Int J Syst Assur Eng Manag (April 2018) 9(2):344–353
https://doi.org/10.1007/s13198-017-0649-x

ORIGINAL ARTICLE

Incorporation of ISO 25010 with machine learning to develop a novel quality in use prediction system (QiUPS)

Osama Alshareet[1] · **Awni Itradat**[2] · **Iyad Abu Doush**[3] · **Ahmad Quttoum**[2]

Received: 3 May 2016 / Revised: 6 January 2017 / Published online: 21 June 2017

Abstract Guided by the eagerness to fulfill business objectives, quality assurance has become one of the highlighted topics in software engineering. With the rise of globalization and free markets, software users are becoming increasingly powerful with their ability to buy or reject computer software. While there is agreement over achieving quality, there is debate over the definition of quality. To illustrate, literature shows inconsistencies between a software development team definition to quality and a user definition to quality. Recently, there is a tendency amongst researchers to appreciate the need for studying quality from a user prospective. Following a systematic approach, this research attempts to develop a QiUPS, an expert system for predicting quality in use in early software development phases. With the scariness of research data in this field, the research generates a dataset from the documentation of Information, Communication, and E-learning Technology Centre software projects. The research methodology followed a comparative approach as it statistically compared four different classification algorithms (CAs) in terms of accuracy in classifying the research dataset. After that, the research results led the researchers to compare the performance of artificial neural networks with convolutional neural networks in three empirical experiments, which is rarely researched. Finally, the research incorporated the best CA with ISO 25010 in order to develop the novel QiUPS. The research results are consistent and contributive to this rarely researched area.

Keywords Quality in use prediction system (QiUPS) · ISO 25010 software quality model · Classification algorithms (CAs) · Artificial neural networks (ANN) · Convolutional neural networks (CNN) · Quality in use (QiU) · User-centered applications (UCA) · Multi-layer perceptron (MLP)

1 Introduction

Software projects failure is common these days, some researchers (Hoffman 1999; Jørgensen and Moløkken-Østvold 2006; Dwivedi et al. 2015) suggested that diverse failure incidences in software projects rise up to over 80% (Hoffman 1999; Jørgensen and Moløkken-Østvold 2006; Dwivedi et al. 2015). Moreover, literature shows that software projects are frequently influenced by large number of development problems, such as lean project management, high program expenses, time lags, and non-efficient advertisement (Verner et al. 2007; El Emam and Koru 2008; Hussain and Mkpojiogu 2016). Since some of these difficulties appear during early software development, literature suggests that early treatment of problematic situations may improve software project success. In

✉ Osama Alshareet
bf99qo@student.sunderland.ac.uk

Awni Itradat
itradat@hu.edu.jo

Iyad Abu Doush
iyad.doush@yu.edu.jo

Ahmad Quttoum
quttoum.ahmad@courrier.uqam.ca

[1] ICET, Hashemite University, Zarqa, Jordan

[2] Department of Computer Engineering, Hashemite University, Zarqa, Jordan

[3] Department of Computer Sciences, Yarmouk University, Irbid, Jordan

addition, existing research emphasized on detecting factors which may affect the quality of project outcome (Cerpa et al. 2010; Reyes et al. 2011; Ahimbisibwe et al. 2015). Practically, the lack of strict software quality methodologies in software development usually lead to severe consequences on software development process as a whole (Gefen and Straub 2001; Jan et al. 2016).

In traditional context, the experience of project managers plays a decisive role in predicting the implications of decisions made during project life cycle. When ignored, signs of project failure could develop into total failure for the whole project. Though some signs of project failures are detectable, many project managers fail to make corrective actions at the right moment. Accordingly, there is an increasing interest in developing expert systems for predicting project outcomes in early stages.

Naturally, project stakeholders put significant pressure over development process, which impacts the quality of IT projects (Heravi et al. 2015). To illustrate, these pressures include changes in user requirements and inconsistency between user definition to quality and software development team definition to quality (Woodroof and Kasper 1998).

Since there is a large variety of software published online, users are considerably selective in what they consider the best software for them. Accordingly, it is important to study how a user interprets software quality, and how that interpretation differs from a software developer interpretation to quality. Garvin (Deming 2000) identified two forms of quality, qualitative and quantitative. The qualitative part of quality is based on opinions, and personal views. On the other hand, quantitative part of quality is based on numbers and statistics. ISO/IEC 25010:2011 is composed of two models, "quality in use" model and "product quality" model. This research covers the first model, which is composed of five characteristics. Table 1 summarizes these characteristics briefly as described by ISO (2011). With regard to QiU applications, QiU is increasingly adopted in many software programming fields, particularly in user-centered applications (UCA). For instance, they are common in business process management (Heinrich 2014), web applications (Orehovački et al. 2013; Lippert and Govindarajulu 2015), and mobile applications (Alnanih et al. 2012; Kim et al. 2015).

