Neural Network with Multiple Training Methods for Web Service Quality of Service Parameter Prediction

Lov Kumar
NIT Rourkela, India
lovkumar505@gmail.com

Ashish Sureka
ABB Corporate Research, India
ashish.sureka@in.abb.com

***Abstract*—Web services have several Quality of Service (QoS) properties. Some of the QoS parameters for web services are availability, response time, throughput, modularity, reliability, and interoperability. A client of a web service can have several web services with similar functionality but different QoS properties for application integration. QoS properties play a decisive factor in selecting the best web services from amongst services having similar functionality. Often QoS parameters are not available, not easy to compute or outdated. We present a method to estimate the QoS parameters of web services from the information contained in web service interfaces. We propose a method based on extracting several data, procedural and structural quantity metrics from the web service interfaces and using them as predictors for estimating the QoS properties. We apply neural network method with 6 different training methods for building a predictive model. Our results demonstrate that the proposed approach is effective. Our experimental results reveal that the structural quality metrics outperforms the procedural and data quality metrics in-terms of the RMSE (Root-Mean-Square Error) performance metric. We conclude that the NLM method (Neural Network with Levenberg-Marquardt training method) out performs five other popular neural network training methods.**

Keywords*—*Software Engineering, Source Code Metrics, Neural Networks, Machine Learning, Quality of Service (QoS), Web Services

I. Research Motivation and Aim

Service-oriented computing is a paradigm which facilitates increase of homogeneity and reduction of heterogeneity between information systems through the usage of standards [1][2][3]. Web services are web application components within the service-oriented and distributed computing paradigm enabling easier application integration through open web protocols and XML standards [1][2]. The interfaces for web services are published and standard-based and the service can be accessible over the web through a URL. One of the important attributes of web services is that the service provider of the web service should be able to make the web service discoverable and be able to define and describe its interfaces or functionalities [1][2]. A complex distributed application serving as a client for a web service accesses the service through various web protocols such as SOAP (Simple Object Access Protocol), WSDL (Web Service Definition Language) and UDDI (Universal Description, Discovery, and Integration).

Web services providing similar functionality can have different QoS (Quality of Service) properties and hence the clients of web services select the best service for their requirements from amongst the similar functionality web services based on QoS attributes [4][5]. QoS properties of web services such as response time, throughput, availability, reliability, modularity and interoperability applies to both atomic web services as well as composite web services created by combining several atomic web services. QoS parameter values are sometimes not available and may change as the web service implementation and environment changes. QoS parameters of web services can be measured by conducting various types of tests such as sending multiple requests across different times and then averaging the response times. However, estimating the QoS parameters of web services through the source code implementing the web services or the interface describing and defining web services can be useful for both the developers implementing the web services and the clients of such services. Ability to estimate the QoS for web services enables based on source code metrics developers to provide guidance in terms of identifying regions of code that can improve the quality of web services. Similarly, ability to predict the QoS parameters enables a web service client to select the best service based on available web services having similar functionality and for which the QoS values are either not available or outdated.

The research motivation of the work presented in this paper is to investigate correlation between certain source code metrics (implementing the web services) and the QoS parameters and examine whether the QoS parameters can be predicted or estimated using the source code metrics as features in a machine learning model. Specifically the research aim of our study presented in this paper is to study the correlation between 8 structural quality metrics, 19 data quality metrics and 21 procedural quality metrics with 12 QoS parameters. Our aim is to also investigate the application of neural networks with multiple training algorithms for building a predictive model (by using source code metrics as predictors) based on historical data and then predicting the QoS parameters of web services for which the QoS parameters are not known.

II. Related Work

Kumar et al. apply extreme learning machines with different types of kernel methods for estimating the quality of service parameters for web services [4]. There are several substantial differences between the work by Kumar et al. and the study presented this paper. Kumar et al. apply extreme learning machines whereas we investigate the application of neural networks with different training methods. The predictors used by Kumar et al. are Chidamber and Kemerer object oriented metrics and metrics such as Baski & Misra metrics whereas our focus is on examining 48 structural quality, data

quality and procedural quality metrics. The metrics used by us for predicting QoS parameters are novel in context to existing work. Coscia et al. demonstrate that several classic software engineering metrics can be used to predict important web service WSDL document quality attributes [6][7]. Their research reveals that there is a statistically significant correlation between object oriented metrics and WSDL level service metrics [7][6].

Huang et al. present a method for web service selection based on using QoS properties as decisive factors from candidate services having similar functionality [8]. Our work on QoS estimation is motivated by the need to have QoS values of web services available for optimal selection. Another line of related work to our research is on the application of neural network based techniques for predicting software quality and maintainability. Cail et al. present an approach based on neural networks for modeling software reliability [9].

III. Research Contributions

In context to existing work and literature, the study presented in this paper makes the following novel research contributions

1) The study presented in this paper is the first study on investigating the correlation between 8 structural quality metrics, 19 data quality metrics and 21 procedural quality metrics of the source code derived from the web service interfaces implementing the web services with 12 QoS parameters and using the metrics as predictors for the 12 QoS properties in a neural network based model. We examine several variants of neural network algorithm by applying six different types of training methods.
2) We conduct a series of experiments on publicly available real world dataset and present the effectiveness of three types of metrics (structural quality, data quality, procedural quality) and six neural network training methods (gradient descent, gradient descent with momentum, variable learning rate gradient descent, levenberg-marquardt, bayesian regularization, BFGS quasi-newton method) to build predictive models for estimating the QoS parameters. We evaluate the performance of the predictive models using three performance parameters MMRE, RMSE and PRED(25) and compare the performance of our approach and results with previous work on similar problem.

