APSIPA Transactions on Signal and Information Processing, 2022, 11, e9

Original Paper

Predicting Pair Success in a Pair Programming Eye Tracking Experiment Using Cross-Recurrence Quantification Analysis

Maureen M. Villamor[1] and Maria Mercedes T. Rodrigo[2*]

[1] *University of Southeastern Philippines, Davao City, Philippines*
[2] *Ateneo de Manila University, Quezon City, Philippines*

ABSTRACT

Pair programming is a model of collaborative learning. It has become a well-known pedagogical practice in teaching introductory programming courses because of its potential benefits to students. This study aims to investigate pair patterns in the context of pair program tracing and debugging to determine what characterizes collaboration and how these patterns relate to success, where success is measured in terms of performance task scores. This research used eye-tracking methodologies and techniques such as cross-recurrence quantification analysis. The potential indicators for pair success were used to create a model for predicting pair success. Findings suggest that it is possible to create a model capable of predicting pair success in the context of pair programming. The predictors for the pair success model that can obtain the best performance are the pairs' proficiency level and degree of acquaintanceship. This was achieved using an ensemble algorithm such as Gradient Boosted Trees. The performance of the pairs is largely determined by the proficiency level of the individuals in the pairs; hence, it is recommended that the struggling students be paired with someone who is considered proficient

*Corresponding author: Maureen M. Villamor, maui@usep.edu.ph.

Received 29 June 2021; Revised 02 February 2022
ISSN 2048-7703; DOI 10.1561/116.00000031

in programming and with whom the struggling student is comfortable working with.

1 Introduction

Pair programming (PP) transpires when two programmers interact and execute programming activities together, either co-located or remotely separated, to build a shared understanding [1, 21]. It is an ideal example of a collaboration paradigm because it embodies all the essential elements of collaborative learning. Therefore, it is frequently used to teach introductory programming courses to investigate how students benefit from it in terms of learning and self-esteem [6, 14].

Chong and Hurlbutt [4] suggest that the gaps in expertise between two individuals have influence on the PP interactions. It means that two programmers jointly engage in the task when they have the same level of expertise, but when the pairs have mixed levels of expertise the programmer with more expertise dominated the interaction. They debunked the driver-navigator myth implying that there is really no consistent division of labor between "driver (i.e., does the typing and writes down a design)" and the "navigator (i.e., performs the strategic planning and monitoring)."

Prior studies have tried to gauge how the quality of social interaction between friends as opposed to non-friends influenced collaborative success. Results have shown that groups consisting of friends may perform better since they already have a history of having collaborated together [18]. The study of Jehn and Shah [8] likewise suggests that being friends may increase commitment to the goals of the group, resulting in more successful collaboration. On the other hand, other studies claim that friendships could reduce performance because friends have an inclination to focus more on socialization rather than the group task [22]. Friendship may not even be necessary for effective collaboration suggesting further that skilled strangers grouped together will perform best because their skills, experience, and being adaptable to the actions of their group mates have equipped them how to work well with other highly skilled individuals [22].

In recent years, dual eye tracking in the context of PP has been explored to study joint attention in collaborative learning situations [9, 15, 16, 20]. For instance, a pair of eye trackers are utilized to study the gaze of two individuals collaborating as they solve a problem and to understand how gaze and speech are coupled [16]. The eye tracking studies that use joint attention to assess collaboration in PP often employ the use of gaze coupling [17], which refers to moments when the partners are looking at the same target.

The concept of joint attention and gaze coupling can be manifested during convergent and divergent phases [23]. The pair is said to be converging in their interaction when the collaborating partners jointly work to understand the code. In this phase, the participants in a pair are focused at the same part of the program in what is considered a "stable" manner, which involves looking at a program by fixations less than a given threshold. On the other hand, a divergent episode of interaction is when the participants are looking at different parts of the program as they try to build their own understanding. During convergent episodes, the pair is said to be "looking together."

Collaborative eye-tracking studies have shown the effectiveness of eye tracking as a tool to predict comprehension and gauge the quality of collaboration through the degree of the pairs' gaze coupling [9, 16, 17]. To some extent, these studies have also explored the pair dynamics that take place in a collaborative task. However, open questions still abound as to the potential factors that may influence the success of the collaboration in the pairs in the context of PP.

The success of the collaboration in programming pairs may be influenced by several factors. Although previous work has already identified the types of interactions that may possibly occur in PP (i.e., leader-follower/driver-navigator, convergence-divergence/engagement-disengagement), none of these studies so far have gone deeper to investigate as to the influence of these pair dynamics to the success of the programming pairs. This research is interested, therefore, in exploring pair gaze patterns in PP to determine the impact of these patterns on pair success. The main method that will be used to assess the degree of collaboration in programming pairs in an eye-tracking setup is called Cross-Recurrence Quantification Analysis (CRQA) [33].

CRQA can be used to analyze collaborative eye-tracking. It takes two disparate trajectories of the same information (e.g., two fixation sequences from different collaborators in a PP setup) as input and performs a test of "closeness" between all pairs of points of the two trajectories. This process is visualized using a cross-recurrence plot (CRP), which is essentially just an $N \times N$ matrix that shows and compares if the two trajectories visit identical sections in the phase space. A gaze cross-recurrence happens, for instance, when two fixations from different sequences fall within a provided threshold of each other using some distance metric.

CRQA and CRPs are not entirely new concepts. These have been used in previous works by Cherubini *et al.* [3] and Zheng *et al.* [34] whose findings revealed that cross-recurrence is positively related to team performance. In a study about solving Tangram puzzles by pair, Kuriyama *et al.* [11] showed that cross-recurrence is higher in successful pairs than in unsuccessful pairs. Jermann *et al.* [9] used CRPs to distinguish between a "good" and a "bad" pair, which correlated to a good and bad collaboration quality. Nüssli [14] detected

that there is a decreased level of gaze coupling for a pair with a bad collaboration flow. In prior studies conducted by Villamor and Rodrigo, they used CRQA to characterize collaboration patterns based on prior knowledge [24], degree of acquaintanceship [25], both prior knowledge and acquaintanceship [27]; as well as determine leader-follower patterns [26].

In the larger context, the main objective of this research is to determine the characteristics that are indicative of a successful collaboration in the context of pair program tracing and debugging, as assessed through the pairs' eye tracking data. It began with the investigation of the individual behaviors in the pairs to discern who between these individuals contributed mostly to the success of the pairs and what practices these individuals exercised that led them to be more successful. The work then expanded to the analysis of the pairs to distinguish what separated successful pairs from unsuccessful pairs, where success was measured in terms of debugging scores. These individual and pair patterns were then related to the success of the pairs. Finally, attempts were made to create a model capable of predicting individual success in the pairs as well as pair success.

Specifically, the entirety of this research sought answers to the following questions: (1) What are the characteristics of the eye gaze patterns of the individuals in the pairs? (2) What are the characteristics of the eye gaze patterns of the pairs of programmers? (3) How do these individual and pair gaze patterns relate with pair success? (4) Can a model be created that can predict individual success in the pairs and pair success? The results of this study could enlighten us as to the predictors of a successful collaboration. The results could aid educators to determine the best pairing methods in their classes to help their students persevere and succeed in their introductory programming courses by encouraging them to adopt the best practices employed by the more successful individuals in the pairs.

The first two questions have already been answered, which can be found in these papers: [29–31]. Hence, this paper will cover only part of the third question, specifically the relationship of pair gaze patterns to success. The relationship of individual gaze patterns to pair success was already previously answered in [29]. This paper will also provide answers to the fourth question uncovering the results obtained that led to the creation of the model that will determine the potential predictors for PP success as a follow-up on the previous work conducted [28], where an attempt was made to create an initial model using CRQA metrics to predict success in PP. The initial findings revealed that using CRQA metrics alone as model input features is not sufficient. The model improved with the addition of the pairs' prior knowledge as input. However, the prediction accuracy is still not acceptable using Logistic Regression, which turned out to be a better model than Naïve Bayes. In this study, feature selection was applied using different filter and wrapper approaches, which were not used previously. Hence, this study endeavors to determine whether

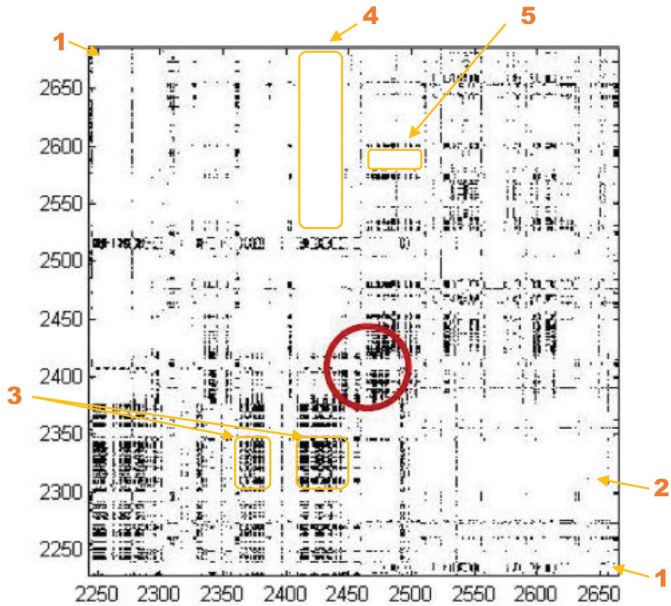

Figure 1: An example of a cross-recurrence plot.

there would be an improvement in the model in predicting success in a PP eye-tracking setup.