Adopting QiU in early stages of software development cycle proved to be beneficial for all stakeholders. However, with the absence of a clear understanding of QiU, it may slow software development process and lead to unexpected project failure. With the scariness of research literature, this research aims to contribute to the process of predicting QiU in early software development cycles through integrating ISO 25010 with machine learning, in order to develop a novel QiUPS.

2 Literature review

There is a tendency amongst researchers to develop QiU models based on ISO standard frameworks. La and Kim (2013) used ISO 9126 model to develop a service-based mobile system. Osman and Osman (2013) calculated QiU through a well-defined questionnaire. Oliveira et al. (2014) used ISO 9126 to develop a tool for evaluating project management tools usability.

The semantic web exploration tools, QiU model (SWET-QUM) uses ISO 25010 to evaluate semantic web exploration tools (González et al. 2012). Meanwhile, Becker et al. (2012) proposed a strategy for understanding and improving quality, as well as they recommended making modifications on operability characteristic to improve QiU measurement.

Ardito et al. (2014) proposed a pattern recognition method that uses a list of QiU evaluation patterns to detect the quality of e-learning systems. Apart from the positive results, the study concluded that pattern recognition application in QiU could be time consuming.

With regard to machine learning applications in predicting QiU, Datamining have been used widely to predict project software project outcome (Abe et al. 2006; Mizuno et al. 2004; Liu et al. 2014; Wang 2007; Smite 2007; Oztekin et al. 2013; Halees 2014). Abe et al. (2006) used a Bayesian Classifier to predict software project outcome. Mizuno et al. (2004) used a tenfold cross-validation (Bayesian Classifier) to predict the output of a runaway software.

Using a k-means algorithm, Wang predicted the output of open source software projects (Wang 2007). Cheng and Wu (2008) proposed a support vector machine and a fast messy genetic algorithm to predict software project success.

Examining research literature, first, machine learning applications for predicting QiU are rarely explored. Second, the research literature shows less interest for comparing the performance of diverse machine learning algorithms in predicting QiU. Third, there is apparent gaps of research in comparing the performance of different Neural Networks in predicting QiU. Finally, there is also less interest in using user requirements and feedback to predict QiU. Consequently, this research attempts to highlight these research gaps and contribute empirically to the process of filling them.

3 Methodology

The research methodology is formed of four subsections. The first subsection discusses deriving measurements from the ISO 25010 Quality Model. Then, the second

Table 1 The characteristics of ISO/IEC 25010, quality in use

Characteristics	Sub-characteristics	Definition
Effectiveness		This characteristic defines the accuracy and completeness of achieving project goals. For example, if the requested objective is develop an object oriented class, then accuracy would be measured by the number of errors after compilation
Efficiency		Utilized resources to achieve the accuracy and completeness objectives. The resources include financial resources and human resources
Satisfaction	Usefulness	To what level a user is satisfied with achieving software system objectives, as well as implications of use
	Trust	To what level a user is confident that the software system will behave as specified in user requirements
	Pleasure	How pleasurable is using the software system in the light of satisfying user needs
	Comfort	Physical comfort with using the software system
Freedom from risk	Economic risk mitigation	To what level a software system mitigates the expected risk to efficient operation, commercial property, financial status, reputation or other resources in the designated contexts of use
	Health and safety risk mitigation	To what level a software system mitigates the expected risk to people in the designated contexts of use
	Environmental risk mitigation	To what level a software system mitigates the expected risk to property or the environment in the contexts of use
Context coverage	Context completeness	To what level a software system can be used with efficiency, effectiveness, satisfaction, and freedom from risk in all the designated contexts of use
	Flexibility	To what level a software system can be used with efficiency, effectiveness, satisfaction, and freedom from risk in contexts beyond those initially specified in the requirements

sub-section explains the research dataset and its source. Finally, the third subsection explains ANN and its use as a CA in this research context while the fourth section discuses CNN structure.

3.1 Deriving measurements from ISO 25010 quality model

The research literature shows that ISO 25010 could be used to derive QiU measurements. This research converted ISO 25010, QiU general specifications into measurements, based on similar research model (Bevan 2009). To illustrate, the following represent examples of converting ISO 25010, QiU general specifications into measurable variables:

Concerning Effectiveness characteristic, the number of achieved objectives as requested by the user compared against the number of all requested objectives by the user. Formula (1) shows how to calculate this measurement.

$$QiUPS\,Effectiveness = \frac{number\ of\ achieved\ objectives}{number\ of\ all\ required\ objectives} \tag{1}$$

With regard to usefulness sub-characteristic of satisfaction, this sub-characteristic is measured by comparing the number of advantageous feedback notes against the total number of feedback notes. Feedback notes are extracted from online form and achieved tasks. Formula (2) shows how to calculate usefulness.

$$QiUPS\,Usefulness = \frac{number\ of\ advantageous\ notes - number\ of\ disadvantageous\ notes}{total\ number\ of\ feedback\ notes} \tag{2}$$

On the subject of context completeness sub-characteristic of context coverage, for a given QiUPS case, this sub-characteristic is measured by calculating the average of user satisfaction and software freedom from risk. Satisfaction is calculated using four measurements, which are usefulness, trust, pleasure, and comfort. To illustrate, Formula (3) demonstrates how to calculate this measurement.

