Affect Recognition in Code Review: An in-situ biometric study of reviewer's affect

Hana Vrzakova*

University of Eastern Finland, School of Computing, Joensuu, Finland

Andrew Begel

Microsoft Research, Redmond, Washington, 98052 U.S.A.

Lauri Mehtätalo, Roman Bednarik

University of Eastern Finland, School of Computing, Joensuu, Finland

Abstract

Code review in software development is an important practice that increases team productivity and improves product quality. Code review is also an example of computer-mediated asynchronous communication prone to the loss of affective information. Since positive affect has been linked to productivity and success in software development, prior research has focused on detection of sentiment in source codes. Although the methods of sentiment analysis have advanced, there are remaining challenges due to numerous domain oriented expressions, subtle nuances, and indications of sentiment. In this work, we explore the potentials of 1) nonverbal behavioral signals such as conventional typing characteristics, and 2) indirect physiology (eye gaze, GSR, touch pressure) that reflect genuine affective states in non-manipulated in-situ code review tasks in a large software company. Nonverbal behavioral signals of 33 professional software developers were unobtrusively recorded while they worked on their daily code review tasks. Using Linear Mixed Effect Models, we observed that affect presented in the written comments was associated with prolonged typing duration. Using features derived from eye gaze, GSR, and touch pressure, a trained Random Forest classifier could predict post-task valence with 90.0% accuracy (F1-score = 0.937) and arousal with 83.9% accuracy (F1-score = 0.856). The results presents potentials for intelligent affect-aware interfaces for code review in-situ.

Keywords: Code Review, Affective Computing, Physiological Signals, CSCW

1. Introduction

The ability to effectively communicate and evaluate affect and emotions is central to human daily activities, and is considered one of the key underlying skills for a functional team collaboration (Islam and Zibran, 2018; Schneider et al., 2018; Graziotin et al., 2018). In remote computer-mediated interaction and textual communication, however, social-behavioral signals, especially nonverbal behaviors, are muted (Hancock et al., 2007; Schulze and Krumm, 2017), and, consequently, affective information becomes less salient or even gets lost.

In these indirect contexts, interpretation of affective states is tremendously hard (Riordan and Trichtinger, 2017) since textual representations contain less affect than for example a phone call (Picard, 1999). Consequently, the tone of the message can quickly, unnoticeably, yet significantly change affective polarity; humor may be interpreted as offense, a critique sounds inadequate and harsh, serious message may get ignored, and a constructive note suddenly may look like a joke. In software development business, affect processing, understanding, and communication have been of high importance and central to successful development process. Understandably, experiencing positive affect has been beneficial for performance and productivity (Wrobel, 2013; Müller and Fritz, 2015; Schneider et al., 2018; Graziotin et al., 2018).

Affect loss becomes imminent in large scale distributed teams. Moving from the activity of a programmer working in isolation to distributed collaborative networks of developers, the number of spatially remote teams is increasing (Herbsleb and Mockus, 2003). Consequently, software business is responding to the trend with utilization of collaborative virtual environments and tools (Storey et al., 2017). These tools, however, lack the abilities to support effective affect signalling and recogni-

Email addresses: hana.vrzakova@uef.fi (Hana Vrzakova*), andrew.begel@microsoft.com (Andrew Begel), lauri.mehtatalo@uef.fi (Lauri Mehtätalo), roman.bednarik@uef.fi (Roman Bednarik)

tion. In this work, we focus on developing methods and tools that can enhance the communication channels by automatic recognition of code reviewer's affective state.

Source code review is an example of a software development activity that evolved from a human-to-human interaction at arranged meetings to the asynchronous and remote computer-mediated textual communication (Bacchelli and Bird, 2013). Although saving time and speeding up product release cycles, the contemporary form is arguably prone to affective loss. To extract the sentiment and emotions in written text, prior research has employed numerous machine-learning tools for sentiment detection (for review, see e.g. Tang et al. (2009); Mäntylä et al. (2018)). In source-code related texts, however, aspects such as context-sensitive variations of words, subtle expressions of sentiment, irony, or sarcasm, still hinder effective recognition of affect (Islam and Zibran, 2018).