IV. Research Framework

Figure 1 illustrates the research framework defining the multi-step process consisting of source code metrics computation from WSDL interfaces, neural network based predictive model creation, performance evaluation and benchmarking. We use the Harry M. Sneed [10] tool obtained from the author and developer of the tool to compute the structural, data and procedural quantity metrics from the WSDL interfaces. The source code complexity and attributes are reflected in the WSDL interfaces captured by the Harry M. Sneed (HMS) tool because of the mapping between the Java and WSDL constructs. There are predefined mappings[1] between Java construct and WSDL constructs. For example, the service endpoint interface in Java construct are mapped to wsdl:portType. The method in Java construct is mapped to wsdl:operation and parameters are mapped to wsdl:input, wsdl:message and wsdl:part. Similarly, there are mappings for user defined exceptions and primitive types. If there is a Java class which consists of two instance variables as p and q then in the corresponding WSDL file there will be two $xsd : element$ tags with attributes $name$ and $type$. The value for the $name$ attribute will be p and q and the value of the type attribute will be $xsd : int$.

As shown in Figure 1, we use a neural network classification algorithm for training a predictive model. We use a neural network schema in which the number of input nodes and hidden nodes are the same. The number of input nodes is the equal to the number of predictors. For example, if we are using SQM (Structural Quantity Metric) as input then the number of input and hidden nodes are 8. We create a neural network for every QoS parameter prediction and hence there are 12 neural networks for predicting 12 QoS parameters for evaluating one class of metric (such as SQM). The six types of neural network variants used in our experiments are: NGD (gradient descent), NGDM (gradient descent with momentum), NGDX (variable learning rate gradient descent), NLM (levenberg-marquardt), NBR (bayesian regularization) and NNM (BFGS quasi-newton method). We use the well-known and widely used performance evaluation metrics such as MMRE, RMSE and PRED(25). MMRE is magnitude of relative error and RMSE is Root mean square error. PRED(25) specifies the percentage of correct predictions within the 25%. We also compare our proposed approach and results with a previous approach.

V. Experimental Dataset

We conduct experiments on QWS Dataset[2] which is publicly available and used by several web service researchers [11][12]. Conducting experiments on freely and openly available dataset makes our experiments easily reproducible and used for benchmarking or comparison. Al-Masri et al. collected 5000 web services from public sources on the web and performed various types of measurements on the web services. They compute 9 quality of web service attributes such as response time, availability and throughput using commercial benchmark tools. They collected a variety of web services from diverse sources such as service portals, web search engines and UDDI registries. We thus conduct experiments on a diverse dataset which eliminates biases and increases the generalization of our results. Kumar et al. create a subset of 200 web services from the original QWS dataset which could be successfully parsed and reversed engineered to Java files [4] for the purpose of conducting experiments on predictive models for web service QoS parameters. They provide a list of 200 web services[3] used in their experiments and we use the same 200 web services in our experiments so that we can compare our results with the results obtained by Kumar et al. [4].

WSDL2Java[4] is an Eclipse plug-in that can be used to generate Java source files from a WSDL file. Similarly,

[1] https://goo.gl/R4KmvT

[2] http://www.uoguelph.ca/~qmahmoud/qws/

[3] https://goo.gl/yrNyWz

[4] https://goo.gl/Drnx1j

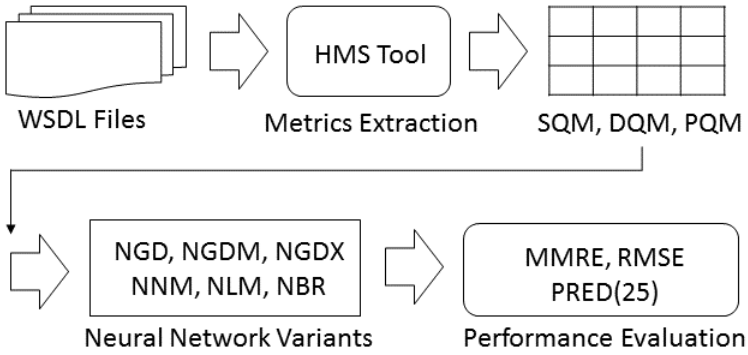

Fig. 1. Research Framework Defining the Multi-Step Process (Source Code Metrics Computation from WSDL Interfaces, Neural Network Based Predictive Model Creation, Performance Evaluation and Benchmarking)

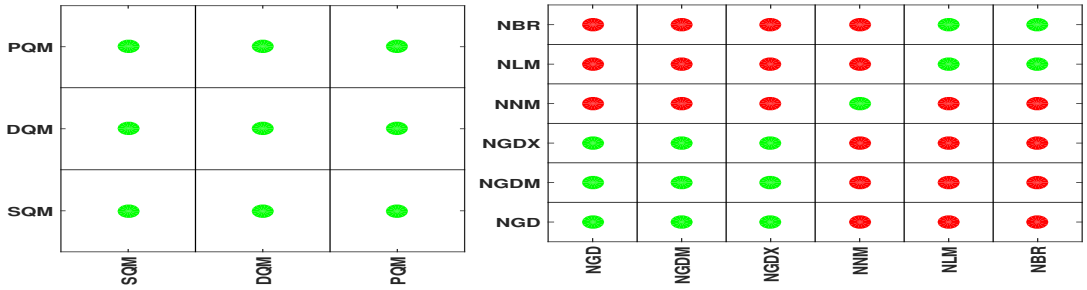

Fig. 2. Wilcoxon Signed Rank Test Results (Statistical Significance Test). A red dot means that H_0 is rejected and a green dot means that H_0 is accepted.

Java2WSDL can be used to generate WSDL from a Java source files. The WSDL2Java and Java2WSDL tools does the mapping between Java and XML types. We use the WSDL2Java tool to convert the 200 WSDL files into their respective Java files. We compute the descriptive statistics of the Java files generated from the WSDL files. The minimum number of Java files are 5 and the maximum is 256. The mean and median values for the number of Java files are 25.25 and 22.5 respectively. The standard deviation, Q1 and Q3 values are 24.73, 13.5 and 25.5 respective. The descriptive statistics shows that the dataset is diverse in-terms of the number of Java files and we have web services of varied sizes in-terms of the number of classes and lines of code.