$$QiUPS\,Context\,Completeness = \frac{Satisfaction + Freedom\,from\,Risk}{2} \tag{3}$$

Touching flexibility sub-characteristic of context coverage, for a given QIUPS case, this sub-characteristic is measured by calculating the average of user satisfaction (in domains beyond user requirements) and software freedom from Risk (in domains beyond user requirements). Formula (4) demonstrates how to calculate this measurement.

$$QiUPS\,Flexibility = \frac{Satisfaction\,(beyond\,user\,requirements) + Freedom\,from\,Risk\,(beyond\,user\,requirements)}{2}. \tag{4}$$

3.2 Research dataset

The research dataset was extracted from archived files and database of Information, Communication, and E-learning Technology Centre (ICET), Hashemite University of Jordan. Since ICET works as a software house for developing diverse software applications, the research used ICET database and archive to generate the main research dataset. Figure 1 illustrates the process of extracting the research dataset. Unfortunately, 317 of software projects were excluded because of missing data. Hence, the dataset was left only with 1899 software projects. Table 2 shows the dataset processing summary with the resulted three classes. Since the dataset fields were basically ratios, the data type of the data fields is real number. With regard to statistical tools, the researcher used WEKA and IBM SPSS to implement the statistical experiments. Looking at Fig. 2, most of the cases are classified as having a medium QiU with 1423 out of 1899 cases. Highlighting the problem of inconsistency between developers definition to quality and users definition to quality, this chart shows that software development team efforts are focused mainly on developing a highly functional software rather than a software with high QiU.

Table 2 Dataset processing summary

Output class	Frequency	Percent	Valid percent
Low QiU = 1	333	17.5	17.5
Meduim QiU = 2	1423	74.9	74.9
High QiU = 3	143	7.5	7.5
Total	1899	100.0	100.0

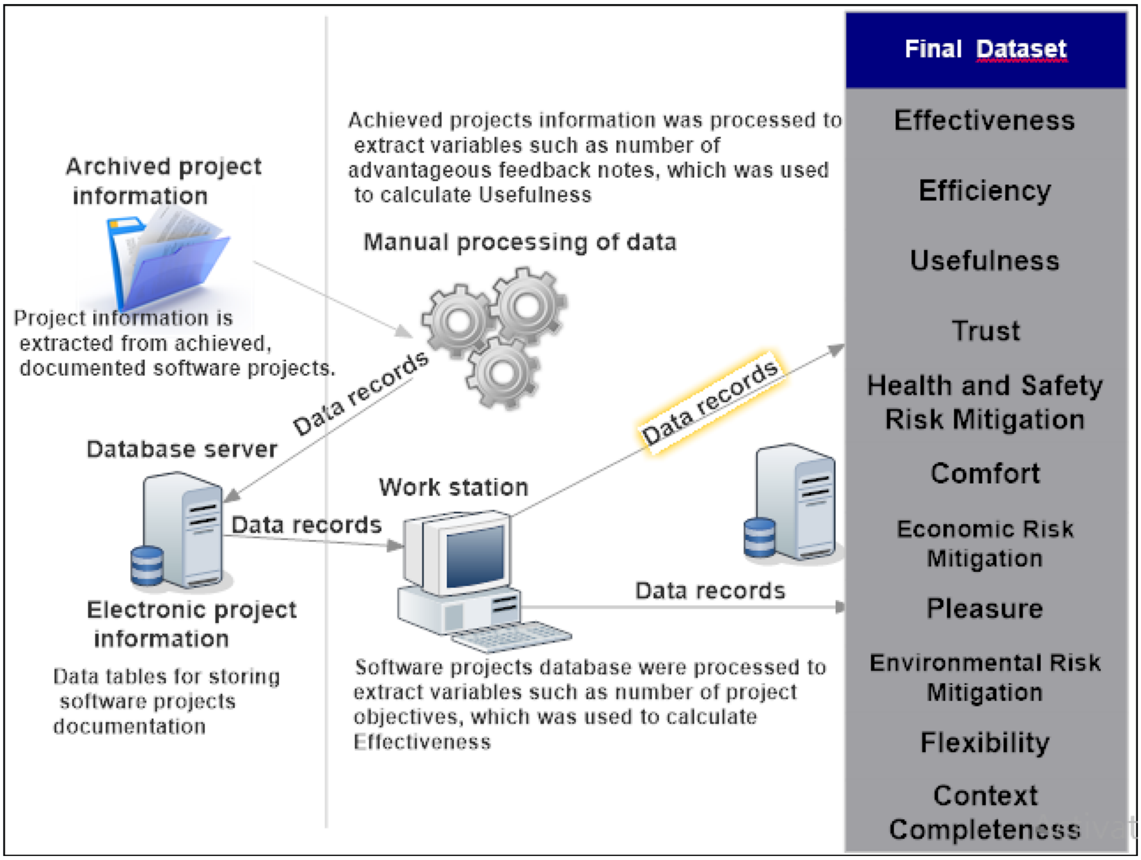
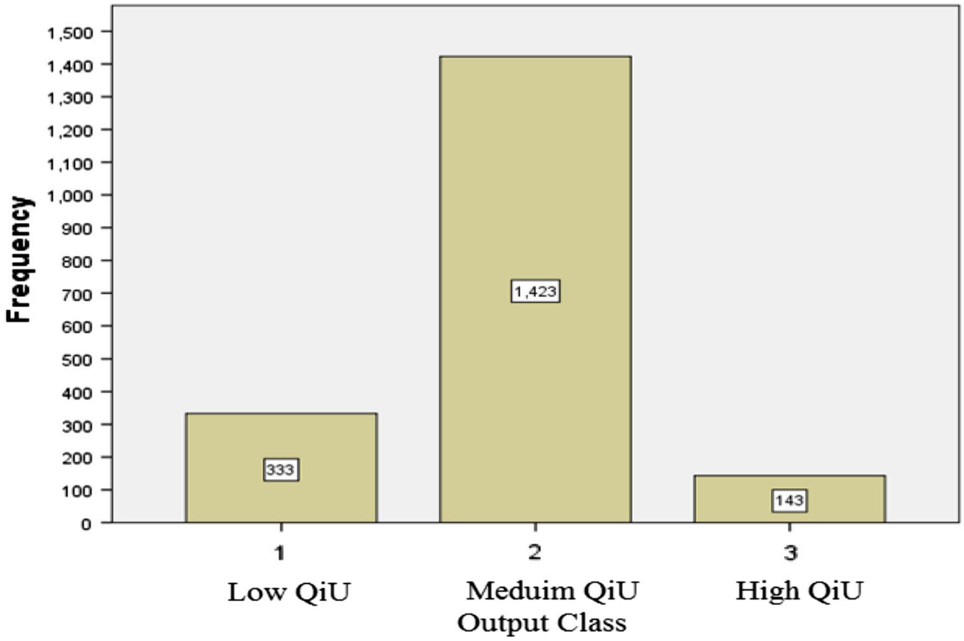