In this work, we build and evaluate a multimodal recognition of affect from programmers' nonverbal signals during the code review tasks. While current research on affect recognition has employed directly observable modalities to recognize basic and elicited emotions (predominantly from facial and acoustic-prosodic expressions, for review see e.g. D'mello and Kory (2015)), the context of code review is less suited for such a approaches, due to lacking speech activity and privacy considerations.

Our work differentiates from- and advances the prior research in numerous aspects and present following contributions:

- With minimal interference to the professional source-code review practice, we collected behavioral and physiological signals, namely typing activity, eye gaze, galvanic skin response, and touch pressure, during the in-situ code review in one of the largest software companies.
- In a variety of code review tasks, we model and analyze *unelicited* affective states, a goal towards automatic detection of reviewer's affect. We investigate two methods of affect detection.
- In the analysis of affect in individual comments, we analyze conventional behavioral metrics (typing duration and comment length) for each individual participant, using Linear Mixed Effect models since such approach does not require any dedicated sensor.
- In the second analysis of overall participant's affect, we extracted features related to physiological states and employed a machine learning based framework to distinguish valence and arousal polarity.
- Effective communication of affect remains unexplored in the software engineering domain. In this work, we weight on the benefits of both approaches and potentials for future *affect-enhanced* code review.

We center our research on following research questions: [

1. *How do organizational, task related, and longterm emotional aspects predict the components of affect after the code-review task?*
2. *How does presence of emotion impact the conventional office behavior, i.e. typing on a keyboard?*
3. *How do nonverbal physiological signals predict components of affect after the code-review task?*

]

The rest of paper is organized as follows. Section 2 provides introduction to the domain of code review, overviews the studies of physiological signals in software engineering. In Section 3, we present experimental settings of the in-situ study; we propose two analysis methods in Section 4. Section 5 summarizes results of the analyses using conventional behavioral measures and results of physiological signal analysis using machine learning. In Section 6, we discuss the results in the light of current research, limitations, and future directions of affect recognition in code review.

2. Background

The task of affect recognition in code review spans the fields of software engineering, affective computing, computer supported collaborative work, and inference of user states from physiological signals. We will provide a brief introduction to a range of corresponding studies with respect to software engineering. First of its kind, we report on a multimodal investigation of affect in code review that was performed in situ; we recorded the data in one of the largest software companies and our participants were professional code reviewers engaged in their everyday tasks.

2.1. Origins of code review

The contemporary code-review has originated from *code inspection* in which source codes were scrutinized at formal project group assemblies that could lasted days (Fagan, 1999). Due to fast pace of software production and understandable impracticality of the long face-to-face meetings, code inspection became a computer mediated task through dedicated interfaces and tools, such as CodeFlow (see Figure 1). In principle, code review still resembles a code inspection session where a developer evaluates others' code, seeks the potential errors, and suggests improvements; however, the current form has evolved to informal, lightweight, and brief code review practice.

The evolution of code review practice came along with the development of tools dedicated for code review.

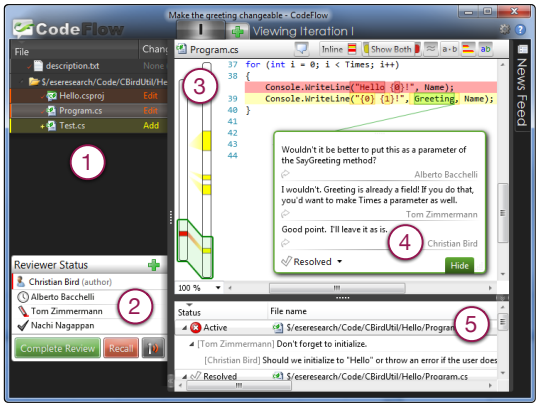

Figure 1: A typical user interface of a corporate code review tool (CodeFlow). A source code is selected in a source code hierarchy (1), displayed in a main window (3), and commented in pop-up windows (4). A review summary and comments of others is displayed in bellow the code (5). Adopted from Bacchelli and Bird (2013).

To an external observer, current user interfaces are unrecognizable from fully functional IDEs with access to linked libraries, classes, and relevant resources. Reviewing a code is no different to commenting a shared document where comments are directly linked to a relevant piece of code. In addition, all the comments are synchronized and shared across reviewers who were invited to a particular code review.