VI. Source Code Metrics Computation

Table I displays the descriptive statistics for the 8 structural quality metrics, 19 data quality metrics and 21 procedural quality metrics computed by applying the HMS tool on the 200 WSDL files. The structural quality metrics shows the number of modules, includes, classes declared, classes inherited, operations declared, operations specified, procedures declared and interfaces declared. The operation element within an abstract interface of a service (PortType) specifies the syntax of how the methods should be called in the interface. From Table Table I, we observe that the mean number of operations declared and specified are 27.28. Table I shows a wide variation between several metrics. We observe that several metrics has a 0 or null value as such values cannot be extracted from the WSDL interfaces and require the complete source code of the web service. However, we observe variations in properties such as number of arguments, results, parameters, data variables, data references, statements, function references and statement types which influences the various QoS properties.

VII. Experimental Results

A. Comparison of Domain Metrics

Our objective is to conduct a relative performance comparison of the three domain metrics SQM, DQM and PQM. We conduct two types of performance comparison between the three metrics. One comparison is based on analyzing the descriptive statistics of MMRE, RMSE and PRED(25) of 72 models generated from 6 types of training methods and 12 QoS parameters. The other comparison is based on conducting a pair-wise Wilcoxon signed-rank test with a Bonferroni correction method which is to investigate if there is a statistically significant difference in the performance metric due to the characteristics of the metrics or is the observed difference by chance.

Descriptive Statistics Based Comparison: Table II displays the descriptive statistics of the performance of the three domain metrics for 12 QoS parameters and 6 neural network training methods. The descriptive statistics is calculated from $72 = 12X6$ values. The performance comparison is done using MMRE, RMSE and PRED (25). Table II reveals that SQM (shaded in grey color) outperforms DQM and PQM in-terms of the three performance parameters. A lower MMRE and RMSE is better whereas a higher PRED(25) is better. From Table II we observe that the mean value of MMRE and RMSE for SQM is the lowest. The mean value of PRED(25) for SQM is highest whereas the median value is higher than

TABLE I. DESCRIPTIVE STATISTICS OF SOURCE CODE METRICS

	Min	Max	Mean	Median	Std Dev	Q1	Q3	Skewness	Kurtosis
STRUCTURAL QUANTITY METRICS (SQM)									
Number of Modules	2.00	2.00	2.00	2.00	0.00	2.00	2.00	NaN	NaN
Number of Includes	0.00	6.00	1.67	0.00	2.36	0.00	2.00	1.11	2.55
Number of Classes declared	2.00	487.00	35.22	29.00	45.89	15.50	33.00	6.07	52.85
Number of Classes inherited	0.00	0.00	0.00	0.00	0.00	0.00	0.00	NaN	NaN
Number of Operations declared	3.00	485.00	27.28	15.00	53.29	10.00	20.00	5.88	42.45
Number of Operations specified	3.00	485.00	27.28	15.00	53.29	10.00	20.00	5.88	42.45
Number of Procedures declared	0.00	0.00	0.00	0.00	0.00	0.00	0.00	NaN	NaN
Number of Interfaces declared	3.00	217.00	18.81	12.00	30.62	8.00	13.00	4.84	28.18
DATA QUANTITY METRICS (DQM)									
Number of Panels processed	0.00	0.00	0.00	0.00	0.00	0.00	0.00	NaN	NaN
Number of Reports produced	0.00	0.00	0.00	0.00	0.00	0.00	0.00	NaN	NaN
Number of Files declared	0.00	0.00	0.00	0.00	0.00	0.00	0.00	NaN	NaN
Number of Data Bases accessed	0.00	0.00	0.00	0.00	0.00	0.00	0.00	NaN	NaN
Number of Data Views selected	0.00	0.00	0.00	0.00	0.00	0.00	0.00	NaN	NaN
Number of Data Structures	4.00	681.00	56.66	40.00	79.71	24.50	50.00	4.71	29.79
Number of Defined Definitions	1.00	479.00	19.18	1.00	43.04	1.00	29.50	7.37	72.77
Number of Data Variables declared	20.00	1762.00	109.43	70.00	192.18	46.50	82.00	5.49	38.75
Number of Data Variables inherited	0.00	0.00	0.00	0.00	0.00	0.00	0.00	NaN	NaN
Number of Data Constants/Enums declared	1.00	512.00	26.42	11.50	46.35	2.00	35.50	6.73	65.51
Number of Redefinitions (Unions)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	NaN	NaN
Number of Arrays (Vectors)	0.00	31.00	1.27	0.00	3.26	0.00	1.00	5.55	42.34
Number of external Data Elements	0.00	0.00	0.00	0.00	0.00	0.00	0.00	NaN	NaN
Number of different Data Types used	4.00	983.00	87.06	67.00	108.92	39.00	84.00	4.58	30.57
Number of Data References	13.00	818.00	127.12	119.00	121.16	54.00	134.00	3.26	16.62
Number of Arguments / Input Variables	2.00	386.00	25.39	10.00	44.49	8.50	29.00	5.49	38.58
Number of Results / Output Variables	2.00	291.00	17.53	10.00	33.37	6.00	12.00	5.72	39.76
Number of Predicates / Conditional Data	0.00	0.00	0.00	0.00	0.00	0.00	0.00	NaN	NaN
Number of Parameters / Function Arguments	2.00	320.00	21.45	11.00	41.80	7.00	12.00	4.93	30.72
PROCEDURAL QUANTITY METRICS (PQM)									
Number of Statements	46.00	4746.00	381.60	295.50	502.74	156.00	354.50	5.19	37.17
Number of Input Operations	1.00	290.00	17.39	10.00	32.70	6.00	12.00	5.70	39.92
Number of Output Operations	2.00	291.00	17.53	10.00	33.37	6.00	12.00	5.72	39.76
Number of File & Database Accesses	0.00	0.00	0.00	0.00	0.00	0.00	0.00	NaN	NaN
Number of Function References	10.00	100.00	63.24	68.00	27.79	42.00	85.50	-0.33	1.97
Number of Foreign Functions referenced	10.00	100.00	63.24	68.00	27.79	42.00	85.50	-0.33	1.97
Number of Macro References	0.00	0.00	0.00	0.00	0.00	0.00	0.00	NaN	NaN
Number of Macros referenced	0.00	0.00	0.00	0.00	0.00	0.00	0.00	NaN	NaN
Number of If Statements	0.00	0.00	0.00	0.00	0.00	0.00	0.00	NaN	NaN
Number of Switch Statements	0.00	0.00	0.00	0.00	0.00	0.00	0.00	NaN	NaN
Number of Case Statements	0.00	0.00	0.00	0.00	0.00	0.00	0.00	NaN	NaN
Number of Loop Statements	0.00	0.00	0.00	0.00	0.00	0.00	0.00	NaN	NaN
Number of Exception Conditions	0.00	0.00	0.00	0.00	0.00	0.00	0.00	NaN	NaN
Number of Return statements	3.00	485.00	27.28	15.00	53.29	10.00	20.00	5.88	42.45
Number of Control Flow Branches	0.00	0.00	0.00	0.00	0.00	0.00	0.00	NaN	NaN
Number of all Control Statements	0.00	0.00	0.00	0.00	0.00	0.00	0.00	NaN	NaN
Number of Literals in Statements	0.00	97.00	8.00	2.00	14.53	0.00	11.50	3.62	20.19
Number of Nesting Levels (Maximum)	4.00	12.00	5.57	5.00	1.32	5.00	6.00	1.06	5.26
Number of Test Cases (Minimum)	5.00	776.00	45.67	26.00	85.97	17.00	33.00	5.81	41.47
Number of different Statement Types	24.00	2374.00	191.55	148.50	251.39	79.00	178.00	5.19	37.17
Number of Assertions made	0.00	988.00	85.35	57.00	119.79	23.00	91.00	4.40	27.39