Final Dataset
Effectiveness
Efficiency
Usefulness
Trust
Health and Safety Risk Mitigation
Comfort
Economic Risk Mitigation
Pleasure
Environmental Risk Mitigation
Flexibility
Context Completeness
Archived project information
Project information is extracted from achieved, documented software projects.
Achieved projects information was processed to extract variables such as number of advantageous feedback notes, which was used to calculate Usefulness
Manual processing of data
Data records
Database server
Data records
Work station
Data records
Data records
Electronic project information
Data tables for storing software projects documentation
Software projects database were processed to extract variables such as number of project objectives, which was used to calculate Effectiveness

Fig. 1 The process of extracting research dataset

Fig. 2 Dataset frequency chart



3.3 Artificial neural networks (ANN)

Partially derived from biological sciences, an ANN consists of a number of coordinated processes that accept a predefined input, process it, and predict a certain output. Based on brain neuron, ANN provides tools for learning rules from formatted examples. Looking at Fig. 3, hidden layer are positioned between the input and output layers. One of ANN major learning objectives is to produce rules in alignment with input and output parameters. Widely used in expert systems research, many researchers agree on several advantages for using ANN, which could be summarized as follows:

1. ANN ability to adaptively learn how to perform tasks based on primal data.
2. ANN learns quickly through developing its organization during learning process.
3. With the support of a dedicated hardware and software, ANN could be implemented effectively in parallel architectures.
4. Comparing to other CAs, ANN can process large amount of databases effectively.