The code reviewing comprises various strategies (Peng et al., 2016; Uwano et al., 2006). Some reviewers briefly proof-read the code, check on the code style, and search for obvious typo-like errors, while others seek for logical errors, and provide mentoring in their comments (Bacchelli and Bird, 2013; Ebert et al., 2018). Similar to revisions in writing, code reviews are repaired in iterations, which also direct reviewer's strategy of proof-reading. While the first rounds of code review may require focused reading, later reviews of the same code may be concise, scrutinizing whether the reviewer's recommendations were implemented.

Independently on the selected strategy, reviewer's reasoning, decision making, and affective states remain hidden to the author of the code unless explicitly written in the comments. Furthermore, the comments are required to be brief, factual, and free of certain emotions; often, a specific professional code of conduct dictates the content and format of the comments (Lutchyn et al., 2015). Therefore, in code review, affective information gets lost due to the form of communication, computer mediation, and professional conduct. Understanding affect in such settings is crucial for efficient functioning in the teams (De Choudhury and Counts, 2013; Dewan, 2015), contributing to effective communication and coordination (Herbsleb et al., 1995; Schneider et al., 2018).

2.2. *Affect in software engineering*

Understanding of developer's affect and emotion awareness in teams, in particular, underlie effective software engineering (Dewan, 2015). To general audiences, however, the domain of software engineering seems exempt from extreme and overt affective expressions and software developers may appear calm, focused, or distant. On the contrary, since software development is highly dependent on developer's cognitive efforts, his performance, productivity, and creativity is influenced by affect (Islam and Zibran, 2018; Schneider et al., 2018). Happy programmers are often more productive (Graziotin et al., 2013, 2018), while negative emotions are detrimental for the software development (Müller and Fritz, 2015; Gachechiladze et al., 2017).

Affect assessment during software development is unsurprisingly challenging (Schmidt et al., 2018). One explanation could be that when one is engaged in already cognitively demanding task, as code review, reflecting on one's cognitive processing and states becomes tremendously difficulty and elevates participant's overall high workload (Ericsson and Simon, 1993). Therefore, traditional qualitative methods of affect assessment fall short and have been observed as costly, time demanding, and challenging for adoption in software development (Schmidt et al., 2018; Lutchyn et al., 2015).

Prior analyses of developer's internal states, as happiness (Graziotin et al., 2013, 2018), frustration (Hernandez et al., 2014; Müller and Fritz, 2015), anger (Gachechiladze et al., 2017), stress (Sano et al., 2017), and workload (Fritz et al., 2014), have laid foundations to the development of novel inferential methods such as sentiment analysis from source code resources (Islam and Zibran, 2018) and non-verbal sensing from direct and indirect physiology (D'mello and Kory, 2015; Shu et al., 2018).

[Recently, several studies have tackled the code review and sentiment analysis from the text. Using Gradient Boosting Tree, SentiCR allows to recognize ... (Ahmed et al., 2017)... Sentiment in code review was analyzed with respect to reviewer's gender (Paul et al., 2019)... (Novielli et al., 2018) and (Calefato et al., 2018)...]

To estimate increasing workload during the SE task, Fritz et al. (2014) have employed galvanic skin response (GSR) and electroencephalography (EEG) together with eye-tracking sensors. In the analysis of multimodal signals, programmer's perceived difficulty with the code has been predicted with 84% precision with every new task. To measure frustration during daily work, Hernandez et al. (2014) have averaged signals from a pressure sensitive keyboard and a capacitive mouse (TouchMouse). Under stress, both typing pressure and contact with the mouse have increased in 75% of participants. Similarly, in the context of software change tasks, Müller and Fritz (2015) measured developer's valence and feeling of progress using GSR, EEG, and heart rate. In the analysis of valence, a classifier predicted developer's emotional reaction with 71.36% accuracy and the feeling with progress with 67.70% accuracy. Signals of indirect physiology and user's activity have also been employed in recognition of stress and suitable timing of

stress micro-interventions. Sano et al. (2017) employed an array of sensors, sensing developer's activity, hear rate variability (HRV), and intervention history, to predict the timing when a preventive intervention should be delivered to be both efficient and unobtrusive. Multi-kernel SVM could differentiate the suitable and unsuitable timing with 80% accuracy.