2 Literature Review

2.1 Gaze Cross-Recurrence Plot

Figure 1 shows an example of a CRP. If two fixations from different collaborators are deemed to be recurrent according to some threshold, these are represented as a black point or pixel, which is plotted against the horizontal and vertical dimensions of the CRP. The CRP's horizontal and vertical axes signify the time intervals in seconds. In Figure 1, for example, both collaborators commenced at about 2250 seconds past the starting time of the experiment, and at about 2467 seconds (horizontal: collaborator A) and 2440 seconds (vertical: collaborator B), both of them happen to look at the same areas on the screen. This is indicated by the pixelated portions inside the red-bordered circle in the figure.

Different textures, each with corresponding interpretations, may be identified on a CRP. For example, rectangular clusters consisting of horizontal and vertical lines reflect that some states either do not change or change very

Table 1: CRQA metrics and brief definition related to eye-tracking.

CRQA metric	Definition
Recurrence rate (RR)	Represents the percentage of cross-recurrent fixations
Determinism (DET)	Refers to the percentage of identical scanpath segments of a given minimal length in the two scanpaths
Average diagonal length (L)	Duration that both systems stay attuned
Longest diagonal length (LMAX)	Gives the longest time where both scanpaths of the two collaborators are synchronized
Entropy (ENTR)	Represents the complexity of the relation between scanpaths of the two eye movement data
Laminarity (LAM)	An indication of prolonged duration of focusing on the same area on screen
Trapping time (TT)	Represents the average time two trajectories stay in the same region

slowly. These are called laminar or "trapped" states. More of these small-scale structures and their interpretations can be found in [13, 33].

Here are some of the textures that may be found on CRPs:

1. Fading portions to the upper left and lower right corner
2. Single and isolated recurrence points
3. Horizontal/vertical lines and rectangular clusters
4. Bands of white space
5. Empty regions
6. Diagonal lines parallel to the main diagonal

CRQA metrics, such as recurrence rate (RR), determinism (DET), average (L) and longest diagonal (LMAX) lengths, entropy (ENTR), laminarity (LAM), and trapping time (TT) can be extracted from the diagonal and vertical dimensions of the CRP. Table 1 outlines the different CRQA metrics relative to eye-tracking. A more comprehensive discussion of these metrics is found in [13, 33].

Collaborator A (TIME: 2467 seconds) Collaborator B (TIME: 2440 seconds)

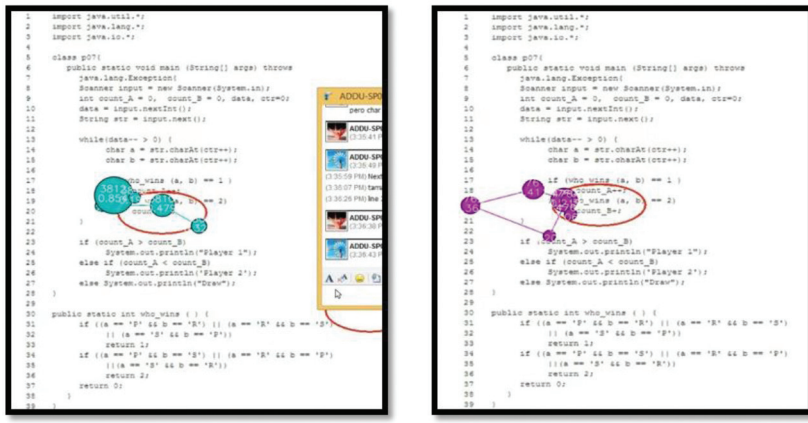

Figure 2: Snapshots showing the location of the fixation points of the two collaborators.

2.2 CRP and Collaborative Eye-Tracking

The relationship of a CRP and collaborative eye tracking is demonstrated using one of the case scenarios in this study. Figure 2 shows snapshots of a program used as one of the actual stimuli in this experiment overlaid with colored circles representing the fixation points of the two collaborators in this pair. The snapshot on the left with aqua-colored circles is from collaborator A, and on the right with purple-colored circles is from collaborator B. Above these snapshots are the times (in seconds) past the starting time of the experiment when these fixations occurred.

At these precise moments, the fixation points are positioned at about the same location on the stimulus making these fixations (from A and B) recurrent according to a defined threshold. In Figure 1, part of the pixelated regions on the CRP enclosed in a red circle informs us that the fixations points of the two collaborators under these times are recurrent.

Figure 3 is the corresponding scan pattern using a line graph of the CRP in Figure 1. The two subplots illustrate the side-by-side comparison of the fixation x-coordinates (upper subplot) and the fixation y-coordinates (lower subplot) of the two collaborators. The aqua and purple line graphs refer to A and B, respectively. The x-axes on the subplots represent the combined timeline of the two collaborators, and the y-axes represent the range of possible values of the fixation x and y coordinates. The sections of the scan patterns enclosed in circles in Figure 3 show the positions in the timeline when these fixations occurred. It can be noted that the fixation x and y coordinates are situated at about 0.5, signifying that these fixation points from the two collaborators are indeed recurrent.

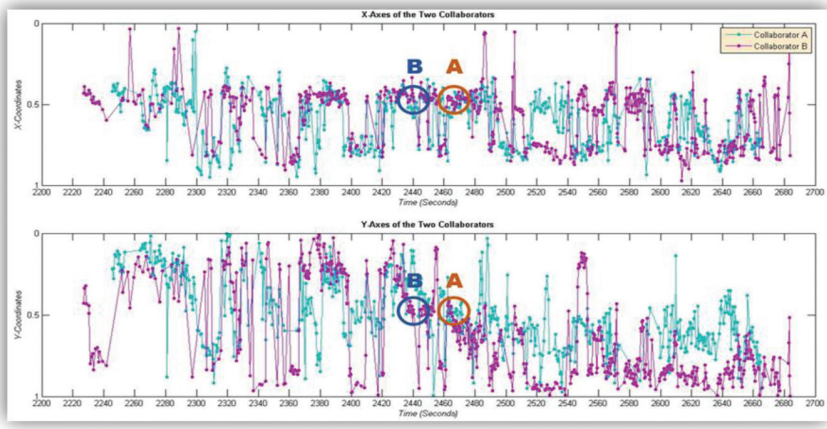

Figure 3: Scan pattern of the two collaborators (upper subplot: fixation x coordinates, lower subplot: fixation y coordinates).

2.3 Some PP Eye-Tracking Studies Using Gaze Cross-Recurrence

Pietinen *et al.* [16] proposed as a metric the number of overlapping fixations to tell how much collaboration is actually done. High fixation duration on the overlapping fixations could inform us if there are problems in understanding. High rate of overlapping fixation could possibly be a sign of efficient collaboration.

Jermann *et al.* [9] used synchronized eye-trackers to assess how programmers collaboratively worked on a segment of code, and they also contrasted a "good" and a "bad" dyad. The analysis that they employed was two-fold: (1) they rated the collaboration quality using a certain rating scheme; and (2) compared the dyad's eye gaze during various moments of the interaction to identify whether gaze indicators are sensitive to contrasts in the rating dimensions. Results showed that high gaze recurrence seems to be typical of a "good" dyad where the flow of interaction is smooth and where partners sustain each other's understanding.

A dual eye-tracking study was also conducted that demonstrated the effect of sharing selection among collaborators in a remote pair-programming scenario [10]. They used gaze cross-recurrence analysis to measure the coupling of the programmers' focus of attention. Their findings showed that pairs who used text selection to perform collaborative references have high levels of gaze cross-recurrence. Gaze cross-recurrence was also highest when selectors accompanied their selections with speech to produce a multimodal reference.

Nüssli [14] analyzed divergence and cross-recurrence and looked at how they are related to the collaboration process. Results showed that collaborators have

a tendency to perform deeper processing when they are working together (i.e., have low divergence), than when they are working alone. It was also found that gaze coupling level is lower for pair with a bad collaboration flow. What was interesting in their results is that gaze coupling does not correlate well with the collaboration flow as there could be bad collaborating pairs that communicate more than a well collaborating one, thus having a higher recurrence level.

3 Methodology

3.1 Participants

The dataset in this study was from the 84 volunteers (56 males and 28 females) that formed a total of 42 pairs. These participants aged 18–23 years old had already taken their college-level fundamental programming courses and were randomly paired with no regard for gender, degree of acquaintanceship, and prior knowledge in programming.

A screening questionnaire was distributed to student volunteers to determine their eligibility to take part in this study. The following were the exclusion criteria: (1) wearing bifocals, trifocals, layered or regression lenses; (2) have difficulty reading a computer screen with contacts and/or eyeglass on; (3) have cataracts; (4) have eye implants; (5) have glaucoma; (6) using a screen reader or magnifier or other assistive technology to use the computer; and (7) if either of the pupils are permanently dilated.

3.2 Procedures

Students who passed the initial screening were asked to take a written program comprehension test and self-efficacy survey to determine the participants' proficiency level and confidence level in programming. The participants were required first to undergo a nine-point eye tracking calibration test prior to the experiment proper. Two *Gazepoint* eye trackers were used to collect the pairs' eye movement data. The pairs were shown 12 programs with bugs as the stimuli and were asked to mark the location of the bugs. There was no need to correct the errors.

A slide sorter program with "Previous," "Reset," "Finish," and "Next" buttons was created to display the 12 programs, each preceded by a program specification. When a bug is found, the participant clicks on the location of the bug using a mouse, and the software then draws an oval to mark it. The participants are free to click any of the buttons and navigate to the next or previous slide at their own pace. The number of bugs in each program is also shown.