TABLE II. DESCRIPTIVE STATISTICS OF THE PERFORMANCE OF THE THREE DOMAIN METRICS FOR 12 QOS PARAMETERS AND 6 NEURAL NETWORK TRAINING METHODS

MMRE							
	Min	**Max**	**Mean**	**Median**	**Std Dev**	**Q1**	**Q3**
SQM	0.10	1.12	0.45	0.43	0.24	0.22	0.63
DQM	0.16	1.31	0.47	0.46	0.24	0.30	0.64
PQM	0.16	1.22	0.45	0.44	0.22	0.30	0.58
RMSE							
	Min	**Max**	**Mean**	**Median**	**Std Dev**	**Q1**	**Q3**
SQM	0.08	0.31	0.18	0.18	0.05	0.16	0.21
DQM	0.08	0.33	0.20	0.20	0.06	0.16	0.23
PQM	0.08	0.32	0.20	0.20	0.05	0.16	0.22
PRED(25)							
	Min	**Max**	**Mean**	**Median**	**Std Dev**	**Q1**	**Q3**
SQM	0.17	0.94	0.55	0.49	0.23	0.38	0.70
DQM	0.17	0.92	0.51	0.45	0.23	0.33	0.66
PQM	0.18	0.92	0.52	0.50	0.22	0.37	0.64

the median value for DQM but lower than the median value for PQM. From Table II we also infer that DQM is the lowest performing metric. The mean of PRED(25) value of DQM is lower than the mean of PRED(25) value of PQM. The mean of MMRE value for PQM is lower than the mean MMRE value of DQM. We a similar trend in the Q1 and Q3 values and hence inferences based on mean and median values also applies to Q1 and Q3.

Wilcoxon Signed-Rank Test with a Bonferroni Correction Based Comparison : We have 72 data points for each of the metric on which pair-wise comparison using the Wilcoxon signed-rank test with a Bonferroni correction can be conducted. We apply a 10 fold cross validation technique to compute one MMRE value (one metric, one QoS parameter and one neural network training algorithm) and generate 72 MMRE values for each metric. The average of the 10 fold cross validation is one MMRE value. Wilcoxon Signed-Rank Test is a pairwise comparison method and can be used to determine if the performance results is a significant indicator of one classifier being better than the other classifier [13]. There

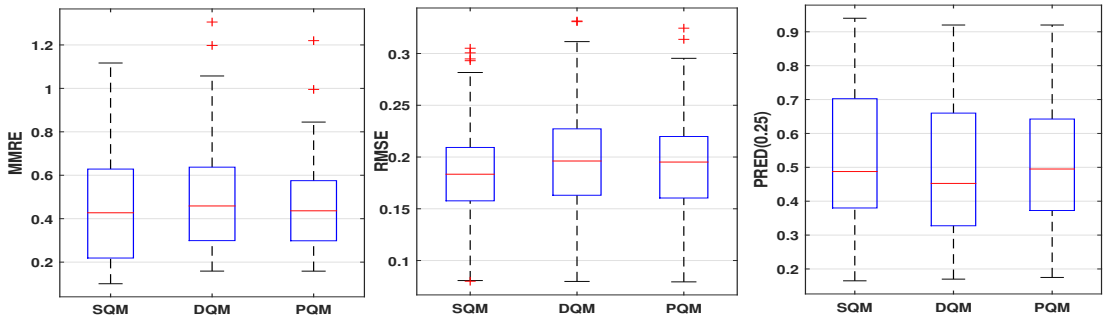

Fig. 3. Box-Plot Displaying the Descriptive Statistics of the Three Domain Metrics Performance in-terms of MMRE, RMSE and PRED(0.25)