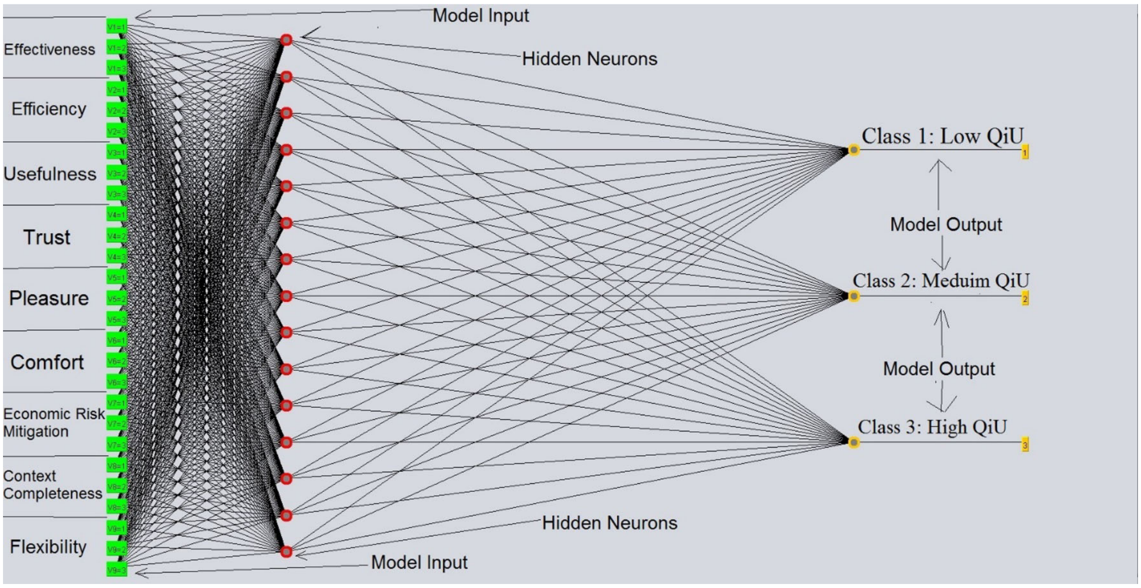


Fig. 3 Using WEKA tool, a visualization of the generated ANN

ANN uses an administered learning method called back propagation for programming the neural network (Burr 2015). Developers train ANN to figure out how to transform data input to a required output, and fit the model to a specified prediction context (Craven and Shavlik 2014; Schmidhuber 2015).

Technically, ANN acts as a managed learning system for studying and solving various logical problems, including pattern recognition and classification. Looking at Fig. 3, the ANN algorithm coordinates the weights of neural connection to reduce error values in the network output. If these modifications resulted error minimization, then the designated ANN learned a new function. Aside from naming conventions, in this research context, the research emphasized the term ANN to distinguish between traditional neural networks and CNNs. Moreover, the research used multi-layer perceptron (MLP) as the experimental ANN model.

3.4 Convolutional neural networks (CNN)

Rarely used outside image processing area, convolutional neural networks (CNN) inherits most of its features from ANN (Sainath et al. 2015). However, CNN structure layers based on selective convolutional principle. The main difference between ANN and CNN is that CNN contain specific layers for convolution and pooling, which implies that the layers after the input are connected. As Fig. 4 shows, composed of five inputs, the first layer (X − 1) contains five inputs. Additionally, each neuron in layers "X" and "X + 1" receives three inputs from the previous layer, presenting balanced architecture of a CNN. To illustrate, this structure allows balanced processing of data. Though not demonstrated in Fig. 4, CNN have more hidden layers than traditional ANN. Accordingly, this research paper selected CNN as one of the experimented CAs.

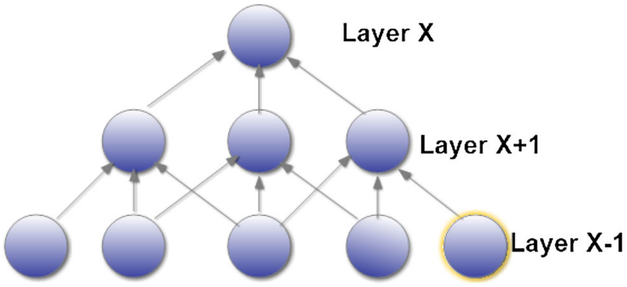


Fig. 4 A visualization of CNN

4 Results and discussion

Divided into three experiments, this section discusses the research main results, which were processed using statistical tools, WEKA and IBM SPSS tools. The following subsections explain the research experiments.

4.1 Experiment 1: Testing the performance of different classification algorithms

The first experiment aims to test different CAs in terms of correctly classified instances, incorrectly classified instances, relative absolute error, root relative squared error, root mean squared error, and mean absolute error. In this experiment context, error values are used to evaluate the difference between model prediction and the real output.

Looking at Fig. 3, ANN model building process lasted for 6.79 s, with nine inputs and three outputs. Table 3 shows that the number of correctly classified instances for ANN and CNN are very close and both are relatively higher than other CAs. ANN predicted 1668 (87.8357%) correctly classified instances while CNN predicted 1654 (87.0985%) correctly classified instances. Additionally, error values for both ANN and CNN have subtle differences, as well as both are relatively lower than other algorithms, including Naïve Bayes CA. On the other hand, Naïve Bayes classification follows both ANN and CNN in terms of correctly classified instances with 1585 (83.465%) instances. However, Naïve Bayes error values are higher than error values for both ANN and CNN, with no exceptions. The J48 algorithm took about 0.14 s to build. When compared with other CAs, J48 performance is the lowest with only 1423 (74.9342%) correctly classified instances. Similarly, error values for the J48 model are the

Table 3 Comparing different machine learning algorithms

Comparison feature	ANN		CNN		Naïve Bayes		J48 tree	
Correctly classified instances	1668	87.8357%	1654	87.0985%	1585	83.465%	1423	74.9342%
Incorrectly classified instances	231	12.1643%	245	12.9015%	314	16.535%	476	25.0658%
Mean absolute error	0.129		0.1436		0.1959		0.2684	
Root mean squared error	0.2558		0.2574		0.2933		0.3661	
Relative absolute error (%)	48.0696		53.509		73.0044		100	
Root relative squared error (%)	69.8811		70.3074		80.1298		100	