A major drawback of the previous typically high-controlled lab-based studies comes from the characteristics of the source code presented. The codes exposed during the experiments have been usually isolated, shortened or simplified to fit the screen. In addition, the materials were not connected to the routine work of the participants, and unrelated to their projects and responsibilities. In addition, the participants had little or no means of any interaction with the code, such as scrolling the code, opening other source codes, switching to necessary libraries, and searching for supporting source codes. In a real world scenario, however, a single source code can easily cover hundreds of lines, is a part of a larger package or a project, and is created and maintained by numerous programmers in the team.

In this work, we investigate covert behavioral signals that have not been extensively studied before, but they can, however, be embedded into the daily work environments. We utilize three sensors of measuring indirect physiology with great potentials for ubiquitous sensing computers, and we link the biometric signals to reviewer's self-assessed affective states.

3. Experiment in-situ

In the study, we observed professional software developers conducting code reviews as part of their daily work at a large international software company. We instrumented participants with three wearable sensors: a Shimmer GSR to measure electrodermal activity (EDA), a 195-point Microsoft capacitive TouchMouse to measure stress levels, and a portable remote Tobii eye tracker to identify reviewer's focus in the code. We chose affordable and mobile sensors that are simple to embed in near-future computers and that do not require lengthy instrumentation of users.

3.1. Task and procedure

The experiment was designed as an in-situ study and held in each participant's office. We brought all the experimental equipment for each session and embedded it into each participant's environment. After the participant became familiar with the task of the experiment and signed a consent form, he answered a set of preparatory questionnaires related to his experience with the code review process in the company and his long-term affect. To that end, we employed the PANAS measurement tool (Watson et al., 1988).

While the participant was answering the questionnaires, we installed the sensors to the experimental computer, plugged it to the participant's primary monitor, and mirrored the participant's work account. The account mirroring and minimal changes in participant's environment ensured high levels of experiment ecological validity. Finally, we calibrated the eye-tracker using a 9-point calibration and validated the calibration by asking the participant to read aloud the first and the last visible line of the source code at the screen.

After the calibration, the participant was free to proceed to the code review of his choice and to work as long as needed to complete the task. After each task, the participant was asked to assess his affective status towards the code, the author of the code, and participant's self (using the PAM scale (Pollak et al., 2011)). We were also interested how much the participant was familiar with the code, how hard the review task was (using the NASA TLX (Hart and Staveland, 1988)), and about his work hierarchy position towards the programmer of the code.

At the end of the experiment session, the reviewer was reimbursed with $8 cafeteria coupons for participation in the study. Each session lasted about an hour, with the first 15 minutes dedicated to the setup and the calibration.

3.2. Participant recruitment

We recruited 37 software developers (2 female, 35 male) in one of the largest U.S. software companies, Microsoft. The age of the participants ranged from 25 to 43 years (mean = 34 years, SD = 4.74). Each participant was the member of a team responsible for building and shipping consumer-focused software products. Potential participants were identified through a focused search of the company-wide code review database. As all participants were newly assigned to review on the daily basis, we repeated our focused search and emailed potential candidates whether they would be willing to participate in the experiment.

3.3. Tools and apparatus

Preserving the in-situ nature of the study was important aspect for mimicking future interaction with the ubiquitous code review environment. We instrumented participants with three portable biometric sensors: Tobii EyeX (60Hz, binocularly), Shimmer3 GSR+ electrodermal activity (EDA) sensor (Shimmer, 2016) and 195-point Microsoft TouchMouse Ultimate (Corporation, 2016).

All sensors were integrated into the code review environment using a corresponding API and synchronized with the reviewer's interaction over Bluetooth. In addition to the physiological signals, we recorded the mouse position in the CodeFlow window. Preprocessing of the recorded data streams were performed using custom made Python scripts with included libraries Scikit-learn (Pedregosa et al., 2011), Numpy (van der Walt et al., 2011), and Pandas(McKinney, 2010). Inferential analysis was

performed using *lme4* and *nlme* libraries (Bates et al., 2015; Pinheiro et al., 2017) available in R (R Core Team, 2015). Training and evaluation of machine learning models were conducted at Taito supercluster [1].

4. Analysis of reviewer's affect in comments

In this work, we evaluate how conventional metrics related to code review reveal affect occurrence in a review comment, and how nonverbal behavioral signals respond to long-term affect during the code review task. First, the conventional metrics were fitted to a Linear Mixed Effect model (LME) to see whether affect influenced the typing speed of the reviewer. Second, reviewer's physiology is encoded to physiological features and employed in training of Random Forest, predicting reviewer's positive and negative valence after the task.