Though the pairs were encouraged to work with their partner and use the chat program, they were not informed that this research was primarily about collaboration. No further instructions were given as to how to proceed with

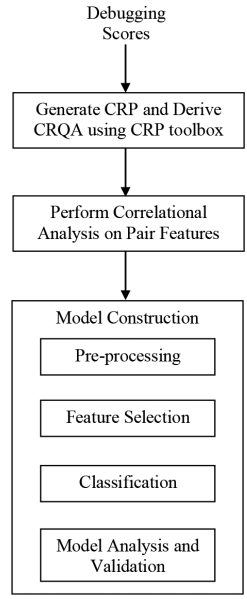

Figure 4: The analysis pipeline.

the task and which problems to solve first. After the experiment, the pair participants were asked to privately and individually fill out a questionnaire to assess the pairs' degree of acquaintanceship. A detailed discussion on data collection, data cleaning, and segmentation can be found in [30].

3.3 Analysis

The analysis pipeline for the pair analysis is shown in Figure 4. Some steps that will not be discussed in this paper (e.g., leader-follower patterns, convergence patterns, pair profiles) are omitted.

Pair success was measured in terms of debugging scores. The pair debugging score was computed by getting the average of the debugging scores of the individuals in the pairs. Two levels of granularity were used in the analysis: *pair-level* (average of all 12 programs) and *case-level* (a program under each pair). For pair-level, a pair is *successful* if the average debugging score for the 12 programs is greater than or equal to the mean score; otherwise, the pair is *unsuccessful*. For case-level, a case is *successful* when both participants in the pairs marked correctly at least 50% of the bugs in each program. A case is *unsuccessful* if both failed to mark all the bugs or if only one participant marked correctly at least 50% of the bugs. One pair was discarded because of the huge fixation count discrepancies between participants, i.e., one had very high fixation counts and the other had very low fixation counts in all 12

programs. Other fixation sequences with very low fixation counts were not good candidates for CRQA and, thus, were not included.

3.3.1 CRP and CRQA

To assess and quantify gaze patterns between participants in a pair, a CRP was constructed for every program for each pair, and CRQA was performed to get the recurrence rate (RR), determinism (DET), average diagonal length (L), longest diagonal length ($LMAX$), entropy ($ENTR$), laminarity (LAM), and trapping time (TT) for each of the 12 programs in each pair. This was done using the CRP toolbox for MATLAB [13].

Using CRQA entails the use of the following parameters: *delay*, *embed*, and *radius* [12, 33]. For this study, no further embedding was done [7] because it involved only two dimensions, which were the fixation x and y coordinates in addition to the timestamps. With an embedding dimension of one, delay was also set equal to one since no points were time delayed [32]. The radius, which was the threshold that determines if two fixation points are recurrent, was set to a default of 10% of the maximal phase space diameter [19].

Threshold adjustments were performed as needed, however, due to varying lengths of the fixation sequences. This was to ensure that the threshold was neither too small nor too large. It is because if the threshold is too small, the recurrence structure of the underlying trajectory may not provide enough information; and if the threshold is too large, almost every point is a neighbor of every other point, which could cause thicker and longer diagonal structures in the CRP as they actually are.

3.3.2 Correlation Analysis

The pair overall debugging scores, and pair per program debugging scores were recorded. The significant metrics identified in the pair analysis were correlated with the pair overall debugging scores and pair per program debugging scores using Pearson's r. These were done to determine the impact of the pair features to the success of the pairs.

3.3.3 Model Construction

All the significant features identified in the pair analysis were first used as inputs to the models to benchmark how well the models work on a complete data set. Pre-processing operations were performed, such as removing dependent variables that significantly correlated with each other to address multicollinearity.

Feature selection or variable elimination was done to make sure that only the relevant features are fed into the models. Filter methods, such as

correlation attribute evaluation and gain ratio evaluation methods from Weka, and wrapper methods such as wrapper subset evaluation method using best-first search from Weka, and Optimize Selection (Evolutionary) operator from RapidMiner were used to generate these relevant features. The selection scheme parameter of the Optimize Selection (Evolutionary) operator was changed from the default "tournament selection" to "non-dominated sorting" while adding the "number of features" as a second performance criterion besides accuracy in order to get the minimum number of features and the maximum predictive accuracy.

The wrapper approaches were trained on two popular machine learning algorithms, Decision Tree and Support Vector Machines. The reason for this choice, aside from being among the commonly used machine learning schemes trained using wrapper approaches, was to use one machine learning classification that will not be used to create the models and another one that will be used to create the models. This was to determine which of these two classification algorithms can produce the most relevant features that will be used to construct the models that can give the best performance.

Since there is no perfect model for all datasets, a model should be chosen that would work best for this particular problem. Three supervised classification algorithms were used. In a study previously conducted [28], Naïve Bayes and Logistic Regression algorithms were used and compared to create a model capable of predicting pair success using the complete data. Between the two, Logistic Regression performed better. Hence, for this study Logistic Regression was used as the baseline algorithm for creating the pair success model on a reduced dataset. A standard or linear Support Vector Machines classifier known to generalize very well, and an ensemble nonlinear method Gradient Boosted Trees known to give better results generally were compared against the baseline algorithm.

The model for predicting pair success used 20-fold cross-validation since it has a small dataset. This was to ensure that each case was used for training and testing an equal number of times. The performance measures used to compare and evaluate the three algorithms were: (1) classification accuracy – the number of correct predictions from all predictions made; (2) Area Under Curve (AUC) – the probability that the classifier ranks a randomly chosen positive example higher than a randomly chosen negative example, (3) precision – the exactness of the model as seen from the confusion matrix; (4) recall – the completeness of the model as seen from the confusion matrix; (5) F Measure – conveys the balance between the precision and recall, and (6) Matthews Correlation Coefficient (MCC) – measures and maximizes the overall accuracy of the classification model and, hence, describes the confusion matrix regardless of the class sizes [2]. However, for the purpose of presentation of the results, the focus will be on the classification accuracy, which will be the basis of the performance comparison of the three algorithms.

Table 2: Relationship between the pair features and debugging scores of the successful and unsuccessful pairs.

Pair features	Pair overall debugging score		Pair per program debugging score	
	r-value	*p*-value	*r*-value	*p*-value
RR	−0.237**	0.000	−0.217**	0.001
DET	−0.200**	0.002	−0.210**	0.001
L	−0.190**	0.003	−0.162**	0.013
ENTR	−0.181**	0.005	−0.169**	0.009
LAM	−0.182**	0.005	−0.198**	0.002
No. of Convergence via Chat	0.248**	0.000		
Prior Knowledge	0.730**	0.000	0.541**	0.000
Degree of Acquaintanceship	0.146*	0.024		

Note: **0.01 level of significance.
*0.05 level of significance.
Empty cells signify no correlation.

4 Results

4.1 Pair Features vs. Success

The identified features considered as pair features are the following: *RR*, *DET* *L*, *ENTR*, *LAM*, *frequency of convergence via chat*, *prior knowledge*, and *degree of acquaintanceship* [21]. The *frequency of chat convergence* was included since the successful pairs were found to significantly chat more than the unsuccessful pairs ($M_{SP} = 1.38/SD_{SP} = 1.36, M_{UP} = 0.93/SD_{UP} = 1.26$) at ($t = -3.346, p = 0.001$). *Prior knowledge* or the proficiency level of the pairs and their *degree of acquaintanceship* were also included as features since these profiles could influence performance of the pairs. The difference in *prior knowledge* (PK) and *degree of acquaintanceship* (DA) between successful and unsuccessful pairs were significant at ($M_{SUCCESSFUL\_PK} = 8.26$, $SD_{SUCCESSFUL\_PK} = 1.89$; $M_{UNSUCCESSFUL\_PK} = 5.43$, $SD_{UNSUCCESSFUL\_PK} = 1.77$; $t\_PK = -11.857$, $p\_PK = 0.000$) and ($M_{SUCCESSFUL\_DA} = 3.83$, $SD_{SUCCESSFUL\_DA} = 0.10$; $M_{UNSUCCESSFUL\_DA} = 3.55$, $SD_{UNSUCCESSFUL\_DA} = 1.05$; $t\_DA = -2.111$, $p\_DA = 0.036$).

Table 2 shows the relationships of these features to the pair overall and per program debugging scores of all the pairs. Tables 3 and 4 show which of these features correlated with the successful pairs and unsuccessful pairs, respectively. The empty cells signify that the respective pair features have no correlation with either the pair overall debugging score or pair per program debugging score.

Table 3: Relationship between the pair features and debugging scores of the successful pairs.

Pair features	Pair overall debugging score		Pair per program debugging score	
	r-value	*p*-value	*r*-value	*p*-value
RR				
DET				
L				
ENTR				
LAM				
No. of Convergence via Chat	0.186*	0.037		
Prior Knowledge	0.483**	0.000	0.237**	0.008
Degree of Acquaintanceship				

Note: **0.01 level of significance.
*0.05 level of significance.
Empty cells signify no correlation.

Table 4: Relationship between the pair features and debugging scores of the unsuccessful pairs.

Pair features	Pair overall debugging score		Pair per program debugging score	
	r-value	*p*-value	*r*-value	*p*-value
RR				
DET				
L				
ENTR				
LAM				
No. of Convergence via Chat				
Prior Knowledge	0.552**	0.000		
Degree of Acquaintanceship	0.230*	0.015		

Note: **0.01 level of significance.
*0.05 level of significance.
Empty cells signify no correlation.