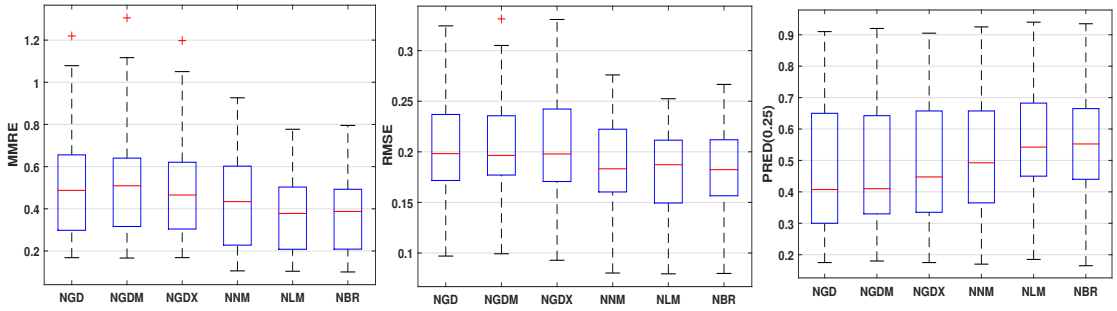

Fig. 4. Box-Plot Displaying the Descriptive Statistics of the Six Different Neural Network Training Method Performance in-terms of MMRE, RMSE and PRED(0.25)

TABLE III. DESCRIPTIVE STATISTICS OF THE PERFORMANCE OF THE SIX NEURAL NETWORK TRAINING TECHNIQUES WITH THREE DOMAIN METRICS AND 12 QOS PARAMETERS

MMRE							
	Min	**Max**	**Mean**	**Median**	**Std Dev**	**Q1**	**Q3**
NGD	0.17	1.22	0.51	0.49	0.26	0.30	0.66
NGDM	0.17	1.31	0.51	0.51	0.26	0.32	0.64
NGDX	0.17	1.20	0.49	0.47	0.25	0.30	0.62
NNM	0.11	0.93	0.43	0.43	0.21	0.23	0.60
NLM	0.10	0.78	0.40	0.38	0.20	0.21	0.50
NBR	0.10	0.80	0.40	0.39	0.20	0.21	0.49
RMSE							
	Min	**Max**	**Mean**	**Median**	**Std Dev**	**Q1**	**Q3**
NGD	0.10	0.32	0.21	0.20	0.06	0.17	0.24
NGDM	0.10	0.33	0.21	0.20	0.06	0.18	0.24
NGDX	0.09	0.33	0.20	0.20	0.06	0.17	0.24
NNM	0.08	0.28	0.18	0.18	0.05	0.16	0.22
NLM	0.08	0.25	0.18	0.19	0.05	0.15	0.21
NBR	0.08	0.27	0.18	0.18	0.05	0.16	0.21
PRED(25)							
	Min	**Max**	**Mean**	**Median**	**Std Dev**	**Q1**	**Q3**
NGD	0.18	0.91	0.48	0.41	0.23	0.30	0.65
NGDM	0.18	0.92	0.48	0.41	0.23	0.33	0.64
NGDX	0.18	0.91	0.50	0.45	0.23	0.34	0.66
NNM	0.17	0.93	0.53	0.49	0.23	0.37	0.66
NLM	0.19	0.94	0.57	0.54	0.21	0.45	0.68
NBR	0.17	0.94	0.57	0.55	0.21	0.44	0.67

are a total of 3 pair-wise comparisons as we have 3 metrics. SQM is compared with DQM and PQM. DQM is compared with PQM. Wilcoxon Signed-Rank Test generates a p-value based on comparing the performance of the two classifiers.

The null hypothesis H_0 is that there is no difference between the two classifiers being compared. The alternate hypothesis H_1 is that the metric characteristics is a significant influencer of the performance. The p-value used as a cut-off

TABLE IV. BENCHMARKING AND COMPARISON OF PROPOSED APPROACH WITH KUMAR ET AL. (REFER [4]) METHOD

MMRE							
	Min	**Max**	**Mean**	**Median**	**Std Dev**	**Q1**	**Q3**
BMS	0.20	1.03	0.54	0.55	0.20	0.37	0.67
HMS	0.05	1.05	0.50	0.53	0.23	0.34	0.65
OOM	0.22	1.03	0.54	0.54	0.19	0.37	0.64
AM	0.05	1.18	0.51	0.48	0.24	0.35	0.69
PCA	0.06	0.96	0.51	0.50	0.21	0.34	0.68
RSA	0.30	1.12	0.58	0.55	0.21	0.37	0.68
SQM	0.10	1.12	0.45	0.43	0.24	0.22	0.63
DQM	0.16	1.31	0.47	0.46	0.24	0.30	0.64
PQM	0.16	1.22	0.45	0.44	0.22	0.30	0.58

to reject the null hypothesis is 0.05. The adjusted p-value after Bonferroni correction is 0.0167. The adjusted p-value is computed by dividing 0.05 from ${}^{metrics}C_2 = 3$ and is a more stringent criteria. Hence, the null hypothesis that "there is no statistically significant difference between the two metrics being compared" is rejected only if $p{-}value \leq \frac{0.05}{3} = 0.0167$. The left hand side of Figure 2 shows the output of the statistical significance test. A red dot means that H_0 is rejected and a green dot means that H_0 is accepted and there is no statistically significant difference between the two metrics. Table II reveals that there is a difference between the performance accuracy of the three metrics but the statistical test results shown in Figure 2 indicates that the values obtained are not statistically significant and can be obtained by chance.