Table 4 Confusion matrices for the experimented CAs

a	b	c	←classified as	a	b	c	←classified as
(a) Confusion matrix for ANN				(b) Confusion matrix for CNN			
232	101	0	| a = 1, Low QiU	232	101	0	| a = 1, Low QiU
70	1319	34	| b = 2, Medium QiU	70	1335	18	| b = 2, Medium QiU
0	26	**117**	| c = 3, High QiU	0	56	87	| c = 3, High QiU
(c) Confusion matrix for Naïve Bayes				(d) Confusion matrix for J48 tree			
232	101	0	| a = 1, Low QiU	0	333	0	| a = 1, Low QiU
70	1353	0	| b = 2, Medium QiU	0	**1423**	0	| b = 2, Medium QiU
0	143	0	| c = 3, High QiU	0	143	0	| c = 3, High QiU

The results represent the output of WEKA classifiers

Bolded rows represent higher classifications

Table 5 Accuracy of the ANN CA by class

	TP rate	FP rate	Precision	Recall	F-measure	MCC	ROC area	PRC area	Class
	0.697	0.045	0.768	0.697	0.731	0.678	0.910	0.652	1
	0.927	0.267	0.912	0.927	0.919	0.671	0.892	0.955	2
	0.818	0.019	0.775	0.818	0.796	0.779	0.974	0.854	3
Weighted avg.	0.878	0.209	0.877	0.878	0.877	0.681	0.902	0.894	

Table 6 Accuracy of the CNN CA by class

	TP rate	FP rate	Precision	Recall	F-measure	MCC	ROC area	PRC area	Class
	0.697	0.045	0.768	0.697	0.731	0.678	0.916	0.688	1
	0.938	0.330	0.895	0.938	0.916	0.642	0.898	0.958	2
	0.608	0.010	0.829	0.608	0.702	0.691	0.981	0.840	3
Weighted avg.	0.871	0.256	0.868	0.871	0.867	0.652	0.908	0.902	

highest with 100% value in both relative absolute error and root relative squared error. Looking at Table 4, WEKA's confusion matrix reveals more indications about the CAs comparative output. With regard to option "a", Low QiU, the first three CAs made exactly the same output (232 correctly classified) whereas J48 tree failed to predict any Low QiU correctly. Looking at option "b", Medium QiU, unexpectedly, J48 tree was the highest as it predicted flawlessly with 1423 (100%) correctly classified instances. Comes in second place, Naïve Bayes classifier as it predicted 1353 out of 1423 instances. After that, CNN predicted 1335 out of 1423 prediction while ANN predicted 1319 out of 1423 prediction. Looking at output "c", High QiU, In contrary to previous results, both Naïve Bayes and J48 tree predicted poorly with not even a single, correct prediction. Conversely, ANN predicted 117 out of 143 prediction whereas CNN predicted 87 out of 143 prediction.

Overall, both ANN and CNN are providing a more balanced and correct classifications for the research dataset, knowing that ANN is slightly better than CNN. Consequently, this research considers the previous results as empirical indications for the feasibility of developing a QiUPS using either ANN or CNN. Accordingly, the researchers decided to study both CAs with two additional experiments.

4.2 Experiment 2: Testing ANN and CNN in terms of statistical measures of the performance

Formula (5) presents true positive rate (TPR) or sensitivity. Looking at Tables 5 and 6, as TPR demonstrates the proportion of correctly predicted positives, the TPR value for ANN classes are convergent with TPR value for CNN classes. However, ANN predictions are more balanced with 0.818 as TPR in class 3, high QiU.

Table 7 Comparing the ANN training data against real test cases

Comparison feature	(a) Testing the CNN with real test cases (558 instance)	(b) Testing ANN with real test cases (558 instance)
Correctly classified instances	473 (84.767%)	492 (88.172%)
Incorrectly classified instances	85 (15.233%)	66 (11.828%)
Mean absolute error	0.1016	0.1298
Root mean squared error	0.3187	0.2512
Relative absolute error (%)	38.7109	49.4603
Root relative squared error (%)	89.0593	70.2124

$$TPR = \frac{True\,Positives\,(TP)}{True\,Positives\,(TP) + False\,Negatives\,(FN)} \tag{5}$$

Formula (6) shows False Positive Rate (FPR). Looking at Tables 5 and 6, as FPR demonstrates the proportion of falsely predicted positives, Clearly, the overall FPR in ANN classes are slightly lower than the overall FPR in CNN classes, particularly in class 3, High QiU. Accordingly, this proves that ANN is a little more effective than CNN in predicting positive cases correctly.

$$FPR = \frac{False\,Positive(FP)}{FP + True\,Negatives(TN)} \tag{6}$$

Defined as the proportion of relevant instances, Formula (7) demonstrates precision. When compared with CNN, the weighted average of precision value in ANN is slightly higher and more balanced, especially in class 3, high QiU. Hence, ANN ability to predict negative cases is slightly better.

$$Precision = \frac{TP}{TP + FP} \tag{7}$$

Representing harmonic mean of precision and recall, F-measure is calculated using Formula (8). Relatively higher in ANN, F-measure shows better predication and balance with ANN CA.

$$F\text{-}measure = 2 \times \frac{Percision \times TPR}{Percision + TPR} \tag{8}$$

Looking at Tables 5 and 6, the precision-recall curve (PRC) or precision to sensitivity provides indications for the accuracy of the CA. Not harmonious with the conclusions from the previous results, PRC values are slightly higher in CNN case. At the same time, receiver operating characteristic (ROC) values are slightly higher with CNN. Though both PRC and ROC are slightly higher in CNN case, the total ability of both CAs to differentiate between a false and true prediction of QiU is significantly convergent, but with slight advantage to CNN.

