the Software Engineering curriculum.

Section 2 briefly describes related research. The methodology is explained in section 3. Results and conclusions are described in section 4 and section 5 respectively.

2 Related Research

Robert Feldt *et al*. present "the AI in SE Application Levels (AI-SEAL) taxonomy" [Fel18]. Applications are categorized according to their point of AI application (process or product), the type of AI technology used and the automation level (1 to 10) allowed." Types of AI include Symbolist, e.g., inverse deduction, Connectionist, e.g., backpropagation, Evolutionary, e.g., genetic programming, Bayesians, e.g., probabilistic inference and Analogizers, e.g., kernel machines. Seventeen papers of previous RAISE workshops (out of 44 papers) are papers with the application of AI to software engineering. The papers are classified based on the three aspects. The results show that there are 12 process, 3 product and 2 runtime-related papers. Eight of them are Analogizer, five Symbolist, and one for Evolutionary and for Connectionist. Most of them have low level of automation (level 2-3).

Tahira Iqbal *et al.* conduct literature review to obtain an overview of how ML are used in requirements engineering (RE) which includes requirements elicitation, requirements analysis, requirements documentation and requirements verification [Iqb18]. The paper summarizes as follows. 1. In requirements elicitation and discovery phase, several kinds of research use mining, ML, and recommendation system to classify requirements into improvement request or not, into *bugs*, *features* and *junk* and to discover evolutionary requirements and related requirements. 2. In requirements specification and analysis phase, ML can use to identify if a set of requirements is Non-functional Requirements (NFR) or not and is functional Requirements (FR) or not, to distinguish FR from NFR, to find Prioritization of Requirements, and to identify if a NFR is security requirement. 3. In requirements validation phase, some researches validate on consistency and traceability. 4. In requirement management activity, some researches focus on visualization of a large group of requirements in order to make better decision and on using ML to classify and cluster information in requirements specification into requirements or information and to grouping similar and related requirements and place them contiguously.

3 Research Method

3.1 Two Aspects of Interests

Two aspects consist of software engineering aspect and AL/ML aspect. Since Tahira Iqbal *et al.* review papers which apply ML to RE [Iqb18] and Robert Feldt *et al*. present papers that use AI on either process or product aspect [Fel18], this paper further identifies specific process activities of software engineering by which AL/ML methods are used in solving problem. Processes as software engineering aspect include Requirements Gathering, Analysis, Design, Coding, Testing, Maintenance, Software Quality Improvement (Product). AL/ML methods as the second aspect include 1. natural language processing, 2. supervised learning (support vector machine (SVM)), 3. unsupervised learning (genetic

In: S. Chawla, B. Wadhwa, P. Muenchaisri (eds.): Proceedings of the 2nd Software Engineering Education Workshop (SEED 2019), Putrajaya, Malaysia, 02-Dec-2019, published at http://ceur-ws.org

algorithms, clustering/classification, similarity, K-Nearest Neighbour (KNN), and 4. reinforcement learning (neural network, Bayesian network, Naïve Bayes).

3.2 Scope of the study

To answer RQ1, 82 regular papers of Asia-Pacific Software Engineering Conference 2018 (APSEC 2018) are reviewed. Seven-teen papers containing keywords on AL/ML methods are found and studied. To answer RQ2, seven interviews and some paper reviews are performed at the Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, Bangkok, Thailand. Five papers [San19], [Poo18], [Mek12], [Man11], [Sre16], [Kae19] use AI/ML for solving software engineering problems.

4 Results

4.1 Software Engineering Activities and AI/ML at APSEC 2018

Table I classifies papers according to software engineering activities and AI/ML methods used. Coding and software quality improvement have more papers using AI/ML methods to solve research problems than other software engineering activities which answers research question#1 (RQ1). The details of each paper are described in the following.

4.1.1 Requirements Engineering (RE) Papers

Requirements and analysis model are classified into functional or dysfunctional kind with The Open Innovation in Requirements Engineering (OIRE) method [Yin18] and using different ML methods. MatGap is a tool providing gap analysis of two sets of Business Rules: A golden reference and target set (one verb concept is removed). Similarity scores are used to find correct matches.

4.1.2 Coding Papers

API usage patterns are automatically generated from the natural language queries [Tin18]. Rules-based regularization method is used to get concise usage patterns. The encoder of the proposed method uses the recurrent neural network with long short-term memory (LSTM) units. Comparisons with other methods are presented. Doc2Vec is an NLP tool that uses neural networks [Ama18]. Comments of original java code and the comment-erased version are assessed with Doc2Vec. Similarity score of each version is computed and checked if the erased-comments version has a high value or not. Shinyama *et al.* analyze code comments to boost program comprehension using a decision-tree based classifier [Shi18]. Three different classifiers are built for each element: Extent, Target and Category. SOQDE is a supervised learning-based (random forest) question difficulty estimation model [Has18]. STAR is a specialized tagging approach for docker repositories [Yin18]. Logistic regression-based classifier is used to determine whether a tag should be assigned to a repository. Four methods of tagging are compared.