4.2 Predicting Pair Success

The potential predictors for this model are the following: *RR, DET, L, ENTR, LAM, prior knowledge, degree of acquaintanceship*, and *frequency of convergence* [21]. *ENTR* and *DET* were removed after addressing multicollinearity.

Table 5: Performance of the classifiers before pair feature selection.

Feature	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	78.48	0.884	79.52	80.83	79.12	0.568
Support Vector Machine	78.07	0.876	80.71	77.14	77.98	0.561
Gradient Boosted Trees	90.64	0.945	92.25	90.95	90.72	0.814

Table 6: Logistic regression confusion matrix using all the pair features.

	True UNSUCCESSFUL	True SUCCESSFUL
Predicted UNSUCCESSFUL	85	24
Predicted SUCCESSFUL	27	101

Table 7: SVM confusion matrix using all the pair features.

	True UNSUCCESSFUL	True SUCCESSFUL
Predicted UNSUCCESSFUL	88	28
Predicted SUCCESSFUL	24	97

Table 8: GBT confusion matrix using all the pair features.

	True UNSUCCESSFUL	True SUCCESSFUL
Predicted UNSUCCESSFUL	101	11
Predicted SUCCESSFUL	11	114

4.2.1 Performance of the Three Classification Models Using All Pair Features

Table 5 shows the performance of the three classification algorithms when all the features are fed into the model using 20-fold cross validation with "successful" as the positive class. Tables 6–8 show the confusion matrices of each classification algorithm.

4.2.2 Performance of the Three Classification Models Using CRQA Features Only

Since the number of pair features is smaller than the individual features, a brute force feature selection method was applied to obtain only the most relevant features that would give the best performance. The first set of inputs are the CRQA metrics, which were fed one by one to see the impact of each metric on the model. The subsequent set of inputs were done in CRQA pairs.

Table 9: Accuracy table (percentage) using the CRQA metrics as input.

CRQA metrics	Logistic regression	Support vector machine	Gradient boosted trees
Using only *RR* as input	60.80	61.10	59.05
Using only *L* as input	56.74	57.77	54.70
Using only *LAM* as input	62.54	61.93	56.44
Using only *RR* and *L* as input	62.23	62.73	61.52
Using only *RR* and *LAM* as input	60.80	60.68	53.90
Using only *L* and *LAM* as input	62.95	60.68	53.48

Table 10: Accuracy table (percentage) using the Non-CRQA metrics as input.

Non-CRQA metrics	Logistic regression	Support vector machine	Gradient boosted trees
Using only *prior knowledge* as input	76.29	76.06	76.44
Using only *degree of acquaintanceship* as input	53.90	47.90	81.89
Using only *frequency of convergence* as input	64.02	51.97	55.75
Using only *prior knowledge* and *degree of acquaintanceship* as input	76.29	76.44	95.80
Using only *prior knowledge* and *frequency of convergence* as input	77.92	76.10	78.11
Using only *degree of acquaintanceship* and *frequency of convergence* as input	62.73	56.89	83.90
Using all non-CRQA metrics as input	77.92	76.89	94.13

Table 9 shows the accuracy table as a result of feeding the CRQA metrics into the models.

4.2.3 Performance of the Three Classification Models Using Non-CRQA Metrics

The next set of features that were fed into the model are the non-CRQA metrics, such as the pairs' *prior knowledge*, *degree of acquaintanceship*, and *frequency of convergence*. The performance results are shown in Table 10.

The performances of the classifiers using other feature combinations and other performance measures are found in the Appendix.

Table 11: Results of the filter methods for pair feature selection.

Feature	Correlation	Gain Ratio
RR	2	4
L	4	3
LAM	3	6
Prior knowledge	1	1
Degree of Acquaintanceship	6	2
Frequency of Convergence	5	5

Table 12: Results of the wrapper subset evaluation method for pair feature selection.

Feature	Decision tree Number of folds (%)	Support vector machine Number of folds (%)
RR	0 (0%)	0 (0%)
L	0 (0%)	3 (30%)
LAM	0 (0%)	10 (100%)
Prior knowledge	10 (100%)	10 (100%)
Degree of Acquaintanceship	10 (100%)	4 (40%)
Frequency of Convergence	0 (0%)	2 (20%)

4.2.4 Performance of the Models with Filter and Wrapper Methods Applied

The feature subsets were compared against the results of the filter and wrapper methods. Table 11 shows the results of the two filter methods, and Table 12 shows the results of the wrapper subset evaluation method using best first search trained on Decision Tree and SVM classifiers.

Figure 5 shows the deviation chart using the pair features. The highest deviations between successful and unsuccessful pairs come from *prior knowledge*, followed by *RR* and degree of acquaintanceship. The result of the optimize selection (evolutionary) operator on a Decision Tree and SVM learner are shown in Figures 6 and 7, respectively.

5 Discussion

5.1 Pair Features vs. Success

Regardless of the pair success category, all the CRQA metrics have weak negative correlations with the pairs' overall and per program debugging scores as shown in Table 2. However, considering the pair success category, none of the CRQA metrics correlated with the *successful* (Table 3) and *unsuccessful*

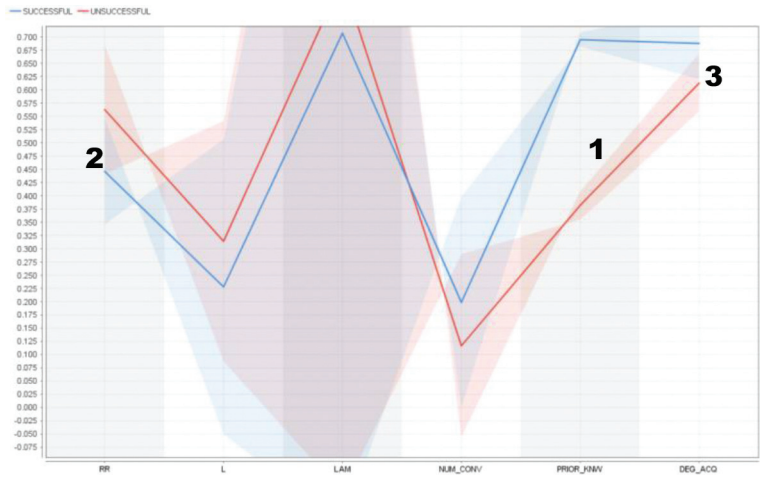

Figure 5: Deviation chart of the complete pair feature set.

Result Individual Selection

All Individuals

Index	Features	Names	accuracy
1	1	DEG_ACQ	0.840
2	1	PRIOR_KNW	0.801
3	2	PRIOR_KNW, DEG_ACQ	0.881
4	3	RR, PRIOR_KNW, DEG_ACQ	0.831
5	3	RR, LAM, PRIOR_KNW	0.768
6	4	RR, NUM_CONV, PRIOR_KNW, DEG_ACQ	0.822
7	4	L, NUM_CONV, PRIOR_KNW, DEG_ACQ	0.843
8	2	PRIOR_KNW, DEG_ACQ	0.881
9	3	L, PRIOR_KNW, DEG_ACQ	0.873
10	3	RR, LAM, PRIOR_KNW	0.768
11	4	RR, NUM_CONV, PRIOR_KNW, DEG_ACQ	0.822
12	3	L, PRIOR_KNW, DEG_ACQ	0.873
13	1	PRIOR_KNW	0.801
14	3	RR, PRIOR_KNW, DEG_ACQ	0.831
15	4	L, NUM_CONV, PRIOR_KNW, DEG_ACQ	0.843
16	3	NUM_CONV, PRIOR_KNW, DEG_ACQ	0.847
17	1	DEG_ACQ	0.840
18	3	NUM_CONV, PRIOR_KNW, DEG_ACQ	0.847
19	1	DEG_ACQ	0.840
20	1	DEG_ACQ	0.840

Selected Individual

ExampleSet (237 examples, 1 special attribute, 2 regular attributes)

Figure 6: Result of optimize selection (evolutionary) on decision tree.

pairs (Table 4). Hence, the CRQA metrics may not be indicators of success, where success is measured in terms of debugging scores.

This confirms previous findings that CRQA might not necessarily correlate with performance since many forms of coordination, including complementarity,

Result Individual Selection

All Individuals

Index	Features	Names	accuracy
1	1	PRIOR_KNW	0.772
2	1	PRIOR_KNW	0.772
3	2	LAM, PRIOR_KNW	0.814
4	2	PRIOR_KNW, DEG_ACQ	0.750
5	3	LAM, NUM_CONV, PRIOR_KNW	0.797
6	5	L, LAM, NUM_CONV, PRIOR_KNW, DEG_ACQ	0.797
7	5	L, LAM, NUM_CONV, PRIOR_KNW, DEG_ACQ	0.797
8	4	RR, L, LAM, PRIOR_KNW	0.797
9	3	LAM, NUM_CONV, PRIOR_KNW	0.797
10	4	LAM, NUM_CONV, PRIOR_KNW, DEG_ACQ	0.797
11	3	L, LAM, PRIOR_KNW	0.810
12	2	PRIOR_KNW, DEG_ACQ	0.750
13	1	PRIOR_KNW	0.772
14	3	L, LAM, PRIOR_KNW	0.810
15	2	LAM, PRIOR_KNW	0.814
16	4	RR, L, LAM, PRIOR_KNW	0.797
17	4	LAM, NUM_CONV, PRIOR_KNW, DEG_ACQ	0.797
18	2	LAM, PRIOR_KNW	0.814
19	2	LAM, PRIOR_KNW	0.814
20	2	LAM, PRIOR_KNW	0.814

Selected Individual

ExampleSet (237 examples, 1 special attribute, 2 regular attributes)

Figure 7: Result of optimize selection (evolutionary) on SVM.

roles, and routines, could possibly cause a decrease in recurrence diagonal structures [5]. This means that CRQA is sensitive in terms of grasping the dynamics of human interactions. Hence, any shift or changes in the coordination and other dynamics can affect its results significantly.

