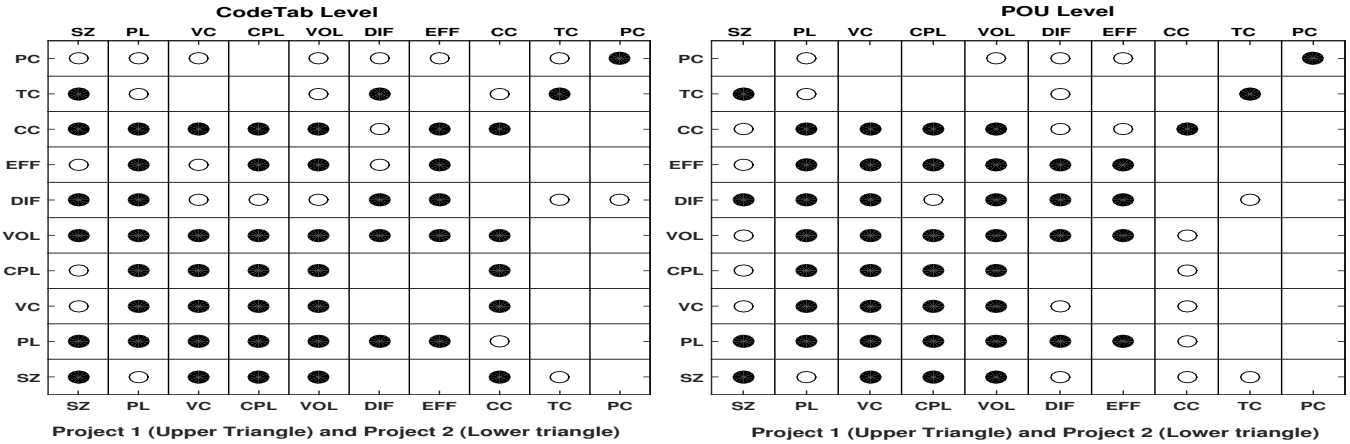


Figure 2: Correlations between 10 Source Code Metrics

We compute the descriptive statistics for the 10 source code metrics for both the projects at the Code Tab and POU Level. Table 2 displays the minimum, maximum, mean, median and standard deviation for all the metrics across both projects. The descriptive statistics in Table 2 describes and characterizes the features of the two versions of the two projects. We observe enough variance and dispersion in majority of the variable values from Table 2 and hence we believe that our experimental dataset provides a good sample or context for our analysis. Table 2 reveals that the cognitive complexity of Project 2 is much higher than the cognitive complexity of Project 1 but the difference between the testing complexity of both the projects is less. Table 2 reveals wide variance in various source code complexity metrics within the same project across Code Tabs and POUs. The variance in code complexity metrics shows that some modules or units are more complex than others. We notice that the range computed as the difference between the minimum and maximum values in the distribution of the ten metrics between the two versions of the same project varies substantially. We use range as one measure of variability but the higher standard deviation for several metrics shows a wide distribution around the mean.

We compute the dependency between 10 code metrics using Pearson Correlation Coefficient. The coefficient of correlation (r) measures the strength and direction of the linear relationship between two variables. Figure 2 displays the correlation between all the metrics for both the projects at the Code Tab and POU level. A black circle denotes a r value between 0.7 and 1.0 indicating a strong positive linear relationship and a white circle denotes a r value between 0.3 and 0.7 indicating a weak positive linear relationship. An empty cell indicates no linear relationship. We did not find instances of negative relationship. Figure 2 shows whether there is a correlation between different metrics and in the subsequent sections we investigate whether there is a cause and effect relationship between the metrics and change proneness.

7. DATA ANNOTATION USING FUZZY CLUSTERING

We define change proneness of a POU or a Code Tab into three categories: High (H), Medium (M) and Low (L). We compute the number of changed lines of code between two versions of the system at the Code Tab and POU level both the projects. Instead of arbitrarily defining a fixed threshold for the number of lines of code to categorize each Code Tab or POU into H, M or L, we apply fuzzy clustering technique (data driven) to automatically derive or determine the threshold and classification of each unit into categories. Table 3 shows the output of the fuzzy clustering algorithm and the center point of the clusters. Table 4 shows the descriptive statistics of Code Tabs and POUs in-terms of categorization into High, Medium and Low lines changed. Table 4 reveals that there are 127, 21 and 9 Code Tabs belonging to the Low, Medium and High category.