An automatic approach using KNN and Random forest [Kim18] is proposed to validate log levels in a class or a method: Trace, Debug, Info, Warn, Error, or Fatal. A tool for Tuning the Level of Parallelism of Spark Applications Optimizations with KMeansClustering and StreamingWordCount is proposed [Ros18]. An approach with neural network, naive Bayes, logistic regression and SVM, DTPre based on decision tree [Moh18] is proposed to predict which pull requests will get reopened in GitHub. SLAMPA tool recommends code snippets with statistical language Model [Zho18] using a deep neural network called Recurrent Neural Network (RNN).

4.1.3 Software Quality Papers

A hybrid analysis method is designed to detect malicious JavaScript code [He18]. Several classifiers are used in constructing classification models, such as Random Forests (RF), Logistic Regression (LR), Naive Bayes (NB) and Support Vector Machine (SVM). Detecting Duplicate Bug Reports with Convolutional Neural Networks (CNN) is presented in [Xie18].

In: S. Chawla, B. Wadhwa, P. Muenchaisri (eds.): Proceedings of the 2nd Software Engineering Education Workshop (SEED 2019), Putrajaya, Malaysia, 02-Dec-2019, published at http://ceur-ws.org

A ML-based approach is proposed to categorize and predict Invalid vulnerabilities on common vulnerabilities and exposures [Che18]. A machine learning model adopts several classic classification algorithms including naïve Bayes, multinomial naive Bayes, SVM and Random Forest for learning from the whole dataset of invalid CVEs. A Comparison of Nano-patterns and software metrics in Vulnerability Prediction is presented in [Sul18]. A vulnerability prediction model using the nano-patterns extracted from vulnerable and neutral (we use the term "neutral" to refer to methods where no known vulnerability exists) code of different software systems. Three machine-learning techniques are used to classify vulnerable code including Naive Bayes (NB), Support Vector Machine (SVM) and Logistic regression (LR).

A Top-k Learning to Rank (LTR) Approach using Random forest is designed to predict cross- project software defect [Wan18]. A bug localization model is constructed with two main parts including character-level convolutional neural network (CNN) and recurrent neural network (RNN) language model [Xia18].

Table I Research papers with Software Engineering activities by which ML/AI methods are used for problem solving.

Paper#	Software Engineering Activities*	AI/ML Type**	Methods***	Reference
1	RE	ML	NB, Max, DT, RF, SVM	[Yin18]
2	RE	Natural	Similarity	[Min18]
3	Coding	ML	Clustering, neural network	[Tia18]
4	Coding	Natural, ML	NN, similarity	[Ama18]
5	Coding	ML, Natural	DT, CoreNLP	[Shi18]
6	Coding	Natural, ML	CoreNLP, RF, KNN, BN	[Has18]
7	Coding	ML	LR, Similarity	[Yin18]
8	Coding	ML, Natural	KNN, RF	[Kim18]
9	Coding	ML	KmeansClustering	[Ros18]
10	Coding	ML	NN, NB, LR and SVM, DTPre based on DT	[Moh18]
11	Coding	ML	Deep NN (Recurrent NN)	[Zho18]
12	SWQ-defect	ML	RF, LR, NB, SVM	[He18]
13	SWQ-defect	ML, Natural	NN	[Xie18]
14	SWQ-security	ML	NB, SVM, RF	[Che18]
15	SWQ-security	ML	NB, SVM, LR	[Sul18]
16	SWQ-defect	ML	RF	[Wan18]
17	SWQ-defect	ML	Convolution NN, RNN	[Xia18]

*SE activities: RE: Requirements Engineering, Design, Coding, Testing, SWQ: Software Quality Improvement (SWQ-defect, SWQ-security),

**Natural: Natural language processing, ML: Machine Learning

***NB:Naïve Bayes, Max:MaxEnt, DT: Decision Trees, RF: Random Forest, SVM: Support vector machine, NN: Neural Networks, BN: Bayes Network, LR: Logistic Regression, ARL: Association Rule Learning, EBL:Explanation-based learning,

4.2 Software Engineering Activities and AI/ML at CPCU

This section explores research conducted at the Department of Computer Engineering, Chulalongkorn University, Thailand. There are 33 faculty members. Twelve faculty members have main researches in AI/ML and five in SE. Only two faculty members solve software engineering research problems using AI/ML.