The negative correlations between the CRQA metrics and debugging scores also corroborate with previous findings, where the negative correlations may be interpreted as an effect of task constraint, that is, doing similar things repeatedly may have the opposite of the desired effect [5]. In this case, the pairs were given 12 programs where they had to repeatedly check for errors, which had to be done within one hour. Due to the artificial time constraint imposed during the experiment, the pairs may have the tendency to perform more shotgun debugging all throughout the experiment instead of strategically checking for errors so that they can finish within the time allotted. This may have led to lower debugging scores but higher CRQA metric results.

However, it is also possible that the negative correlations may be a result of the pairs' chatting patterns. It was found that the CRQA metrics have positive correlations with the fixation count (RR: $r = 0.478^{**}/p = 0.000$, L: $r = 0.527^{**}/p = 0.000$, LAM: $r = 0.543^{**}/p = 0.000$) and total fixation time in chat (RR: $r = 0.246^{**}/p = 0.000$, L: $r = 0.243^{**}/p = 0.000$, LAM: $r = 0.288^{**}/p = 0.000$); hence, CRQA metrics may increase when the pairs are chatting. The CRQA of those pairs who chatted have negative relationships

with the pairs' overall and per program debugging scores. Therefore, it is possible that the more the pairs chat, the less remaining time they may have to look for other errors causing a decrease in the debugging scores of the pairs.

In Table 3, *prior knowledge* of successful pairs has moderate and weak positive relationships with their pair overall and per program debugging score, respectively. In Table 4, *prior knowledge* has a strong positive relationship in unsuccessful pairs but only with their pair overall debugging score. This confirms that the performance of the pairs may largely depend on their proficiency level. The higher the proficiency level, the more successful the pairs may become.

The *frequency of chat convergence* correlated only with the successful pairs' pair overall debugging score (Table 3). In contrast, the *degree of acquaintanceship* correlated only with the unsuccessful pairs' pair overall debugging score (Table 4). The former may suggest that if successful pairs engage more in a productive chat, they may achieve much better performance. The latter may also imply that unsuccessful pairs may benefit from being friends with their partners to collaborate more and have better chances of finding more errors together.

To sum it up, though the CRQA metrics can be used to distinguish the collaboration patterns between successful and unsuccessful pairs [30], these measures had no bearing on whether the pairs would be successful or not, where success is measured in terms of debugging scores. Evidence suggests that not all highly gaze coordinated pairs were successful in finding bugs [14]. Conversely, those with more "low CRQA" but had more incidences of convergence turned out to be successful.

Convergence or chatting patterns may play a factor in the pairs' success, but chatting must be done in a productive manner because chatting may impact the performance of the pairs positively or negatively. Lastly, the pairs' proficiency level or prior knowledge in programming may greatly impact the pairs' success.

5.2 Predicting Pair Success

In Table 5, the performance of the Logistic Regression and SVM classifiers are comparable, and the GBT classifier performed way better. The performance of the three classifiers when using only the CRQA metrics was just slightly beyond random guessing as shown in Table 9. Hence, the CRQA metrics alone are incapable of predicting pair success, proving that CRQA metrics are not indicators of pair success.

As for the non-CRQA features, using only *prior knowledge* as input in Table 10, the performances of the three classifiers are almost the same. With only *degree of acquaintanceship* as input, the performances of Logistic Regression and SVM classifiers are either more or less due to chance. The GBT classifier, however, performed satisfactorily with 81.89% accuracy in classifying the obser-

vations according to its correct class. Using only the *frequency of convergence* as input resulted in the classifiers performing only slightly better than chance.

The best performance is achieved using the GBT classifier with *prior knowledge* and *degree of acquaintanceship* as input with an accuracy rating of 95.80%. However, the performance of the GBT classifier using all non-CRQA metrics is not far behind at 94.13% accuracy rating.

The Logistic Regression classifier obtained its best performance at 81.44% accuracy rating with *prior knowledge, degree of acquaintanceship*, and *LAM* as the minimum features as shown in Table A38 in the Appendix. The SVM classifier achieved its best performance at 80.95% accuracy rating also using this feature subset. However, the same performance is obtained using only *prior knowledge* and *LAM* as input, as shown in Table A16 in the Appendix.

In Table 11, *prior knowledge* ranked first in both the correlation and gain ratio filter methods. *Prior knowledge* was likewise considered as the most relevant pair feature using the wrapper subset evaluation methods trained on both Decision Tree and SVM as shown in Table 12.

Figure 5 shows the deviation chart using the pair features. The chart shows the mean values for each feature of both classes. The transparent regions represent the standard deviations of the features for each class. There are three areas where the classes "successful" and "unsuccessful" widely differ but the widest difference comes from *prior knowledge*.

The results of the optimize selection (evolutionary) operator on a Decision Tree and SVM learner are shown in Figures 6 and 7. For the Decision Tree classifier (Figure 6), only two features are needed to obtain its maximum predictive accuracy of 88.1%. These features are *prior knowledge* and *degree of acquaintanceship*. For the Support Vector Machine classifier (Figure 7), two features are needed for the highest accuracy rating of 81.4% to be achieved. These features are *LAM* and *prior knowledge*.

Comparing the results of the brute force method and using the filter and wrapper approaches, the gain ratio filter evaluation method, the wrapper subset evaluation method and the optimize selection (evolutionary) operator trained on a Decision Tree classifier were able to capture the minimum number of predictors, namely, *prior knowledge* and *acquaintanceship*, as produced by the brute force approach. These two predictors gave the best performance using a GBT classifier for predicting pair success.

In summary, an ensemble method such as GBT can create a model capable of predicting pair success with an outstanding performance. The overall performance of this model is: Accuracy: 95.80%, AUC: 0.973, Precision: 94.64%, Recall: 98.54%, F Measure: 96.19%, and MCC: 0.916. The relevant predictors for this model are the pairs' proficiency level and degree of acquaintanceship, whose feature importance are 155.98 (0.70) and 88.75 (−0.70), respectively.

The results confirm the findings in a prior work conducted stating that the performance of the pairs is highly dependent on the proficiency level of the

individuals in pairs [28]. In addition, the degree of acquaintanceship could also be a factor in the pairs' success since how often pairs communicate depends on whether the individuals in pairs are highly or poorly acquainted.

As per the machine learning algorithm used, GBT outperformed LR and SVM in the two models because of the nature of GBT, which "boosts" a weak learner to become better. It got its name "gradient boosting" because the desired outcomes for each observation are set based on the gradient of the error with respect to the prediction.

Basically, it is an additive model whose primary goal is to minimize the overall prediction error by combining a new model with the previous models. It does this by weighing observations and assigning larger weights to observations that are difficult to classify. It then creates and adds new weak learners to the existing ones focusing their training on these hard to classify observations. This is done sequentially until a model is obtained that correctly classifies these observations.

6 Conclusion

The findings reveal that CRQA metrics do not influence the success of the pairs. It contradicts previous studies claiming that pairs who have a higher degree of gaze overlaps reflect better collaboration. The successful pairs in this study, which proved to have collaborated better, as evidenced by their higher debugging scores and better convergence patterns, have lower gaze cross-recurrence rates. On the other hand, unsuccessful pairs who had more incidences where they did not converge have a higher degree of coupled gazes.

Communicating with partners plays a factor in the success of the pairs. However, conversing with partners must be done consistently to work in partnership and be aligned with the performance task goals. The pairs' proficiency level in programming proved to have a considerable impact on the success of the pairs.

The pair success model's predictors that can obtain the best performance are the pairs' *proficiency level* and *degree of acquaintanceship*. This can also be achieved using an ensemble algorithm such as Gradient Boosted Trees.

If we think about collaboration in the CS education classroom, one of the strategies that come to mind is to engage students in PP sessions. Prior research shows that PP has been beneficial to students' learning and self-esteem. However, despite these benefits, studies have shown that PP may not be for everybody. Moreover, it could do more harm than good, as in the more struggling students in programming, which typically happens when PP is not implemented effectively. Hence, to reap these benefits, PP needs to be appropriately implemented.

If the primary goal is to help students who find programming more difficult, it is recommended to look first at the students' profiles being

paired together and not just randomly pair students. Whether they become productive or obtain better performance task scores, the success of the pairs also depends on the individuals being paired. Once pair matching is accomplished, working in partnership and connecting in more conversational processes geared towards the performance task goals must be strongly encouraged.

For PP to be implemented effectively to help students who find programming difficult, CS educators should not pair struggling students together, particularly if these low proficiency students are not well acquainted or do not have any history of working together. On the other side of the spectrum, it is also not wise to pair low proficiency students who are best friends because it may negatively impact their performance [27]. Since the pairs' performance is largely determined by the proficiency level of the individuals in pairs, it is recommended that the struggling students be paired with someone considered proficient in programming and with whom the struggling student is comfortable working.