Table 3: Fuzzy Cluster Centers [Lines Changed] of POU

	Project 1		Project 2	
Proneness	**CT Level**	**POU Level**	**CT Level**	**POU Level**
C1	4.93	16.02	8.45	23.04
C2	60.30	205.42	97.87	331.91
C3	172.03	505.45	244.49	788.43

Table 4: Descriptive Statistics of Code Tabs and POUs in-terms of Categorization into High, Medium and Low Lines Changed [CP: Change Proneness]

	Project 1				Project 2			
CP	**Code Tab Level**		**POU Level**		**CodeTab Level**		**POU Level**	
	No.	%	No.	%	No.	%	No.	%
Low (L)	127	80.89	48	84.21	166	79.43	59	81.94
Medium (M)	21	13.38	7	12.28	33	15.79	11	15.28
High (H)	9	5.73	2	3.51	10	4.78	2	2.78

8. EXPERIMENTAL RESULTS

8.1 Feature Extraction and Selection

Table 5 shows the output of Principal Component Analysis (PCA) displaying a smaller number of 4 variables accounting for the majority of the variance in the dependent variable. Our objective is to determine the principal components and then use them as predictors for change proneness. Table 5 shows each of the 10 source code metrics score on each of the four principal component. Table 5 shows the

Table 5: Principal Component Analysis

	Project 1								Project 2								
	CodeTab Level				POU Level				CodeTab Level				POU Level				
	PC1	PC2	PC3	PC4	PC1	PC2	PC3	PC4	PC1	PC2	PC3	PC4	PC1	PC2	PC3	PC4	PC5
Size	0.33	0.30	0.32	0.05	0.31	0.43	0.21	0.11	0.35	-0.23	-0.30	0.20	0.34	-0.27	-0.31	-0.06	-0.35
ProgLen	0.36	-0.13	0.02	-0.09	0.36	-0.10	0.04	-0.32	0.39	0.16	0.20	0.06	0.39	0.16	0.14	0.14	-0.10
vocabulary	0.32	-0.30	-0.01	0.55	0.34	-0.26	0.19	0.21	0.36	-0.29	0.04	-0.14	0.35	-0.33	0.24	-0.07	-0.16
Cproglen	0.32	-0.35	0.01	0.43	0.34	-0.30	0.18	0.17	0.36	-0.33	0.08	-0.13	0.35	-0.35	0.25	-0.05	-0.18
Volume	0.36	-0.19	0.00	-0.17	0.36	-0.15	0.02	-0.32	0.40	0.08	0.20	0.03	0.40	0.10	0.15	0.13	-0.11
Difficulty	0.31	0.35	-0.18	0.14	0.34	0.24	-0.15	-0.16	0.27	0.52	0.03	0.08	0.32	0.44	-0.01	0.12	0.16
Effort	0.34	-0.12	-0.12	-0.55	0.33	-0.11	-0.24	-0.45	0.26	0.48	0.32	0.05	0.30	0.47	0.04	0.24	0.15
CC	0.33	-0.16	0.22	-0.38	0.31	-0.21	0.13	0.60	0.33	-0.37	0.02	0.09	0.26	-0.40	-0.08	0.00	0.86
TC	0.22	0.62	0.40	0.12	0.21	0.70	0.21	0.09	0.16	0.20	-0.72	0.47	0.22	0.05	-0.85	-0.11	-0.07
% cmt	0.23	0.32	-0.80	0.02	0.19	0.10	-0.86	0.35	0.18	0.21	-0.45	-0.83	0.14	0.27	0.14	-0.93	0.08
Eigenvalues	6.98	1.36	0.68	0.49	6.97	1.28	0.89	0.43	5.87	1.88	1.19	0.71	5.96	1.67	0.96	0.84	0.42
% Variance	69.81	13.56	6.76	4.89	69.75	12.83	8.87	4.34	58.68	18.76	11.91	7.11	59.59	16.74	9.56	8.41	4.20
Cumulative % variance	69.81	83.37	90.13	95.02	69.75	82.58	91.45	95.79	58.68	77.44	89.35	96.46	59.59	76.34	85.89	94.31	98.50