4.1. Affect in code review comments

All comments were first gathered from a CodeFlow database and annotated according to affect occurrence as *neutral* (negative class) or *emotional* (positive class). Annotators were instructed to search for emotional words, expressions of feelings, and emojis, that occurred in the comments. During the annotation process, two independent annotators processed the labels independently on each other, and then, corrected the labels after the joint discussion (inter-rater agreement = 66.92%). Altogether, reviewers produced 259 comments, 238 neutral comments (92.31%) and 21 emotional comments (7.69%). Comments with affect were produced by 10 reviewers (out of 33) who also wrote other neutral comments.

To investigate, whether the affect in the comments can be captured by tools already available, we extracted metrics related to comments. Using database query, we exported the timestamps of comment opening and closing and the content of the comment. Each comment was represented by its *typing duration* (seconds) and *comment length* (number of characters in the comment). To reveal how *typing duration* was influenced by comment *emotionality* and *comment length*, a Linear Mixed Effect model was fitted to model the the annotated dataset.

As fixed effects, we entered *comment emotionality*, *comment length* and their interaction into the model; as random effects, we had intercepts and random slopes for each *participant*. Nested random effects for *task* within participant were also tried but omitted since it explained negligible amount of variance. Finally, we relaxed the assumption of constant residual variance, since the residuals inclined to increase as a function of fitted value in the residual plots. Significance testing was obtained using Wald's F-tests of the full model. The resulting model is defined as

$$y_{ij} = \beta_0 + \beta_1 x_{ij}^{(1)} + \beta_2 x_{ij}^{(2)} + \beta_3 x_{ij}^{(1)} x_{ij}^{(2)} + b_i^{(1)} + b_i^{(2)} x_{ij}^{(2)} + \varepsilon_{ij}\,, \tag{1}$$

where y_{ij} is the typing duration for comment j of participant i, $x_{ij}^{(1)}$ is the emotionality and $x_{ij}^{(2)}$ is the centralized comment length (ranging from 3 to 334 characters per a comment), $(b_i^{(1)}, b_i^{(2)})'$ are the random effects for participant i, independent among participants and having bivariate normal distribution with mean zero and unknown variance, and ε_{ij} are independent normally distributed zero-mean residuals with variance

$$var(\varepsilon_{ij}) = \sigma^2 \left|\widehat{y}_{ij}\right|^{2\delta}\,. \tag{2}$$

During model development, visual inspection of residual plots and Q-Q plots revealed two outliers in the neutral class. In these two cases, the participants opened a comment and a web-browser, and spent over four and seven minutes simultaneously typing and searching for additional information online. Since the comments did not present a typical behavior, we removed the comments as a outlier.

4.2. Data preprocessing and feature engineering

Affordable biometric sensors often output a noisy signal, unsuitable for direct statistical inference. To filter and clean the input data, we performed several transformations. Since data transfer during the experiment was established over Bluetooth, all data streams were recorded with best effort sampling frequency and fluctuated in time. The frequency was unified to 50Hz using a mean of values and a backward propagation of missing values. If a data point was missing in the re-sampled data frame (a gap between data points was bigger than 20ms in the original data frame), the last data sample from the previous data frame was propagated. The re-sampling routine reestablished signal continuity and filtered out the missing values.

Each data stream was filtered to remove noise. Raw GSR data was first normalized with Z-score and smoothed with exponential filter ($\alpha = 0.08$). A decomposition of electrodermal activity followed the routine introduced by Fritz et al. (2014) and split the signal to *phasic component* (skin conductance response, SCR), which is associated with fast events as a shock or surprise, and *tonic component* (skin conductance level, SCL), which responds to slow changes in autonomic arousal (Braithwaite et al., 2013). SCL was extracted using low-pass Butterworths filter (0.05 Hz, 5^{th} order), revealing the slow trends in participant's arousal, while SCR was obtained with the high-pass filter (0.33 Hz, 5^{th} order), capturing spikes in arousal.