Based on the previous results, the researchers decided to make an independent *t* test to find the statistical significance of both ANN and CNN predictions, with μANN = μCNN as null hypothesis. Using WEKA experimenter, the *t* test for both ANN and CNN predictions showed that the value of Sig. (2-tailed) for correctly classified instances is over 0.05, which means that there is no evidence for a statistically significant difference between ANN predictions and CNN predictions, which strengthens the null hypothesis. Consequently, the researchers resorted to experiment 3.

4.3 Experiment 3: Testing the developed QiUPS against real test cases

After developing the QiUPS, the researchers used the QiUPS to predict QiU for 558 published software projects, which were compiled from another projects from ICET. Looking at Table 7, clearly, testing real 558 instances provided results that aligns with the previous results in Tables 3 and 4. In ANN case, 492 (88.172%) of the cases were classified correctly in the training test whereas 473 (84.767%) cases were classified correctly in CNN case. Similarly, error values in both tests a and b are diverse, with absolute errors better in CNN while squared errors better in ANN. Practically, the results align with previous results from the QiUPS tenfold *t* test. On the other hand, examining Table 8, confusion matrices for both tests show better balance and classification with ANN algorithm, particularly in predicting class 3, High QiU. Accordingly, though previous tests show non-significant differences between ANN predictions and CNN predictions, this test shows clear advantage for ANN predictions, especially in predicting software with high QiU.

5 Conclusion

In conclusion, the research highlighted the problem of inconsistency between software developers interpretation to quality and users interpretation to quality. Moreover, the research provided empirical evidences to support the feasibility of integrating a CA with ISO 25010 quality model in order to develop a QiUPS. As an experimented QiU

Table 8 Comparing the ANN training confusion matrix against real tested confusion matrix

a	b	c	←classified as	a	b	c	←classified as
(a) Confusion matrix for the CNN with tenfold cross validation paired *t* test 558				(b) Confusion matrix for the ANN with real test cases (558 instance)			
70	25	0	ǀ a = 1, Low QiU	70	25	0	ǀ a = 1, Low QiU
23	403	0	ǀ b = 2, Medium QiU	23	391	12	ǀ b = 2, Medium QiU
0	37	0	ǀ c = 3, High QiU	0	6	31	ǀ c = 3, High QiU

model, ISO 25010 showed its ability to cover QiU aspects in real situations, which is rarely researched.

Never been discussed in QiU context, the statistical measures of performance showed convergent performance between ANN and CNN with clear advantage for ANN when tested on real cases. Moreover, statistical measures of performance showed more balanced predictions when using ANN. Hence, this research recommends integrating ANN with QiU model to develop a QiUPS, a novel recommendation based on the previous empirical evidences.

Researching in rarely explored research area, research results are original and should be treated as empirical indications for the feasibility of predicting QiU using ANNs. The research dataset is considered as one of the research main contributions due to its originality and harmonization with the CAs under study. Based on that, the research recommends using original dataset in similar contexts.

With regard to developing a decision support sstems (DSS) for measuring QiU, the research recommends developing the DSS based on QiUPS architecture, which was sufficiently discussed and applied in this research context.

References

Abe S et al (2006) Estimation of project success using bayesian classifier. In: Proceedings of the 28th international conference on software engineering. ACM, pp 600–603

Ahimbisibwe A, Cavana RY, Daellenbach U (2015) A contingency fit model of critical success factors for software development projects: a comparison of agile and traditional plan-based methodologies. J Enterp Inf Manag 28(1):7–33

Alnanih R, Ormandjieva O, Radhakrishnan T (2012) A new methodology (CON-INFO) for context-based development of a mobile user interface in healthcare applications. In: Pervasive health. Springer, London, pp 317–342

Ardito C, Lanzilotti R, Sikorski M, Garnik I (2014) Can evaluation patterns enable end users to evaluate the quality of an e-learning system? An exploratory study. In: Universal access in human–computer interaction. Universal access to information and knowledge. Springer, New York, pp 185–196

Becker P, Lew P, Olsina L (2012) Specifying process views for a measurement, evaluation, and improvement strategy. Adv Softw Eng. doi:10.1155/2012/949746

Bevan N (2009) Extending quality in use to provide a framework for usability measurement. In: Kurosu M (ed) HCD 2009, LNCS, vol 5619. Springer, Heidelberg, pp 13–22