B. Comparison of Neural Network Variants

Our objective is to conduct a relative performance comparison of six neural network variants: NGD, NDM, NGDX, NNM, NLM and NBR. Similar to the method

TABLE V. DETAILED PERFORMANCE MATRIX SHOWING ALL QOS PARAMETERS AND SOURCE CODE METRICS DERIVED FROM WSDL INTERFACES AND NEURAL NETWORK VARIANTS

	MMRE						RMSE						PRED(25)					
STRUCTURAL QUANTITY METRICS (SQM)																		
	NGD	**NGDM**	**NGDX**	**NNM**	**NLM**	**NBR**	**NGD**	**NGDM**	**NGDX**	**NNM**	**NLM**	**NBR**	**NGD**	**NGDM**	**NGDX**	**NNM**	**NLM**	**NBR**
Response Time	0.46	0.52	0.43	0.45	0.44	0.39	0.19	0.21	0.19	0.18	0.19	0.18	0.40	0.38	0.40	0.35	0.39	0.39
Availability	0.17	0.17	0.17	0.16	0.16	0.16	0.15	0.15	0.16	0.15	0.15	0.15	0.90	0.91	0.91	0.92	0.91	0.92
Throughput	0.68	0.70	0.69	0.66	0.67	0.67	0.19	0.20	0.20	0.19	0.19	0.19	0.22	0.21	0.19	0.19	0.19	0.17
Successability	0.20	0.20	0.19	0.18	0.18	0.18	0.18	0.19	0.17	0.16	0.16	0.16	0.88	0.87	0.88	0.90	0.89	0.90
Reliability	0.27	0.32	0.30	0.30	0.31	0.30	0.19	0.24	0.20	0.20	0.20	0.19	0.69	0.65	0.66	0.63	0.65	0.66
Latency	0.41	0.41	0.36	0.23	0.21	0.21	0.10	0.10	0.10	0.08	0.08	0.08	0.37	0.44	0.46	0.57	0.64	0.60
Maintainability	1.08	1.12	1.05	0.93	0.78	0.80	0.30	0.31	0.29	0.28	0.25	0.25	0.29	0.33	0.32	0.33	0.46	0.43
Modularity	0.54	0.56	0.52	0.48	0.45	0.49	0.18	0.18	0.18	0.17	0.16	0.17	0.36	0.34	0.38	0.42	0.46	0.45
Reusability	0.70	0.76	0.76	0.70	0.73	0.72	0.23	0.23	0.23	0.23	0.23	0.22	0.47	0.48	0.55	0.55	0.49	0.55
Testability	0.50	0.59	0.42	0.46	0.44	0.42	0.17	0.18	0.16	0.17	0.15	0.16	0.36	0.39	0.47	0.50	0.57	0.62
Interoperability	0.54	0.51	0.48	0.41	0.37	0.37	0.29	0.28	0.27	0.23	0.21	0.21	0.33	0.38	0.44	0.49	0.59	0.61
Conformity	0.19	0.26	0.23	0.11	0.10	0.10	0.14	0.17	0.18	0.09	0.08	0.09	0.77	0.72	0.75	0.93	0.94	0.94
DATA QUANTITY METRICS (DQM)																		
	NGD	**NGDM**	**NGDX**	**NNM**	**NLM**	**NBR**	**NGD**	**NGDM**	**NGDX**	**NNM**	**NLM**	**NBR**	**NGD**	**NGDM**	**NGDX**	**NNM**	**NLM**	**NBR**
Response Time	0.53	0.48	0.55	0.42	0.40	0.39	0.22	0.21	0.23	0.18	0.18	0.18	0.36	0.39	0.35	0.36	0.40	0.39
Availability	0.17	0.17	0.17	0.17	0.16	0.16	0.15	0.16	0.16	0.15	0.15	0.14	0.91	0.89	0.90	0.92	0.91	0.92
Throughput	0.76	0.69	0.66	0.65	0.65	0.67	0.23	0.20	0.20	0.19	0.20	0.19	0.20	0.19	0.18	0.17	0.19	0.18
Successability	0.19	0.19	0.19	0.18	0.18	0.18	0.19	0.17	0.18	0.16	0.16	0.16	0.88	0.90	0.89	0.90	0.89	0.90
Reliability	0.29	0.31	0.32	0.30	0.30	0.29	0.20	0.21	0.21	0.19	0.20	0.19	0.66	0.65	0.66	0.67	0.67	0.67
Latency	0.34	0.33	0.35	0.21	0.19	0.20	0.10	0.11	0.11	0.08	0.08	0.08	0.56	0.56	0.57	0.65	0.72	0.68
Maintainability	1.06	1.31	1.20	0.74	0.73	0.66	0.31	0.33	0.33	0.23	0.23	0.22	0.29	0.27	0.30	0.45	0.46	0.54
Modularity	0.55	0.59	0.58	0.55	0.52	0.49	0.18	0.19	0.20	0.17	0.17	0.16	0.24	0.26	0.28	0.27	0.29	0.30
Reusability	0.78	0.73	0.76	0.73	0.73	0.72	0.24	0.23	0.24	0.23	0.22	0.23	0.55	0.53	0.49	0.54	0.54	0.55
Testability	0.63	0.52	0.53	0.45	0.38	0.43	0.21	0.19	0.17	0.16	0.15	0.16	0.36	0.42	0.36	0.43	0.55	0.56
Interoperability	0.53	0.53	0.46	0.44	0.38	0.35	0.30	0.28	0.28	0.27	0.23	0.22	0.26	0.25	0.32	0.34	0.45	0.54
Conformity	0.47	0.47	0.46	0.43	0.35	0.45	0.29	0.28	0.28	0.25	0.22	0.26	0.41	0.40	0.41	0.44	0.55	0.44
PROCEDURAL QUANTITY METRICS (PQM)																		
	NGD	**NGDM**	**NGDX**	**NNM**	**NLM**	**NBR**	**NGD**	**NGDM**	**NGDX**	**NNM**	**NLM**	**NBR**	**NGD**	**NGDM**	**NGDX**	**NNM**	**NLM**	**NBR**
Response Time	0.46	0.43	0.48	0.38	0.37	0.39	0.21	0.19	0.19	0.18	0.18	0.18	0.41	0.42	0.37	0.42	0.40	0.40
Availability	0.17	0.17	0.17	0.16	0.16	0.16	0.16	0.15	0.16	0.15	0.15	0.15	0.90	0.92	0.89	0.91	0.91	0.92
Throughput	0.78	0.69	0.68	0.65	0.70	0.66	0.22	0.21	0.20	0.19	0.20	0.19	0.18	0.18	0.19	0.20	0.19	0.18
Successability	0.20	0.19	0.19	0.19	0.18	0.18	0.19	0.17	0.17	0.17	0.16	0.16	0.89	0.88	0.89	0.89	0.90	0.90
Reliability	0.30	0.27	0.31	0.30	0.31	0.30	0.20	0.20	0.21	0.20	0.21	0.20	0.64	0.64	0.66	0.68	0.66	0.65
Latency	0.40	0.35	0.24	0.22	0.21	0.21	0.10	0.10	0.09	0.08	0.08	0.08	0.51	0.56	0.75	0.63	0.70	0.61
Maintainability	1.22	1.00	0.84	0.69	0.49	0.46	0.32	0.29	0.26	0.21	0.17	0.17	0.31	0.33	0.44	0.52	0.59	0.62
Modularity	0.58	0.59	0.57	0.50	0.47	0.35	0.18	0.21	0.20	0.16	0.21	0.12	0.26	0.24	0.25	0.27	0.38	0.46
Reusability	0.75	0.72	0.77	0.72	0.71	0.71	0.24	0.22	0.25	0.22	0.21	0.22	0.53	0.51	0.51	0.55	0.52	0.54
Testability	0.44	0.54	0.46	0.42	0.37	0.41	0.16	0.18	0.16	0.16	0.13	0.23	0.44	0.40	0.43	0.48	0.52	0.60
Interoperability	0.53	0.51	0.54	0.44	0.35	0.32	0.31	0.30	0.29	0.25	0.22	0.20	0.26	0.26	0.25	0.37	0.49	0.52
Conformity	0.46	0.47	0.47	0.46	0.41	0.46	0.27	0.28	0.29	0.26	0.25	0.27	0.41	0.35	0.46	0.43	0.48	0.44