Table 6: Selected Set of Source Code Metrics using Rough Set Analysis

Project	Level	Source Code Metrics
Project 1	CodeTab	Size, ProgLen, Vocabulary, Cproglen, Difficulty, Effort, TC, %cmt
	POU	Cproglen, Volume, Effort, TC, % cmt,
Project 2	CodeTab	Size, ProgLen, Vocabulary, Cproglen, Volume, Difficulty, TC,% cmt
	POU	Size, Vocabulary, Volume, Difficulty, Effort, CC

eigenvalues which indicates the variance in the data in the direction of the eigenvector. Table 5 shows the correlation between each of the 10 independent variables and the 4 principal components. Table 5 shows that some of the variables are strongly correlated and some are weakly correlated with the four principal components. For example, vocabulary has a strong correlation with PC4 and % cmt has a strong correlation with PC3. Similarly, we observe a strong correlation between TC and PC2. We observe that PC1 has nearly equal correlation with all the variables. The third principal component decreases significantly with increase in % cmt and increases significantly with increase in size and TC.

We apply rough set theory and expectation maximization based clustering algorithm for feature selection [2][6]. Table 6 shows the subset of source code metrics selected after applying the rough set theory based technique. PCA technique falls in the category of feature extraction wherein we mapped the 10 source code metrics feature into a new space consisting of 4 principal components. Rough set analysis falls into the category of feature selection wherein we chose the most informative subset of features from the original set of features. Table 6 reveals that the dimensionality of the attributes has been reduced from 10 to 5 for Project 1 at POU level. Similarly, the dimensionality has been reduced for Project 2 at both the Code Tab and POU level.

8.2 ANN Training and Performance Evaluation

We consider three different subset of metrics as input to design a model for predicting change proneness of PLC programs using ANN with three different type of training algorithm i.e., GD, NM, and LM. The performance of each prediction model is evaluated in terms of two different performance parameters i.e., accuracy and F-measure. Table 7, 8 and 9 shows the resulted confusion matrix after applying the ANN algorithm (we do not show the confusion matrix for Project 2 at POU level due to limited space in the paper).

Table 7: Confusion Matrix (Project 1 [Code Tab Level])

(a) ALL (ANN+GD)

	L	M	H
L	124	3	0
M	18	3	0
H	7	1	1

(b) PCA (ANN+GD)

	L	M	H
L	127	0	0
M	20	0	1
H	9	0	0

(c) RSA (ANN+GD)

	L	M	H
L	121	6	0
M	17	4	0
H	7	2	0

(d) ALL (ANN+NM)

	L	M	H
L	124	2	1
M	15	6	0
H	7	2	0

(e) PCA (ANN+NM)

	L	M	H
L	125	2	0
M	20	0	1
H	9	0	0

(f) RSA (ANN+NM)

	L	M	H
L	125	2	0
M	18	2	1
H	8	1	0

(g) ALL (ANN+GD)

	L	M	H
L	119	7	1
M	10	11	0
H	2	4	3

(h) PCA (ANN+GD)

	L	M	H
L	123	4	0
M	14	5	2
H	7	1	1

(i) RSA (ANN+GD)

	L	M	H
L	121	5	1
M	13	8	0
H	5	1	3

Table 7, 8 and 9 describes the performance of all the classification models and shows the type and number of errors made by the respective classifiers. The column represents the predicted class whereas the rows represents the actual class. Table 7 reveals that the classifier is confusing between the classes M and L indicating areas of improvement. Similarly, in the error matric of Table 9, we observe false positives and negatives between the L and M class. However, a detailed analysis of Tables 7, 8 and 9 a good proportion of correct predictions.