Although it is not surprising that the proficiency level and degree of acquaintanceship between individuals in the pairs may be critical to the success of the pairs, this study provided empirical validation that these two features are indeed considered the most relevant features to pair success.

It is highly recommended that the results of this study would be validated on students to substantiate the findings. It is also recommended to gather more data. The use of image processing on the CRPs is also recommended to capture important phenomena or meaningful information that relate to specific collaboration patterns. This method was attempted in this study, however, due to the experimental design using 12 different programs with different levels of complexity, this resulted in varying lengths of the fixation sequences of the individuals in the pairs. This in turn required different thresholds, which generated CRPs with mostly unique structures and, thus, made the clustering seem impossible.

It is also recommended that a control and experiment group be used where the joint attention in the experiment group can be manipulated to determine and properly measure its effects on the collaboration of the pairs, and so that the resulting CRPs can be better interpreted. The artificial time limit should be removed and instead of using a chat program where the participants will have to type everything they want to say, a better option would be to use voice chat so that they can communicate more naturally. Freedom from these constraints will likely result in a more valid collaboration.

Biographies

Maureen M. Villamor is a faculty member from the College of Information and Computing, University of Southeastern Philippines. She received her Ph.D. degree in Computer Science in 2018.

Maria Mercedes T. Rodrigo is a senior faculty member from the Department of Information Systems and Computer Science of the Ateneo de Manila University. She is the dissertation adviser of the first author.

Acknowledgment

The authors would like to thank Ateneo de Davao University, Ateneo de Manila University, Ateneo de Naga University, University of Cordillera, University of San Carlos, and University of Southeastern Philippines for allowing us to conduct the eye tracking experiment.

Ethical Standards

The authors assert that all procedures contributing to this work comply with the ethical standards of the relevant national and institutional committees on human experimentation and with the Helsinki Declaration of 1975, as revised in 2008.

Appendix

Result of the Brute Force Pair Feature Selection

Table A1: Performance of the classifiers using only *RR* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	60.80	0.667	62.75	68.81	64.41	0.209
Support Vector Machine	61.10	0.668	61.66	71.07	65.56	0.217
Gradient Boosted Trees	59.05	0.653	58.46	75.12	65.18	0.173

Table A2: Performance of the classifiers using only L as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	56.74	0.618	57.09	70.48	62.47	0.120
Support Vector Machine	57.77	0.615	58.72	80.95	66.91	0.149
Gradient Boosted Trees	54.70	0.524	54.78	72.14	61.74	0.082

Table A3: Performance of the classifiers using only *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	62.54	0.658	66.86	57.38	60.69	0.255
Support Vector Machine	61.93	0.659	69.00	52.98	59.46	0.255
Gradient Boosted Trees	56.44	0.572	56.43	76.67	64.37	0.119

Table A4: Performance of the classifiers using only *RR* and *L* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	62.23	0.695	63.07	67.86	64.31	0.236
Support Vector Machine	62.73	0.681	63.53	73.10	67.03	0.252
Gradient Boosted Trees	61.52	0.607	61.56	71.19	65.16	0.226

Table A5: Performance of the classifiers using only *RR* and *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	60.80	0.664	62.75	68.81	64.41	0.209
Support Vector Machine	60.68	0.657	61.17	72.02	65.57	0.208
Gradient Boosted Trees	53.90	0.589	55.90	63.33	58.80	0.071

Table A6: Performance of the classifiers using only *L* and *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	62.95	0.649	66.70	59.05	61.73	0.262
Support Vector Machine	60.68	0.659	66.62	52.98	58.67	0.227
Gradient Boosted Trees	53.48	0.567	54.66	66.90	59.60	0.058

Table A7: Performance of the classifiers using prior knowledge as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	76.29	0.850	77.24	80.12	78.17	0.525
Support Vector Machine	76.06	0.862	80.76	75.48	76.02	0.519
Gradient Boosted Trees	76.44	0.856	74.45	87.86	79.58	0.534

Table A8: Performance of the classifiers using degree of acquaintanceship as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	53.90	0.576	54.41	75.36	62.70	0.062
Support Vector Machine	47.90	0.588	50.07	76.67	60.30	−0.089
Gradient Boosted Trees	81.89	0.919	81.46	87.26	83.24	0.637

Table A9: Performance of the classifiers using frequency of convergence as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	64.02	0.625	67.93	66.07	65.49	0.281
Support Vector Machine	51.97	0.535	54.70	84.17	63.32	0.002
Gradient Boosted Trees	55.75	0.645	59.06	82.26	65.65	0.100

Table A10: Performance of the classifiers using prior knowledge and degree of acquaintanceship as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	76.29	0.849	77.24	80.12	78.17	0.525
Support Vector Machine	76.44	0.861	85.44	68.10	73.93	0.542
Gradient Boosted Trees	95.80	0.973	94.64	98.54	96.19	0.916

Table A11: Performance of the classifiers using prior knowledge and frequency of convergence as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	77.92	0.851	81.15	78.69	78.87	0.560
Support Vector Machine	76.10	0.856	80.91	72.98	75.84	0.522
Gradient Boosted Trees	78.11	0.869	74.79	90.95	81.49	0.573

Table A12: Performance of the classifiers using degree of acquaintanceship and frequency of convergence as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	62.73	0.600	66.57	63.57	63.55	0.257
Support Vector Machine	56.89	0.599	56.55	77.38	65.01	0.128
Gradient Boosted Trees	83.90	0.933	86.39	83.93	84.52	0.679

Table A13: Performance of the classifiers using the non-CRQA metrics as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	77.92	0.847	81.15	78.69	78.87	0.560
Support Vector Machine	76.89	0.859	83.11	72.38	76.27	0.542
Gradient Boosted Trees	94.13	0.967	91.90	98.33	94.75	0.884

Table A14: Performance of the classifiers using prior knowledge and RR as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	79.28	0.859	81.01	81.79	80.64	0.585
Support Vector Machine	79.32	0.887	82.27	81.01	80.23	0.583
Gradient Boosted Trees	81.74	0.878	83.70	84.17	82.78	0.636

Table A15: Performance of the classifiers using prior knowledge and L as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	80.98	0.898	83.83	83.81	82.17	0.619
Support Vector Machine	78.90	0.868	83.88	76.90	79.02	0.580
Gradient Boosted Trees	82.23	0.881	84.48	82.38	82.89	0.645

Table A16: Performance of the classifiers using prior knowledge and *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	80.15	0.881	83.51	80.95	80.91	0.602
Support Vector Machine	80.95	0.872	84.76	80.48	81.66	0.620
Gradient Boosted Trees	79.66	0.880	81.56	81.55	80.29	0.593

Table A17: Performance of the classifiers using prior knowledge, *RR* and *L* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	77.23	0.886	77.79	80.12	78.14	0.542
Support Vector Machine	77.20	0.877	79.39	78.81	78.44	0.543
Gradient Boosted Trees	80.57	0.881	82.44	82.98	81.77	0.610

Table A18: Performance of the classifiers using prior knowledge, *RR* and *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	79.73	0.875	83.07	80.95	80.63	0.594
Support Vector Machine	80.95	0.869	84.76	80.48	81.66	0.620
Gradient Boosted Trees	83.86	0.887	85.38	86.90	84.99	0.678

Table A19: Performance of the classifiers using prior knowledge, *L*and *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	80.15	0.878	84.05	80.24	80.67	0.603
Support Vector Machine	80.53	0.872	84.68	78.81	80.57	0.612
Gradient Boosted Trees	85.11	0.903	88.70	85.71	85.78	0.704

Table A20: Performance of the classifiers using prior knowledge, *RR*, *L*, and *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	78.52	0.881	80.95	79.40	79.17	0.569
Support Vector Machine	79.28	0.876	82.33	79.52	79.92	0.586
Gradient Boosted Trees	83.90	0.897	87.29	83.21	84.54	0.679

Table A21: Performance of the classifiers using degree of acquaintanceship and *RR* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	62.35	0.654	63.97	69.52	65.76	0.245
Support Vector Machine	61.44	0.662	61.32	73.57	65.89	0.226
Gradient Boosted Trees	76.82	0.844	77.09	83.33	79.30	0.535

Table A22: Performance of the classifiers using degree of acquaintanceship and *L* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	57.73	0.620	59.59	70.48	63.02	0.147
Support Vector Machine	53.86	0.636	54.52	72.02	60.89	0.064
Gradient Boosted Trees	76.86	0.850	77.98	80.00	78.03	0.534

Table A23: Performance of the classifiers using degree of acquaintanceship and *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	59.02	0.648	62.68	60.83	59.94	0.179
Support Vector Machine	59.66	0.660	62.15	51.90	57.78	0..210
Gradient Boosted Trees	82.31	0.883	83.73	82.86	82.63	0.644

Table A24: Performance of the classifiers using degree of acquaintanceship *RR*, and *L* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	60.72	0.659	62.27	66.67	63.12	0.210
Support Vector Machine	61.97	0.665	64.15	73.81	66.61	0.235
Gradient Boosted Trees	74.66	0.818	74.68	81.55	77.12	0.492

Table A25: Performance of the classifiers using degree of acquaintanceship *RR*, and *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	61.52	0.645	63.46	67.98	64.66	0.227
Support Vector Machine	60.19	0.664	60.79	72.86	65.25	0.200
Gradient Boosted Trees	76.82	0.835	76.03	82.38	78.55	0.534