Raw eye-tracking data were filtered out in real-time during experiment using a median filter with 10s sliding window to reduce amount of missing data. To characterize attentional behavior and gaze shifts during the code

[1] https://research.csc.fi/taito-supercluster

review, we employed the measures of *Euclidean distance* and *velocity* derived from consecutive gaze samples. In addition, each raw data point was mapped to a line in the source code, expressed with the absolute line number. This data source, however, was omitted from the analysis due to a large portion of missing data, which was a result of eye-tracker miscalibration.

Raw TouchMouse data were recorded in form of a 2D grid, representing the surface of the mouse and the capacitance of the touch. The 2D information was processed into two components - a sum of the capacitive pixels (TouchMouseSum) and a number of fingers detected from the grid (TouchMouseCount) (Hernandez et al., 2014).

Pre-processed data series from each sensor were sliced with two second time window with no overlap. To characterize fluctuations within the observed 5 minutes prior to the end of task, a battery of statistical features (*mean*,*median*, *variance*, *minimum*, *maximum*, *sum*) were computed for each 2-second data slice (see Table 1). The final feature set contained 55 features (12 from GSR (phasic and tonic component: 2x6), 33 from eye gaze (eye-gaze distance: 3x6; eye-gaze velocity: 3x5), and 10 from TouchMouse (TouchMouse SUM and COUNT: 2x5).

Table 1: Features computed from indirect physiology.

Modality	Measure	Feature
GSR	Tonic component	Mean
		Median
		Variance
	Phasic component	Maximum
		Minimum
		Sum per s
Gaze	Euclidean distance	Mean
	Horizonal Euclidean distance	Median
	Vertical Euclidean distance	Variance
	Velocity	Maximum
	Horizontal velocity	Minimum
	Vertical velocity	Sum
TouchMouse	Sum of capacitive pixels	Mean
		Median
		Variance
	Number of fingers detected	Maximum
		Minimum
		Sum

4.3. Machine learning

For overall affect recognition, we investigated how features derived from the physiological signals predict the reviewer's affect after the code review. Target labels were retrieved from the PAM questionnaires, where each cell in the grid corresponds to the level of valence and arousal (1-4) (Pollak et al., 2011). Figure 3 illustrates the distribution of valence and arousal ratings after the review. The range of PAM values were binarized to +1 (positive valence) and -1 (negative valence).

Because affect has been known to develop slowly in time (Ekman and Davidson, 1994), to predict the outcomes after the task, we explored last five minutes before the end of the task. Code reviews shorter than five minutes were omitted from the analysis. The final dataset consisted of 3900 feature vectors with 55 features.

Recognition performance of physiological features was evaluated using a Random Forest classifier. Classifier parameters were first optimized using a random grid search with Area Under the ROC Curve (AUC) as the optimization criterion. Next, the classifier was validated with selected parameters in 5x5 crossvalidation. In each fold, the feature set was randomly shuffled and split with stratified sampling to sustain the original class imbalance. Class distribution in training folds were balanced using the SMOTE approach (Chawla et al., 2002); class distribution in testing folds remained imbalanced. In this work, we report on average accuracy, F1-score, true positive and true negative rates averaged over the testing sets.

5. Results

We report on three primary findings. First, we evaluate what was participants' affect after the code review task and how other external factors potentially contributed to the resulting affective state. Next, we report on affect presented in the written code review comments and perform a regression of the affect presence with comment typing characteristics. Lastly, we discuss a recognition performance of valence- and arousal-based classification that was trained using physiological features.

5.1. Self-assessed affect after the code review task

After each task, participants assessed their current affective state using the Photographic Affect Meter (PAM scale) (Pollak et al., 2011) illustrated in Figure 2. As seen in Figure 3, participants' affect after the task was skewed towards positive valence (on the x-axis) and was fairly balanced in terms of arousal (on the y-axis).

We hypothesized that numerous effects could impact reviewer's affective state. We expected that covariates such as the long-term affect prior to the experiment (measured as sum of positive and negative affect scores in PANAS), familiarity with the code, seniority of the programmer, and task duration and difficulty (measured as a sum of NASA TLX scores) could impact reviewer's affect. We examined these as independent variables using linear regression.

After controlling for covariates (see Table 2), only positive long-term affect prior to the experiment (PANAS) predicted participant's valence after the task (B=0.062, p=0.028). Reviewer's long-term negative affect, task duration and workload, familiarity with the code, nor seniority to the code author did not statistically explain reviewer's affect after the task.