Figure 3, 4, 5, 6 shows the box-plot diagrams for each of the experimental results enabling a visual comparison. The

Table 8: Confusion Matrix (Project 1 [POU Level])

(a) ALL (ANN+GD)

	L	M	H
L	48	0	0
M	4	3	0
H	1	0	1

(b) PCA (ANN+GD)

	L	M	H
L	48	0	0
M	6	1	0
H	2	0	0

(c) RSA (ANN+GD)

	L	M	H
L	46	2	0
M	5	2	0
H	1	0	1

(d) ALL (ANN+NM)

	L	M	H
L	48	0	0
M	4	2	1
H	0	1	1

(e) PCA (ANN+NM)

	L	M	H
L	48	0	0
M	6	1	0
H	1	1	0

(f) RSA (ANN+NM)

	L	M	H
L	47	1	0
M	4	3	0
H	1	0	1

(g) ALL (ANN+GD)

	L	M	H
L	47	1	0
M	2	5	0
H	0	1	1

(h) PCA (ANN+GD)

	L	M	H
L	48	0	0
M	6	1	0
H	1	0	1

(i) RSA (ANN+GD)

	L	M	H
L	47	1	0
M	1	5	1
H	0	2	0

line in the middle of each box represents the median of the performance parameters. We apply 10 fold cross validation for all the combinations and the accuracy and f-measure metric values are summarized in the box blots. Figure 3, 4, 5, 6 contains three different type of box-plots, one for all metrics set, one for PCA, and the last one for RSA. In our study, ANN with three different training algorithm and two different performance parameters have been consider for change-proneness prediction of PLC project and hence 6 different box-plot diagrams have been displayed (one for each combination). Each box-plot diagram is partitioned into three parts: one for all metrics, one for PCA and one for RSA. The box-plot diagrams presents performance of all feature selection methods within a single diagram. Table 10 shows the performance results after applying ANN with three different training algorithm for PLC projects.

Figure 3 and 4 shows the distributional characteristics of the accuracy results. Figure 3 and 4 reveals that the median or middle quartile of the accuracy (marked with a red line) varies significantly across the box-plots. We observe that for project 2 at the code tab level, the box plots are relatively short in comparison to box plot for project 2 at the POU level. A taller box plot for both the projects at the POU level shows that accuracy varies significantly with different folds in the training and testing data. We observe from Figure 5 and 6 an uneven size in various box plots. We observe that for project 1 at the code tab level (ALL), the Q1, Q2 and Q3 are higher for LM in comparison to the Q1, Q2 and Q3 for GD and NM. For project 2 at code tab level (RSA), we notice that the Q1 for LM is higher than the Q3 of both GD and NM. At the POU level (ALL) for project 1, the median value of F-Measure is the same for GD, NM and LM.

We use pairwise t-test to compare the performance of feature selection techniques and classifier training methods. We use pairwise t-test to investigate if the differences between the multiple classifiers in terms of their accuracy is a coincidence or random or they are real [1]. We consider ANN with three different types of training methods to develop

Table 9: Confusion Matrix (Project 2 [Code Tab Level])

(a) ALL (ANN+GD)

	L	M	H
L	156	20	6
M	10	12	4
H	0	1	0

(b) PCA (ANN+GD)

	L	M	H
L	164	33	10
M	0	0	0
H	2	0	0

(c) RSA (ANN+GD)

	L	M	H
L	156	21	3
M	9	12	7
H	1	0	0

(d) ALL (ANN+NM)

	L	M	H
L	157	16	2
M	9	16	3
H	0	1	5

(e) PCA (ANN+NM)

	L	M	H
L	144	17	3
M	21	16	5
H	1	0	2

(f) RSA (ANN+NM)

	L	M	H
L	156	13	2
M	10	20	6
H	0	0	2

(g) ALL (ANN+GD)

	L	M	H
L	160	11	2
M	5	21	3
H	1	1	5

(h) PCA (ANN+GD)

	L	M	H
L	152	22	2
M	13	11	8
H	1	0	0

(i) RSA (ANN+GD)

	L	M	H
L	163	12	3
M	3	20	2
H	0	1	5

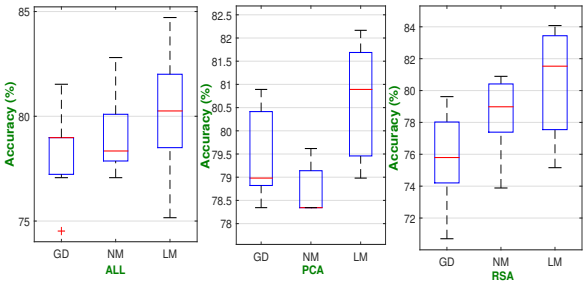
Table 10: Performance Results Based on Accuracy and F-Measure