Table A26: Performance of the classifiers using degree of acquaintanceship *L*, and *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	59.43	0.641	62.38	62.550	60.37	0.189
Support Vector Machine	58.03	0.653	59.15	60.12	57.99	0.162
Gradient Boosted Trees	75.00	0.861	75.66	77.38	75.93	0.500

Table A27: Performance of the classifiers using degree of acquaintanceship *RR*, *L*, and *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	60.30	0.662	61.97	65.95	62.65	0.202
Support Vector Machine	59.85	0.666	61.58	70.71	64.29	0.191
Gradient Boosted Trees	75.08	0.820	75.83	78.21	76.61	0.290

Table A28: Performance of the classifiers using frequency of convergence and *RR* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	65.34	0.687	66.77	72.98	68.92	0.304
Support Vector Machine	65.34	0.685	66.77	66.55	66.94	0.307
Gradient Boosted Trees	58.22	0.633	58.19	70.36	62.72	0.156

Table A29: Performance of the classifiers using frequency of convergence and *L* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	61.97	0.665	63.30	70.48	65.80	0.235
Support Vector Machine	60.19	0.655	58.82	76.07	66.90	0.200
Gradient Boosted Trees	55.38	0.582	57.85	66.90	61.06	0.095

Table A30: Performance of the classifiers using frequency of convergenceand *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	63.33	0.697	65.97	63.57	63.08	0.266
Support Vector Machine	59.36	0.680	67.90	44.17	53.40	0.219
Gradient Boosted Trees	56.63	0.643	57.18	66.55	61.71	0.122

Table A31: Performance of the classifiers using frequency of convergence *RR*, and *L* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	65.38	0.689	66.75	72.98	68.99	0.304
Support Vector Machine	65.42	0.689	66.25	69.05	67.72	0.305
Gradient Boosted Trees	56.59	0.605	57.34	64.64	59.95	0.123

Table A32: Performance of the classifiers using frequency of convergence *RR*, and *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	65.38	0.688	66.76	70.71	67.91	0.304
Support Vector Machine	64.51	0.680	66.02	64.88	65.85	0.290
Gradient Boosted Trees	57.73	0.635	58.95	63.81	60.78	0.150

Table A33: Performance of the classifiers using frequency of convergence *L*, and *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	62.08	0.678	63.93	64.29	62.94	0.238
Support Vector Machine	59.36	0.675	67.90	44.17	53.40	0.219
Gradient Boosted Trees	53.14	0.607	54.51	59.88	57.47	0.056

Table A34: Performance of the classifiers using frequency of convergence *RR*, *L*, and *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	66.25	0.696	67.92	70.71	68.44	0.321
Support Vector Machine	59.47	0.692	63.26	56.19	59.32	0.194
Gradient Boosted Trees	57.35	0.626	58.20	58.81	57.98	0.146

Table A35: Performance of the classifiers using prior knowledge, degree of acquaintanceship, and *RR* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	77.61	0.886	79.03	80.95	78.84	0.551
Support Vector Machine	78.45	0.866	80.39	79.64	79.45	0.568
Gradient Boosted Trees	90.23	0.959	91.06	91.90	90.57	0.805

Table A36: Performance of the classifiers using prior knowledge, degree of acquaintanceship, and L as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	78.94	0.869	80.60	81.79	80.29	0.576
Support Vector Machine	77.99	0.864	81.20	77.38	78.73	0.561
Gradient Boosted Trees	88.98	0.957	88.59	91.79	89.53	0.780

Table A37: Performance of the classifiers using prior knowledge, degree of acquaintanceship, and LAM as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	81.44	0.878	83.75	83.45	82.47	0.627
Support Vector Machine	80.95	0.863	84.76	80.48	81.66	0.620
Gradient Boosted Trees	90.64	0.966	90.33	93.45	91.04	0.814

Table A38: Performance of the classifiers using prior knowledge, frequency of convergence, and RR as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	77.61	0.885	78.37	81.43	78.80	0.551
Support Vector Machine	78.94	0.877	80.82	79.64	79.54	0.577
Gradient Boosted Trees	76.79	0.870	80.54	78.81	78.03	0.560

Table A39: Performance of the classifiers using prior knowledge, frequency of convergence, and L as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	78.86	0.880	80.32	82.26	79.96	0.576
Support Vector Machine	78.48	0.869	82.86	75.00	77.88	0.573
Gradient Boosted Trees	81.44	0.887	84.77	82.14	82.37	0.628

Table A40: Performance of the classifiers using prior knowledge, frequency of convergence, and LAM as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	80.57	0.890	81.09	84.05	81.46	0.610
Support Vector Machine	79.32	0.876	83.27	76.43	78.79	0.589
Gradient Boosted Trees	77.99	0.872	80.48	82.14	79.42	0.560

Table A41: Performance of the classifiers using prior knowledge, degree of acquaintanceship, RR, and L as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	77.31	0.885	77.79	80.95	78.47	0.542
Support Vector Machine	76.36	0.873	78.46	77.98	77.52	0.526
Gradient Boosted Trees	88.56	0.958	88.10	91.90	89.18	0.772

Table A42: Performance of the classifiers using prior knowledge, degree of acquaintanceship, RR, and LAM as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	78.90	0.878	82.18	80.12	79.72	0.577
Support Vector Machine	80.11	0.866	83.26	79.64	80.69	0.603
Gradient Boosted Trees	89.81	0.954	90.64	90.95	90.13	0.797

Table A43: Performance of the classifiers using prior knowledge, degree of acquaintanceship, L, and LAM as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	80.61	0.875	82.38	82.62	81.69	0.610
Support Vector Machine	80.11	0.866	83.97	78.81	80.18	0.604
Gradient Boosted Trees	91.06	0.958	91.81	93.45	91.56	0.823

Table A44: Performance of the classifiers using prior knowledge, degree of acquaintanceship, *RR*, *L*, and *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	78.14	0.881	79.58	80.24	79.17	0.560
Support Vector Machine	79.28	0.875	82.08	79.52	79.95	0.585
Gradient Boosted Trees	89.81	0.948	90.91	90.95	90.00	0.797

Table A45: Performance of the classifiers using prior knowledge, frequency of convergence, *RR*, and *L* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	76.82	0.886	77.78	80.83	78.16	0.534
Support Vector Machine	78.48	0.875	79.87	80.36	79.31	0.568
Gradient Boosted Trees	79.28	0.879	81.16	81.43	80.33	0.585

Table A46: Performance of the classifiers using prior knowledge, frequency of convergence, *RR*, and *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	79.28	0.887	80.32	82.38	80.23	0.585
Support Vector Machine	79.77	0.882	82.78	78.10	79.47	0.596
Gradient Boosted Trees	79.24	0.878	81.27	81.19	79.85	0.585

Table A47: Performance of the classifiers using prior knowledge, frequency of convergence, *L*, and *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	80.61	0.889	82.08	82.50	81.31	0.610
Support Vector Machine	78.86	0.875	83.37	76.43	78.65	0.580
Gradient Boosted Trees	80.95	0.897	84.95	80.83	81.45	0.620

Table A48: Performance of the classifiers using prior knowledge, frequency of convergence, *RR*, *L*, and *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	78.48	0.886	79.32	81.55	79.43	0.568
Support Vector Machine	79.73	0.874	82.30	78.69	79.59	0.595
Gradient Boosted Trees	82.16	0.892	85.65	79.76	81.81	0.647

Table A49: Performance of the classifiers using prior knowledge, degree of acquaintanceship, *RR*, *L*, and *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	78.14	0.881	79.58	80.24	79.17	0.560
Support Vector Machine	79.28	0.875	82.08	79.52	79.95	0.585
Gradient Boosted Trees	89.81	0.948	90.91	90.95	90.00	0.797

Table A50: Performance of the classifiers using prior knowledge, degree of acquaintanceship, *RR*, and *L* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	65.00	0.683	66.04	72.14	68.34	0.295
Support Vector Machine	63.30	0.686	65.24	68.33	66.15	0.262
Gradient Boosted Trees	75.11	0.815	74.59	82.38	77.64	0.501

Table A51: Performance of the classifiers using degree of acquaintanceship, frequency of convergence, *RR*, and *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	65.38	0.676	66.69	70.60	67.87	0.304
Support Vector Machine	63.22	0.678	65.68	65.60	65.34	0.263
Gradient Boosted Trees	75.87	0.845	75.33	82.50	78.27	0.518

Table A52: Performance of the classifiers using degree of acquaintanceship, frequency of convergence, L, and LAM as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	60.42	0.674	62.40	61.19	60.51	0.206
Support Vector Machine	59.36	0.678	67.06	45.83	54.29	0.214
Gradient Boosted Trees	77.95	0.866	79.11	81.43	79.12	0.559

Table A53: Performance of the classifiers using degree of acquaintanceship, frequency of convergence, RR, L, and LAM as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	64.13	0.695	64.31	67.74	66.40	0.280
Support Vector Machine	61.14	0.687	64.10	64.17	63.49	0.221
Gradient Boosted Trees	74.58	0.835	74.80	78.93	76.18	0.491

Table A54: Performance of the classifiers using prior knowledge, degree of acquaintanceship, and RR as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	78.45	0.886	78.95	82.26	79.62	0.568
Support Vector Machine	78.07	0.873	80.64	78.10	78.66	0.560
Gradient Boosted Trees	88.94	0.968	88.50	91.90	89.69	0.780