followed for comparing the performance of the three domain metrics, we perform two types of comparison to identify the best performing neural network model. One comparison is based on analyzing the descriptive statistics of MMRE, RMSE and PRED(25) of 36 models generated from 3 types of domain metrics and 12 QoS parameters (and hence $36 = 3 * 12$ models). Similar to the The other comparison is based on conducting a pair-wise Wilcoxon signed-rank test with a Bonferroni correction method.

Descriptive Statistics Based Comparison: Table III displays the descriptive statistics of the performance of the six neural network variants distinguished by the six different types of training method used for 12 QoS parameters and 3 domain metrics. The descriptive statistics is calculated from $36 = 12X3$ values. The performance comparison is done using MMRE, RMSE and PRED(25) metrics. Table III reveals that for NLM the mean MMRE value is 0.40 and the median value is 0.38 which is lowest in comparison to the other neural network variants. Similarly, the RMSE and PRED(25) values for the six neural network variants shows that NLM outperforms all other classifiers. The NLM model dominates in-terms of the mean values of MMRE, RMSE and PRED(25) performance metrics. From Table III, we observe that NBR performance is nearest to NLM. The mean values of MMRE, RMSE and PRED(25) for both NLM and NBR is same but NLM is slightly better in terms of other values such as maximum, median, standard deviation, Q1 and Q3.

Wilcoxon Signed-Rank Test with a Bonferroni Correction Based Comparison: We conduct statistical tests similar to the ones conducted for investigating the performance of the three domain metrics. We have 36 data points for each of the six neural network variant on which pair-wise comparison using the Wilcoxon signed-rank test with a Bonferroni correction is conducted. We conduct a total of 15 pair-wise comparisons as we have 6 neural network variants. In the case of neural network variants, the adjust p-value is even more stringent as the number of models to be compared are 6 in comparison to the number of domain metrics which is 3.

The left hand side of Figure 2 shows the output of the statistical significance test. A red dot means that H_0 is rejected and a green dot means that H_0 is accepted and there is no statistically significant difference between the two metrics. Figure 2 reveals several red dots which shows that there is a statistically significant difference between the performances of the various neural network training methods. From Figure 2, we observe that the NLM model outperforms NNM, NGDX, NGDM, NBR and NGD in-terms of the MMRE, RMSE, PRED(25) values as well as from the perspective of Wilcoxon signed-rank test with a Bonferroni correction based comparison.

C. Detailed Results as Supplementary Information

Figure 3, Figure 4 and Table V displays detailed results in addition to the results provided in Table II and Table III. Table II and Table III shows a summary of results whereas

Figure 3 and Figure 4 displays side-by-side box plots clearly and graphically showing the variability, distribution, outliers, degree of dispersion as well as skewness. Table V shows a detailed performance matrix showing all QoS parameters and source code metrics derived from WSDL interfaces and neural network variants. Table V reveals results for every combination of metric and neutral network training method. We conclude from Table V that the interplay of metric and neural network training method influences the predictive performance of the machine learning model. The model, feature and their specific combination plays a role in influencing the outcome.