			Accuracy (%)			F-Measure		
Project	**Level**	**Classifier**	**AM**	**PCA**	***RSA***	**AM**	**PCA**	**RSA**
Project 1	Code Tab	GD	81.53	80.89	79.62	0.93	0.93	0.92
		NM	82.80	79.62	80.89	0.94	0.92	0.93
		LM	84.71	82.17	84.08	0.94	0.93	0.94
	POU	GD	91.23	85.96	85.96	0.97	0.95	0.95
		NM	89.47	85.96	89.47	0.96	0.95	0.96
		LM	92.98	87.72	91.23	0.98	0.96	0.97
Project 2	Code Tab	GD	80.38	78.47	80.38	0.92	0.92	0.92
		NM	85.17	77.51	85.17	0.95	0.91	0.95
		LM	89.00	77.99	89.95	0.96	0.91	0.96
	POU	GD	90.28	83.33	91.67	0.97	0.94	0.97
		NM	93.06	88.89	91.67	0.98	0.96	0.97
		LM	95.83	90.28	93.06	0.99	0.97	0.98

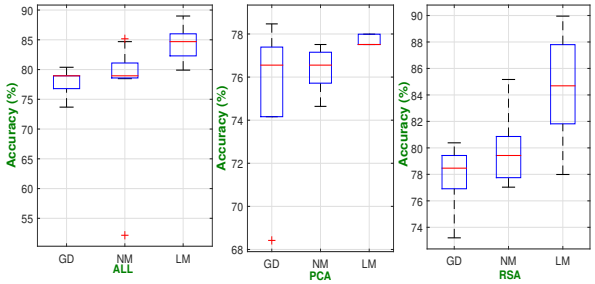
a model to predict change-proneness. We use three different subset of metrics of two different version of PLC projects with two different performance parameters i.e., accuracy and F-Measure. Hence, for each prediction technique a total number of two sets (one for each performance) are used, each with 12 data point [(2 feature selection method + 1 considering all features) * 4 datasets)]. The results of t-test analysis for different performance parameters are summarized in Table 12. Table 11 displays two parts. One part of the Table 11 shows the p-value and the other part shows the mean difference values of performance parameter. Table 12 reveals that for most of the cases there is a significant difference between different approaches as the p-value is lesser than 0.05. When p-value is less than 0.05 then we refer to it as statistically significant (significance level) and we reject the null hypothesis. We observe that the p-value of the GD and NM combination is 0.05. According to the value of mean difference, LM i.e., ANN with LM yields better result compared to other classifiers.

9. THREATS TO VALIDITY

We believe that multiple experiments on multiple dataset can increase the confidence level of the results and confirm the findings. We have applied one of the methods called as

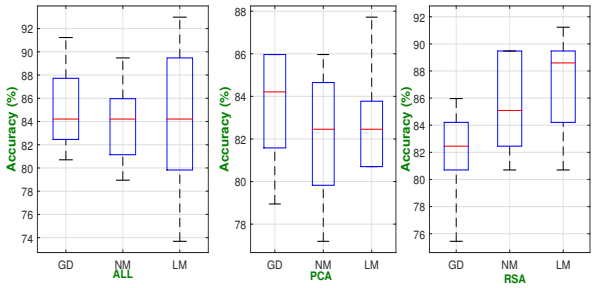


(a) Project 1 (Code Tab Level)

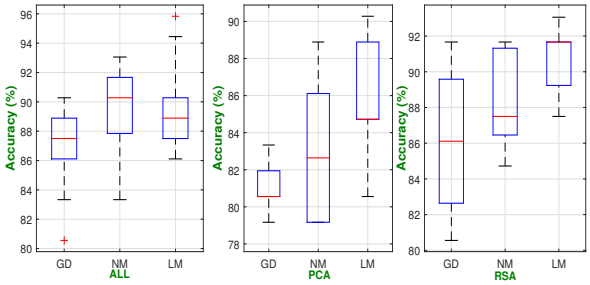


(b) Project 2 (Code Tab Level)