In: S. Chawla, B. Wadhwa, P. Muenchaisri (eds.): Proceedings of the 2nd Software Engineering Education Workshop (SEED 2019), Putrajaya, Malaysia, 02-Dec-2019, published at http://ceur-ws.org

Table II shows research papers which use AI/ML methods solving software engineering research problems. Naive Bayes method is used to classify short text of requirements [San19]. Association rule learning (ARL) is used to find impact factors for rejection of pull requests on GitHub [Poo18]. Explanation-based learning in a meta-programming approach is used to detects of Object-Oriented design defects [Mek12]. This paper uses machine learning methods (Naive Bayes, Logistic, IB1, Ibk, VFI, J48 and Random forest) to predict bad-smells design from software design model [Man11]. Defect-related keywords are discovered using natural language process (NLP) by analyzing user feedback to extract defect related keyword [Sre16]. Prioritizing software maintenance plan uses Analytical Hierarchy Process (AHP). Software problem report types are classified using machine learning [Kae19]. Mobile application user reviews are classified for generating tickets on issue tracking system.

Software quality improvement (SWQ-defect) has more papers using AI/ML methods to solve research problems than other software engineering activities which answers research question#2 (RQ2). However, the result from only 7 papers is not sufficient to make any general conclusion.

Table II Research papers with Software Engineering activities by which AI/ML methods are used for problem solving at the Department of Computer Engineering, Chulalongkorn University

1	RE	ML	NB	[San19]
2	Coding	ML	ARL	[Poo18]
3	SWQ-defect	ML	EBL	[Mek12]
4	SWQ-defect	ML	NB, LR, RF	[Man11]
5	SWQ-defect	NLP	AHP	[Sre16]
6	Maintenance	NLP/ML	NB	[Kae19]
7	Maintenance	NLP/ML	NB, DT	[Phe19]

5 Conclusions and future works

This paper preliminary investigates on using AI/ML of software engineering activities from papers published in APSEC 2018 and at the Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, Thailand. In APSEC 2018, there are more papers in coding and in software quality improvement using AI/ML methods than other software engineering activities. At CPCU, most AI/ML researches are focused mainly in theoretical aspects and in applying in several application domains. Only two faculty members uses AI/ML in software engineering research problems. Future works include 1. review more papers with automatic tool 2. extend scope to cover research conducted in industry 3. find a possibility to include AI/ML into SE courses/curriculum.

References

[Iqb18] T. Iqbal et al. *A Bird's Eye View on Requirements Engineering and Machine learning*, the 25th Asia-Pacific Software Engineering Conference 2018 (APSEC 2018), December 4-7, Nara, Japan.

[Fel18] R. Feldt et al. *Ways of Applying Artificial Intelligence in Software Engineering*, the Workshop on Realizing Artificial Intelligence Synergies in Software Engineering (RAISE 2018), May 2018, Gothenburg, Sweden.

[San19] K. Sangounpao and P.Muenchaisri. *ONTOLOGY-BASED NAIVE BAYES METHOD SHORT TEXT CLASSIFICATION FOR A SMALL DATASET*, 20th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD 2019), Japan.

[Poo18] P. Pooput and P.Muenchaisri. *Finding Impact Factors for Rejection of Pull Requests on GitHub*, The VII International Conference on Network, Communication and Computing (ICNCC 2018), Taipei, Taiwan, Dec.14-16, 2018.

[Mek12] S. Mekruksavanichi, P. P. Yupapin and P. Muenchaisri. *Analytical Learning Based on a Meta-programming Approach for the Detection of Object-Oriented Design Defects*, Information Technology Journal, 11: 1677-1686, 2012.

In: S. Chawla, B. Wadhwa, P. Muenchaisri (eds.): Proceedings of the 2nd Software Engineering Education Workshop (SEED 2019), Putrajaya, Malaysia, 02-Dec-2019, published at http://ceur-ws.org

[Man11] N. Maneera and P. Muenchaisri. *Bad-smell Prediction from Software Design Model Using Machine Learning Techniques*, the 8th International Joint Conference on Computer Science and Software Engineering (JCSSE2011), Nakhon Pathom, Thailand, May 11-13, 2011.

[Sre16] K. Srewuttanapitikul and P. Muenchaisri. *Prioritizing Software Maintenance Plan by Analyzing User Feedback*, The International Conference on Information Science and Security 2016 (ICISS 2016), December 19th-22nd, 2016, Pattaya, Thailand.