Burr GW (2015) Experimental demonstration and tolerancing of a large-scale neural network (165,000 synapses) using phase-change memory as the synaptic weight element. IEEE Trans Electron Dev 62(11):3498–3507

Cerpa N, Bardeen MD, Kitchenham B, Verner JM (2010) Evaluating logistic regression models to estimate software project outcomes. Inf Softw Technol 52(9):934–944

Cheng M, Wu Y (2008) Dynamic prediction of project success using evolutionary support vector machine inference model. In: Proceedings of the 25th international symposium on automation and robotics in construction

Craven MW, Shavlik JW (2014) Learning symbolic rules using artificial neural networks. In: Proceedings of the tenth international conference on machine learning, pp 73–80

Deming WE (2000) Out of the crisis. MIT Press, Cambridge

Dwivedi YK et al (2015) Research on information systems failures and successes: status update and future directions. Inf Syst Front 17(1):143–157

El Emam K, Koru AG (2008) A replicated survey of IT software project failures. IEEE Softw 25(5):84–90

El Halees AM (2014) Software usability evaluation using opinion mining. J Softw 9(2):343–349

Gefen D, Straub D (2001) The relative importance of perceived ease-of-use in IS adoption: a study of e-commerce adoption. JAIS 1:1

González JL, García R, Brunetti JM, Gil R, Gimeno JM (2012) SWET-QUM: a quality in use extension model for semantic web exploration tools. In: Proceedings of the 13th international conference on Interacción Persona-Ordenador. ACM, New York, pp 15:1–15:8

Heinrich R (2014) Business process quality. In: Aligning business processes and information systems, vol 22. Springer Fachmedien Wiesbaden, Wiesbaden

Heravi A, Coffey V, Trigunarsyah B (2015) Evaluating the level of stakeholder involvement during the project planning processes of building projects. Int J Project Manag 33(5):985–997

Hoffman T (1999) Study: 85% of IT departments fail to meet business needs. Computerworld 33:24

Hussain A, Mkpojiogu EO (2016) Requirements: towards an understanding on why software projects fail. In: AIP conference proceedings. AIP Publishing LLC

International Organization for Standardization (2011) ISO/IEC 25010:2011. http://www.iso.org/iso/catalogue_detail.htm?csnumber=35733. Accessed 5 Jan 2017

Jan SR et al (2016) Issues in global software development (communication, coordination and trust)—a critical review. Training 6(7):8

Jørgensen M, Moløkken-Østvold K (2006) How large are software cost overruns? A review of the 1994 CHAOS report. Inf Softw Technol 48:297–301

Kim J, Jeong DH, Lee D, Jung H (2015) User-centered innovative technology analysis and prediction application in mobile environment. Multimed Tools Appl 74(20):8761–8779

La HHJ, Kim SDS (2013) A model of quality-in-use for service-based mobile ecosystem. In: 2013 1st international workshop on the engineering of mobile-enabled systems (MOBS). IEEE, New York, pp 13–18

Lippert SK, Govindarajulu C (2015) Technological, organizational, and environmental antecedents to web services adoption. Commun IIMA 6(1):14

Liu B, Lin J, Sadeh N (2014) Reconciling mobile app privacy and usability on smartphones: could user privacy profiles help? In: Proceedings of the 23rd international conference on world wide web. ACM, pp 201–212

Mizuno O, Hamasaki T, Takagi Y, Kikuno T (2004) An empirical evaluation of predicting runaway software projects using Bayesian classification. Springer, Berlin

Oliveira J, Tereso A, Machado RJ (2014) An application to select collaborative project management software tools. New perspectives in information systems and technologies, vol 1. Springer, New York, pp 467–476

Orehovački T, Granić A, Kermek D (2013) Evaluating the perceived and estimated quality in use of Web 2.0 applications. J Syst Softw 86(12):3039–3059

Osman NB, Osman IM (2013) Attributes for the quality in use of mobile government systems. In: 2013 International conference on computing, electrical and electronics engineering (ICCEEE), pp 274–279

Oztekin A et al (2013) A machine learning-based usability evaluation method for e-learning systems. Decis Support Syst 56:63–73

Reyes F, Cerpa N, Candia-Véjar A, Bardeen MD (2011) The optimization of success probability for software projects using genetic algorithms. J Syst Soft 84(5):775–785

Sainath TN et al (2015) Deep convolutional neural networks for large-scale speech tasks. Neural Netw 64:39–48

Schmidhuber J (2015) Deep learning in neural networks: an overview. Neural Netw 61:85–117

Smite D (2007) Project outcome predictions: risk barometer based on historical data. In: International conference on global software engineering (ICGSE 2007), pp 103–112

Verner JM, Evanco WM, Cerpa N (2007) State of the practice: how important is effort estimation to software development success? Inf Softw Technol 49:181–193

Wang Y (2007) Prediction of success in open source software development. Master, University of California

Woodroof J, Kasper GM (1998) A conceptual development of process and outcome user satisfaction. In: Garrity EJ, Saunders GL (eds) Information system success measurement. Idea Publishing Group, Hershey, pp 122–132