Figure 3: Accuracy (%) [CodeTab Level]



(a) Project 1 (POU Level)



(b) Project 2 (POU Level)

Figure 4: Accuracy (%) [POU Level]

Table 11: t-test - Feature Selection Techniques

Accuracy						
	P-value			**Mean Difference**		
	ALL	PCA	RSA	ALL	PCA	RSA
ALL	NaN	0.00	0.06	0.00	4.80	1.11
PCA	0.00	NaN	0.01	-4.80	0.00	-3.70
RSA	0.06	0.01	NaN	-1.11	3.70	0.00
F-Measure						
	P-value			**Mean Difference**		
ALL	NaN	0.00	0.05	0.00	0.02	0.00
PCA	0.00	NaN	0.007014693	-0.02	0.00	-0.02
RSA	0.05	0.01	NaN	0.00	0.02	0.00

Table 12: Classification Methods

Accuracy						
	P-value			**Mean Difference**		
	GD	NM	LM	GD	NM	LM
GD	NaN	0.05	0.00	0.00	-1.66	-4.11
NM	0.05	NaN	0.00	1.66	0.00	-2.44
LM	0.00	0.00	NaN	4.11	2.44	0.00
F-Measure						
	P-value			**Mean Difference**		
GD	NaN	0.04	0.00	0	-0.00	-0.01
NM	0.04	NaN	0.00	0.00	0	-0.00
LM	0.00	0.00	NaN	0.016	0.00	0

pairwise t-test to statistically compare multiple classifiers in-terms of their accuracy. In order to remove bias, several other methods can be applied to compare the performance of the learning algorithm on multiple datasets. ANN training requires several parameters and an optimal selection of the parameter values can significantly impact the accuracy of the classifier.

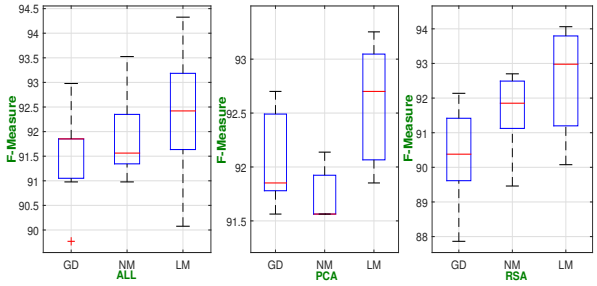
10. CONCLUSION

Our main conclusion is that it is possible to accurately predict the change-proneness of Structured Text programs using source code metrics by employing ANNs and PCA and RSA based feature selection techniques. We conclude that that Artificial Neural Networks with Levenberg-Marquardt training method results in better accuracy (highest median and maximum values of performance parameters) in comparison to other training methods.

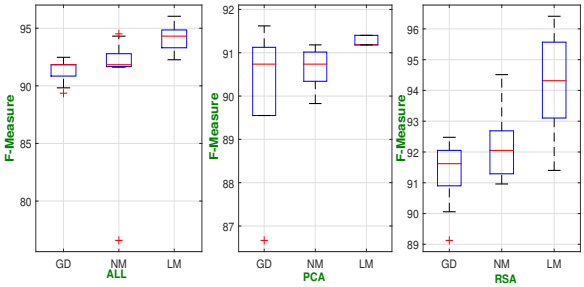
The result obtained from our study indicates that it is possible to identify a reduced subset of source code metrics and attributes based on feature extraction and selection technique for the task of change proneness prediction in the domain of structured text programmable logic control programs. From t-test analysis, it is evident that, there is a significant difference between various models developed using different several set of metrics, due to the fact that p-value being greater than 0.05. However, by judging the value of mean differences, all 10 metrics as a feature set yields better result compared to other approaches. Our results indicate that despite different syntax and language semantics of domain specific languages like ST in comparison to that of general purpose languages, classical source code metrics are a good indicator of change proneness.