D. Comparison with Benchmark

Kumar et al. predict Quality of Service (QoS) parameters of web services using Extreme Learning Machines (ELM) with various kernel methods [4]. They use a total of 37 source code metrics as predictors for the web services QoS parameters categorized into three classes: Chidamber and Kemerer Metrics (OOM), Harry M. Sneed Metrics (HMS) and Baski and Misra Metrics (BMS). They apply two different types of feature selection techniques: Principal Component Analysis (PCA) and Rough Set Analysis (RSA). PCA and RSA results in a subset of metrics after removing irrelevant metrics. Hence there are six sets of metrics which are used as input for experiments by Kumar et al. including the All Metrics (AM) [4]. We conduct a comparison of the 9 different metrics in-terms of the MMRE performance metric. The MMRE for BMS, HMS, OOM, AM, PCA and RSA is calculated using ELM with linear, polynomial and RBF kernel. Table IV shows the average MMRE values by keeping the input metric as constant and taking the average of the MMRE for all the ELM kernel method in-case of BMS, HMS, OOM, AM, PCA and RSA and similarly taking the average of the MMRE for all the neural network training methods in-case of SQM, DQM and PQM.

Table IV reveals that the SQM metric outperforms all the 6 metrics used by Kumar et al. [4]. The best performing metric from Kumar et al. [4] approach is AM having a mean MMRE value of 0.51 and a median MMRE value of 0.48. The PCA metric shows nearly similar performance having a mean MMRE value of 0.51 and a median MMRE value of 0.50. We observe from Table IV that the SQM metric proposed in our study outperforms both AM and PCA metric. From Table IV, we infer that the mean MMRE value of all the three metrics SQM, DQM and PQM outperforms all the six metrics used in previous work on the same dataset and same problem. Hence we conclude that the structural, procedural and data quantity metrics derived from the web service interfaces are better indicators than the classic object-oriented metrics as well as metrics like Distinct Message Ratio (DMR), Data Weight of a WSDL (DW), Message Entropy (ME), Message Repetition Scale (MRS), Operations Per Service (OPS) and Distinct Message Count (DMC) which are computed from Baski and Misra source code metric computation tools.

VIII. Conclusion

SQM metric outperforms DQM and PQM based on three performance metrics, 6 neural network training methods and 12 QoS parameters. PQM performs better than DQM. While SQM metric performs best, the statistical significant test do not favor a strong relationship between the different types of metric characteristics and performance metric. We conclude that while performance metric demonstrates differences between SQM, DQM and PQM metrics, the stringent statistical significant test on the current dataset do not reject the null hypothesis. Our conclusion is that more empirical analysis on a much larger and diverse dataset is required to further understand the influence of SQM, DQM and PQM metrics. NLM neural network model outperforms NNM, NGDX, NGDM, NBR and NGD in-terms of the MMRE, RMSE, PRED(25) values as well as from the perspective of Wilcoxon signed-rank test with a Bonferroni correction based comparison.

We conclude that neural network training method influences the performance outcome of the web service QoS property prediction even under stringent statistical test conditions. Based on a benchmark and comparison exercise with a previous approach, we conclude that neural networks with structural, procedural and data quality metrics outperforms extreme learning based methods with classic object-oriented and Baski and Misra metrics.

References

[1] G. Alonso, F. Casati, H. Kuno, and V. Machiraju, "Web services," in *Web Services*. Springer, 2004, pp. 123–149.

[2] F. Curbera, M. Duftler, R. Khalaf, W. Nagy, N. Mukhi, and S. Weerawarana, "Unraveling the web services web: an introduction to soap, wsdl, and uddi," *IEEE Internet computing*, vol. 6, no. 2, pp. 86–93, 2002.

[3] L. Kumar, S. Rath, and A. Sureka, "Using source code metrics to predict change-prone web services: A case-study on ebay services," in *2017 IEEE Workshop on Machine Learning Techniques for Software Quality Evaluation (MaLTeSQuE)*, 2017, pp. 1–7.

[4] L. Kumar, S. K. Rath, and A. Sureka, "Predicting quality of service (qos) parameters using extreme learning machines with various kernel methods," in *4th International Workshop on Quantitative Approaches to Software Quality*, 2016, p. 11.

[5] L. Zeng, B. Benatallah, A. H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang, "Qos-aware middleware for web services composition," *IEEE Transactions on software engineering*, vol. 30, no. 5, pp. 311–327, 2004.

[6] J. L. Ordiales Coscia, M. Crasso, C. Mateos, and A. Zunino, "Web service interface quality through conventional object-oriented metrics," *CLEI Electronic Journal*, vol. 16, no. 1, pp. 5–5, 2013.

[7] J. L. O. Coscia, M. Crasso, C. Mateos, and A. Zunino, "Estimating web service interface complexity and quality through conventional object-oriented metrics." in *CIbSE*, 2012, pp. 154–167.

[8] A. F. Huang, C.-W. Lan, and S. J. Yang, "An optimal qos-based web service selection scheme," *Information Sciences*, vol. 179, no. 19, pp. 3309–3322, 2009.

[9] K.-Y. Cai, L. Cai, W.-D. Wang, Z.-Y. Yu, and D. Zhang, "On the neural network approach in software reliability modeling," *Journal of Systems and Software*, vol. 58, no. 1, pp. 47–62, 2001.

[10] H. M. Sneed, "Measuring web service interfaces," in *Web Systems Evolution (WSE), 2010 12th IEEE International Symposium on*. IEEE, 2010, pp. 111–115.

[11] E. Al-Masri and Q. H. Mahmoud, "Qos-based discovery and ranking of web services," in *Computer Communications and Networks, 2007. ICCCN 2007. Proceedings of 16th International Conference on*. IEEE, 2007, pp. 529–534.

[12] ——, "Investigating web services on the world wide web," in *Proceedings of the 17th international conference on World Wide Web*. ACM, 2008, pp. 795–804.

[13] S. Garcia and F. Herrera, "An extension on statistical comparisons of classifiers over multiple data sets"for all pairwise comparisons," *Journal of Machine Learning Research*, vol. 9, no. Dec, pp. 2677–2694, 2008.