Table A55: Performance of the classifiers using prior knowledge, degree of acquaintanceship, frequency of convergence, and$L$ as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	78.86	0.879	80.32	82.26	79.96	0.576
Support Vector Machine	78.07	0.869	81.86	75.00	77.36	0.564
Gradient Boosted Trees	88.18	0.958	88.93	90.36	88.77	0.763

Table A56: Performance of the classifiers using prior knowledge, degree of acquaintanceship, frequency of convergence, and *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	78.90	0.884	80.12	81.67	79.87	0.576
Support Vector Machine	78.90	0.871	83.10	75.60	78.26	0.581
Gradient Boosted Trees	92.73	0.963	93.09	94.29	93.01	0.856

Table A57: Performance of the classifiers using prior knowledge, degree of acquaintanceship, frequency of convergence, *RR*, and *L* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	77.23	0.887	78.20	80.83	78.45	0.542
Support Vector Machine	78.52	0.876	80.65	78.81	79.02	0.569
Gradient Boosted Trees	89.81	0.951	91.10	91.07	90.25	0.797

Table A58: Performance of the classifiers using prior knowledge, degree of acquaintanceship, frequency of convergence, *RR*, and *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	79.28	0.885	80.32	82.38	80.23	0.585
Support Vector Machine	78.94	0.883	82.61	76.43	78.40	0.580
Gradient Boosted Trees	90.23	0.945	91.30	90.95	90.44	0.805

Table A59: Performance of the classifiers using prior knowledge, degree of acquaintanceship, frequency of convergence, *L*, and *LAM* as input.

Classifier	Acc (%)	AUC	Prec (%)	Recall (%)	F Msr (%)	MCC
Logistic Regression	79.77	0.883	81.85	80.83	80.41	0.594
Support Vector Machine	78.45	0.873	83.21	75.60	78.12	0.572
Gradient Boosted Trees	90.23	0.965	91.39	91.79	90.56	0.805

References

[1] P. Baheti, "Assessing Distributed Pair Programming," in, *Companion of the 17th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications – OOPSLA 02*, 2002.

[2] S. Boughorbel, F. Jarray, and M. El-Anbari, "Optimal Classifier for Imbalanced Data Using Matthews Correlation Coefficient Metric," *PloS One*, 12(6), 2017, e0177678.

[3] M. Cherubini, M. A. Nüssli, and P. Dillenbourg, "This is it!: Indicating and Looking in Collaborative Work at Distance," *Journal of Eye Movement Research*, 3(5), 2010, 1–20.

[4] J. Chong and T. Hurlbutt, "The Social Dynamics of Pair Programming," in, *29th International Conference on Software Engineering (ICSE07)*, 2007, 354–63.

[5] R. Fusaroli, I. Konvalinka, and S. Wallot, *Analyzing Social Interactions: The Promises and Challenges of Using Cross Recurrence Quantification Analysis*, Translational Recurrences. Springer International Publishing, 2014, 137–55.

[6] J. E. Hannay, T. Dybå, E. Arisholm, and D. I. Sjøberg, "The Effectiveness of Pair Programming: A Meta-analysis," *Information and Software Technology*, 51(7), 2009, 1110–22.

[7] J. S. Iwanski and E. Bradley, "Recurrence Plots of Experimental Data: To Embed or Not to Embed?" *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 8(4), 1998, 861–71.

[8] K. A. Jehn and P. P. Shah, "Interpersonal Relationships and Task Performance: An Examination of Mediation Processes in Friendship and Acquaintance Groups," *Journal of Personality and Social Psychology*, 72(4), 1997, 775.

[9] P. Jermann, D. Mullin, M.-A. Nüssli, and P. Dillenbourg, "Collaborative Gaze Footprints: Correlates of Interaction Quality," in, *Connecting Computer-Supported Collaborative Learning to Policy and Practice: CSCL2011 Conference Proceedings*, Vol. 1, 2011, 184–91.

[10] P. Jermann and M.-A. Nüssli, "Effects of Sharing Text Selections on Gaze Cross-Recurrence and Interaction Quality in a Pair Programming Task," in, *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work*, ACM, 2012, 1125–34.

[11] N. Kuriyama, A. Terai, M. Yasuhara, T. Tokunaga, K. Yamagishi, and T. Kusumi, "Gaze Matching of Referring Expressions in Collaborative Problem Solving," in, *International Workshop on Dual Eye Tracking in CSCW*, 2001.

[12] N. Marwan and J. Kurths, "Nonlinear Analysis of Bivariate Data with Cross Recurrence Plots," *Physics Letters A*, 302(5–6), 2002, 299–307.

[13] N. Marwan, M. C. Romano, M. Thiel, and J. Kurths, "Recurrence Plots for the Analysis of Complex Systems," *Physics Reports*, 438(5–6), 2007, 237–329.

[14] M. A. Nüssli, "Dual Eye-Tracking Methods for the Study of Remote Collaborative Problem Solving," PhD thesis: École Polytechnique Fédérale de Lausanne, *PhD thesis*, 2011.

[15] J. K. Olsen, M. Ringenberg, V. Aleven, and N. Rummel, "Dual Eye Tracking as a Tool to Assess Collaboration," *ISLG 2015 Fourth Workshop on Intelligent Support for Learning in Groups*, 25, 2015.

[16] S. Pietinen, R. Bednarik, T. Glotova, V. Tenhunen, and M. Tukiainen, "A Method to Study Visual Attention Aspects of Collaboration," in, *Proceedings of the 2008 Symposium on Eye Tracking Research & Applications – ETRA 08*, 2008, 39–42.

[17] D. C. Richardson and R. Dale, "Looking to Understand: the Coupling between Speakers and Listeners Eye Movements and Its Relationship to Discourse Comprehension," *Cognitive Science*, 29(6), 2005, 1045–60.

[18] M. Rittenbruch and G. McEwan, "An Historical Reflection of Awareness in Collaboration," 2009, 3–48.

[19] S. Schinkel, O. Dimigen, and N. Marwan, "Selection of Recurrence Threshold for Signal Detection," *The European Physical Journal Special Topics*, 164(1), 2008, 45–53.

[20] B. Schneider and R. Pea, "Real-Time Mutual Gaze Perception Enhances Collaborative Learning and collaboration Quality," *International Journal of Computer-Supported Collaborative Learning*, 8(4), 2013, 375–97.

[21] M. Schrage, "Shared Minds: The New Technologies of Collaboration," 1991.

[22] P. P. Shah and K. A. Jehn, "Do Friends Perform Better than Acquaintances? The Interaction of Friendship, Conflict, and Task," *Group Decision and Negotiation*, 2(2), 1993, 149–65.

[23] K. Sharma, P. Jermann, M.-A. Nüssli, and P. Dillenbourg, "Gaze Evidence for Different Activities in Program Understanding," in, *24th Annual conference of Psychology of Programming Interest Group*, 2012.

[24] M. Villamor and M. M. Rodrigo, "Characterizing Collaboration based on Prior Knowledge in a Pair Program Tracing and Debugging Eye-Tracking Experiment," in, *15th National Conference on Information Technology Education (NCITE 2017)*, 2017.

[25] M. Villamor and M. M. Rodrigo, "Do Friends Collaborate and Perform Better?: A Pair Program Tracing and Debugging Eye-Tracking Experiment," in, *18th Philippine Computing Science Congress*, 2018.

[26] M. Villamor and M. M. Rodrigo, "Exploring Lag Times in a Pair Tracing and Debugging Eye-Tracking Experiment," in, *25th International Conference on Computers in Education*, New Zealand, 2017.

[27] M. Villamor and M. M. Rodrigo, "Impact of Prior Knowledge and Acquaintanceship on Collaboration and Performance: A Pair Program Tracing and Debugging Eye-Tracking Experiment," in, *25th International Conference on Computers in Education*, New Zealand, 2017.

[28] M. Villamor and M. M. Rodrigo, "Predicting Successful Collaboration in a Pair Programming Eye Tracking Experiment," in, *26th Conference on User Modeling, Adaptation and Personalization (UMAP '18)*, Singapore, 2018.

[29] M. M. Villamor and M. M. T. Rodrigo, "Characterizing Individual Gaze Patterns of Pair Programming Participants," in, *Proceedings of the 26th International Conference on Computers in Education (ICCE 2018)*, 2018, 193–8.

[30] M. M. Villamor and M. M. T. Rodrigo, "Gaze Collaboration Patterns of Successful and Unsuccessful Programming Pairs Using Cross-Recurrence Quantification Analysis," *Research and Practice in Technology Enhanced Learning*, 14(1), 2019, 25.

[31] M. M. Villamor and M. M. T. Rodrigo, "Impact of Pair Programming Dynamics and Profiles to Pair Success," in, *Proceedings of the 26th International Conference on Computers in Education (ICCE 2018)*, 2018, 123–32.

[32] C. L. Webber Jr and J. P. Zbilut, "Recurrence Quantification Analysis of Nonlinear Dynamical Systems," *Tutorials in Contemporary Nonlinear Methods for the Behavioral Sciences*, 1998, 26–94.

[33] J. P. Zbilut, A. Giuliani, and C. L. Webber, "Detecting Deterministic Signals in Exceptionally Noisy Environments Using Cross-Recurrence Quantification," *Physics Letters A*, 246(1–2), 1998, 122–8.

[34] B. Zheng, N. Hajari, and M. S. Atkins, "Revealing Team Cognition from Dual Eye-Tracking in the Surgical Setting," in, *Ninth Biennial ACM Symposium on Eye Tracking Research and Applications*, 2016.