11. ACKNOWLEDGEMENTS

We acknowledge the support of our colleagues Raoul Jet-

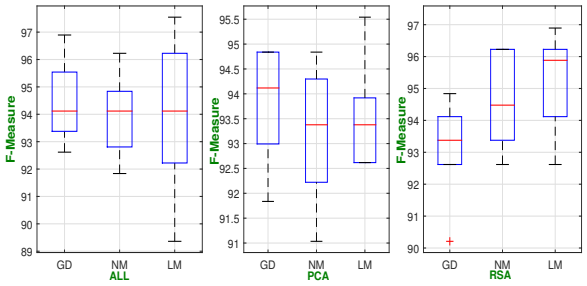


(a) Project 1 (Code Tab Level)

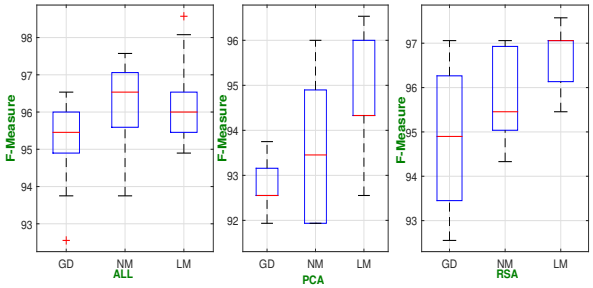


(b) Project 2 (Code Tab Level)

Figure 5: F-Measure (Code Tab Level)



(a) Project 1 (POU Level)



(b) Project 2 (POU Level)

Figure 6: F-Measure (POU Level)

ley and Sreeja Nair in helping us getting access to the experimental data and base code on top of which we implemented our code.

References

[1] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, 7(Jan):1–30, 2006.

[2] Farideh Fazayeli, Lipo Wang, and Jacek Mandziuk. Feature selection based on the rough set theory and expectation-maximization clustering algorithm. *Rough Sets and Current Trends in Computing RSCTC*, pages 272–282, 2008.

[3] Karl-Heinz John and Michael Tiegelkamp. *IEC 61131-3: programming industrial automation systems: concepts and programming languages, requirements for programming systems, decision-making aids*. Springer Science & Business Media, 2010.

[4] A Güneş Koru and Hongfang Liu. Identifying and characterizing change-prone classes in two large-scale open-source products. *Journal of Systems and Software*, 80(1):63–73, 2007.

[5] Lov Kumar, Raoul Jetley, and Ashish Sureka. Source code metrics for programmable logic controller (plc) ladder diagram (ld) visual programming language. In *Workshop on Emerging Trends in Software Metrics*, WETSoM, pages 15–21. ACM, 2016.

[6] Lov Kumar, Santanu Rath, and Ashish Sureka. Predicting quality of service (qos) parameters using extreme learning machines with various kernel methods. In *Workshop on Quantitative Approaches to Software Quality (QuASoQ 2016) co-located to (APSEC 2016)*. CEUR, 2016.

[7] Hongmin Lu, Yuming Zhou, Baowen Xu, Hareton Leung, and Lin Chen. The ability of object-oriented metrics to predict change-proneness: a meta-analysis. *Empirical Software Engineering*, 17(3):200–242, 2012.

[8] FJ Malian, JLCM Barbancho, C Leon, A Malian, and A Gomez. Using industrial standards on plc programming learning. In *Control & Automation, 2007. MED'07. Mediterranean Conference on*, pages 1–6. IEEE, 2007.

[9] A. Nair. Product metrics for iec 61131-3 languages. In *Conference on Emerging Technologies Factory Automation (ETFA)*, pages 1–8, Sept 2012.

[10] Andreas Otto and Klas Hellmann. Iec 61131: A general overview and emerging trends. *Industrial Electronics Magazine, IEEE*, 3(4):27–31, 2009.

[11] Herbert Prähofer, Florian Angerer, Rudolf Ramler, Hermann Lacheiner, and Friedrich Grillenberger. Opportunities and challenges of static code analysis of iec 61131-3 programs. In *ETFA*, pages 1–8. IEEE, 2012.

[12] Daniele Romano and Martin Pinzger. Using source code metrics to predict change-prone java interfaces. In *Conference on Software Maintenance (ICSM)*, pages 303–312. IEEE, 2011.

[13] Nieke Roos. Programming plcs using structured text. In *International Multiconference on Computer Science and Information Technology*, pages 20–22. Citeseer, 2008.