Figure 2: An example of Photographic Affect Meter (PAM scale).

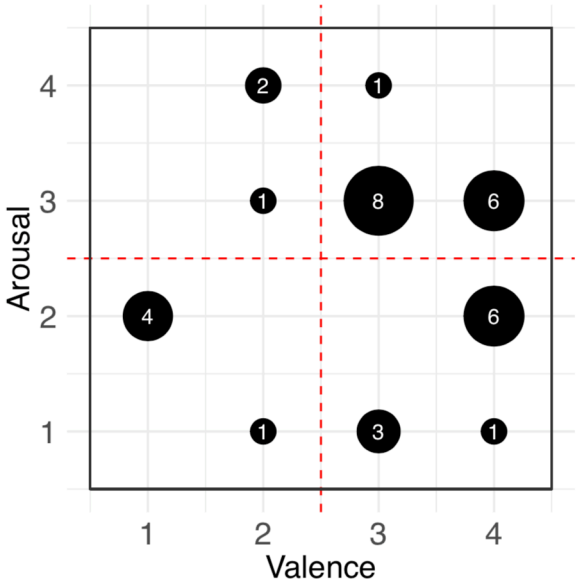

Figure 3: Distribution of reviewers' valence (horizontal) and arousal (vertical) after the task. The location of the points presents four affective quadrants of the PAM scale. The radius of the points corresponds to the number of participants who reported a particular state.

5.2. *Predicting affect from comment characteristics*

Next, we explored whether the emotionality in the comment reflects conventional metrics such as typing duration and typing speed. From the distributions in Figure 4, it is apparent that comment emotionality (white) was related to the increased mean typing duration and variance.

When evaluating predictive power of conventional metrics, the fitted LME model revealed that the typing duration in an average-length comment was 23.21 seconds emotionality increased it by 12.07s (Std.error = 4.00) significantly ($F_{1,218}$ =13.49, p-value=0.003**). The average effect of comment length on the duration was 0.24 seconds/character, and emotionality increased it significantly by 0.17 seconds/character (Std.error 0.06, p-value=0.007**) to 0.41 seconds/character. Table 3 reports on the estimated parameters in detail. What stands out in this table is the comment length did influence the typing duration, however, less than participant's affect.

Table 2: Factors contributing to affect reported after the task. Logistic regressions revealed that only long-term positive affect (assessed by PANAS) were significantly associated with valence after the task.

	Dependent variable:	
	Valence	Arousal
Longterm positive affect	0.062** (0.027)	0.018 (0.025)
Longterm negative affect	−0.017 (0.037)	−0.024 (0.035)
Task workload	−0.004 (0.013)	0.003 (0.012)
Task duration	0.013 (0.017)	−0.003 (0.017)
Code familiarity	−0.847 (0.563)	0.232 (0.537)
Reviewer's seniority	0.073 (0.235)	0.130 (0.225)
Constant	1.719 (1.311)	1.687 (1.251)
Note: n=33		*p<0.1; **p<0.05; ***p<0.01

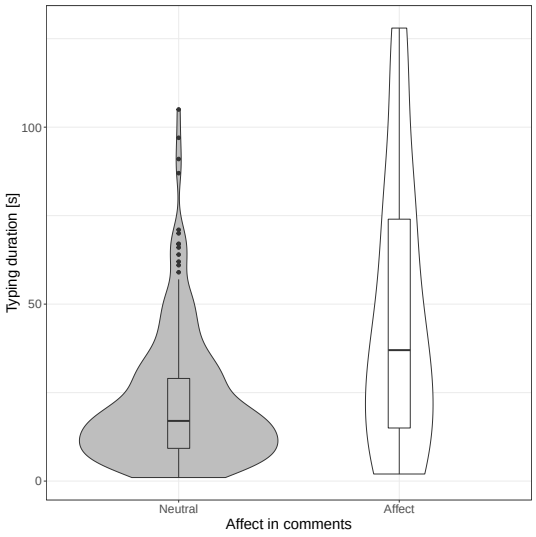

Figure 4: Typing duration with respect to the comment emotionality. Comments with a neutral undertone were produced in shorter time, while comments with emotionality required longer time and presented higher variability.