[Kae19] Phatcharaporn Kaewnoo and Twittie Senivongse, "*Identification of Software Problem Report Types Using Multiclass Classification*," 2019 The 3rd International Conference on Software and e-Business (ICSEB 2019), December 9-11, 2019, Tokyo, Japan

[Phe19] Kittisak Phetrungnapha and Twittie Senivongse, "*Classification of Mobile Application User Reviews for Generating Tickets on Issue Tracking System*," The 12th International Conference on Information & Communication Technology and System (ICTS 2019), July 18, 2019, Surabaya, Indonesia

[Yin18] H. Yin et al. *The OIRE Method - Overview and Initial Validation*, the 25th Asia-Pacific Software Engineering Conference 2018 (APSEC 2018), December 4-7, Nara, Japan.

[Mit18] S. Mitra et al. *MatGap A Systematic Approach to Perform Match and Gap Analysis among SBVR-Based Domain Specific Business Rules*, the 25th Asia-Pacific Software Engineering Conference 2018 (APSEC 2018), December 4-7, Nara, Japan.

[Tin18] Y. Tian et al. *Automatically Generating API Usage Patterns from Natural Language Queries*, the 25th Asia-Pacific Software Engineering Conference 2018 (APSEC 2018), December 4-7, Nara, Japan.

[Ama18] H. Aman et al. *A Doc2Vec-Based Assessment of Comments and Its Application to Change-Prone Method Analysis*, the 25th Asia-Pacific Software Engineering Conference 2018 (APSEC 2018), December 4-7, Nara, Japan.

[Shi18] Y. Shinyama et al. *Analyzing Code Comments to Boost Program Comprehension*, the 25th Asia-Pacific Software Engineering Conference 2018 (APSEC 2018), December 4-7, Nara, Japan.

[Has18] Sk. A. Hassan et al. *SOQDE: A Supervised Learning based Question Difficulty Estimation Model for Stack Overflow*, the 25th Asia-Pacific Software Engineering Conference 2018 (APSEC 2018), December 4-7, Nara, Japan.

[Yin18] K. Yin et al. *STAR: A Specialized Tagging Approach for Docker Repositories*, the 25th Asia-Pacific Software Engineering Conference 2018 (APSEC 2018), December 4-7, Nara, Japan.

[Kim18] T. Kim et al. *An Automatic Approach to Validating Log Levels in Java*, the 25th Asia-Pacific Software Engineering Conference 2018 (APSEC 2018), December 4-7, Nara, Japan.

[Ros18] E. Rosales et al. *lpt: a Tool for Tuning the Level of Parallelism of Spark Applications*, the 25th Asia-Pacific Software Engineering Conference 2018 (APSEC 2018), December 4-7, Nara, Japan.

[Moh18] A. Mohamed et al. *Predicting which pull requests will get reopened in GitHub*, the 25th Asia-Pacific Software Engineering Conference 2018 (APSEC 2018), December 4-7, Nara, Japan.

[Zho18] S. Zhou et al. *SLAMPA: Recommending Code Snippets with Statistical Language Model*, the 25th Asia-Pacific Software Engineering Conference 2018 (APSEC 2018), December 4-7, Nara, Japan.

[He18] X. He et al. *Malicious JavaScript Code Detection Based on Hybrid Analysis*, the 25th Asia-Pacific Software Engineering Conference 2018 (APSEC 2018), December 4-7, Nara, Japan.

[Xie18] Q. Xie et al. *Detecting Duplicate Bug Reports with Convolutional Neural Networks*, the 25th Asia-Pacific Software Engineering Conference 2018 (APSEC 2018), December 4-7, Nara, Japan.

[Che18] Q. Chen et al. *Categorizing and Predicting Invalid Vulnerabilities on Common Vulnerabilities and Exposures*, the 25th Asia-Pacific Software Engineering Conference 2018 (APSEC 2018), December 4-7, Nara, Japan

[Sul18] K. Z. Sultana et al. *A Comparison of Nano-patterns Vs. Software Metrics in Vulnerability Prediction*, the 25th Asia-Pacific Software Engineering Conference 2018 (APSEC 2018), December 4-7, Nara, Japan.

[Wan18] F. Wang et al. *A Top-k Learning to Rank Approach to Cross-Project Software Defect Prediction*, the 25th Asia-Pacific Software Engineering Conference 2018 (APSEC 2018), December 4-7, Nara, Japan.

[Xia18] Y. Xiao et al. *Improving Bug Localization with Character-level Convolutional Neural Network and Recurrent Neural Network*, the 25th Asia-Pacific Software Engineering Conference 2018 (APSEC 2018), December 4-7, Nara, Japan.

In: S. Chawla, B. Wadhwa, P. Muenchaisri (eds.): Proceedings of the 2nd Software Engineering Education Workshop (SEED 2019), Putrajaya, Malaysia, 02-Dec-2019, published at http://ceur-ws.org