5.3. *Predicting affect components from indirect physiology*

The predictive power of our multimodal features was evaluated for both affect components separately (valence and arousal) using a Random Forest classifier and a 5x5 kFold shuffled crossvalidation. Table 4 summarizes the performance of the model achieved on crossvalidation test sets for all features (modality fusion) and features extracted from individual sensors. Baseline performance was obtained on the full feature set using a default dummy classifier.

Table 3: Parameter estimates of the Linear mixed effect model for comment duration in equation (1).

Fixed part			
	Estimate	Std. Error	p-value
β_0: Intercept	23.21	1.34	0.000***
β_1: Emotion	12.07	4.00	0.003**
β_2: Comment length	0.24	0.01	0.000***
β_3: Emotion: Comment length	0.17	0.06	0.0069
Random part and residual			
$var(b_i^{(1)})$	5.70^2		
$var(b_i^{(2)})$	0.0306^2		
$cor(b_i^{(1)}, b_i^{(2)})$	0.998		
δ	0.989		
σ^2	0.498^2		

Table 4: Classification of valence and arousal at the end of the task using last five minutes of data. The best performance was achieved when including all features (modality fusion).

	Measure	Baseline	GSR	Eye gaze	Touch pressure	Modality fusion
	ACC	0.490	0.686	0.858	0.634	0.900
	F1	0.599	0.789	0.912	0.719	0.937
Valence	AUC	0.493	0.670	0.870	0.761	0.937
	TPR	0.488	0.754	0.946	0.605	0.957
	TNR	0.499	0.446	0.547	0.734	0.695
	ACC	0.517	0.629	0.766	0.697	0.839
	F1	0.540	0.664	0.785	0.753	0.856
Arousal	AUC	0.518	0.682	0.860	0.771	0.922
	TPR	0.507	0.654	0.763	0.826	0.853
	TNR	0.529	0.597	0.771	0.532	0.823

In recognition of valence and arousal after the task, the best performance was achieved using the fusion of the modalities. The overall model of valence performed higher ($accuracy$ = 90.0%, $F1score$=0.937) compared to the model of arousal ($accuracy$ = 83.9%, $F1score$ = 0.856); the model of valence predicted better the positive valence labels (TPR = 0.957) than the negative valence (TNR = 0.695), suggesting that the directionality of valence is somewhat reflected in the signals. The model of arousal predicted the high arousal (TPR = 0.853) higher than the negative arousal (TNR = 0.823); however, the differences were minor, suggesting that the polarity of arousal was well distinguishable from the employed physiological signals.

Considering the individual modalities, eye gaze alone performed better than other modalities both in recognition of valence ($accuracy$ = 85.8%, $F1score$ = 0.912) and arousal ($accuracy$ = 76.6%, $F1score$ = 0.785). While scoring higher in favor of valence, however, the gaze-based classifier delivered a balanced performance in favor of arousal. Of all modality combinations, touch pressure predicted better negative valence (TNR = 0.734) compared to positive valence (TPR = 0.605).

Table 5: Classification of valence and arousal at the beginning of the task using first five minutes of data.

	Measure	Baseline	GSR	Eye gaze	Touch pressure	Modality fusion
	ACC	0.488	0.635	0.822	0.678	0.855
	F1	0.597	0.746	0.890	0.770	0.910
Valence	AUC	0.490	0.596	0.826	0.747	0.870
	TPR	0.486	0.688	0.927	0.692	0.950
	TNR	0.493	0.448	0.450	0.632	0.518
	ACC	0.515	0.556	0.723	0.684	0.798
	F1	0.539	0.612	0.750	0.701	0.822
Arousal	AUC	0.517	0.562	0.806	0.773	0.884
	TPR	0.505	0.626	0.740	0.665	0.831
	TNR	0.528	0.467	0.702	0.707	0.756

5.4. Predicting affect in time

It is a reasonable assumption that affect builds up during the task, given the fact that important events occur during interaction with the code. Therefore, we hypothesized, that recognition of the target labels should be more difficult earlier in the data sets. In other words, should such system be implemented in real-life, it is important to understand, whether early recognition based on historical data performs as well as recognition based on more recent inputs. To test this hypothesis, we extracted the same feature sets from the beginning of the task, set the same labels as measured after the task, and repeated the analyses.

As illustrated in Table 5, recognition results on past data were approximately 4% lower in both fusion models compared to the models based on recent data from the end of the task. The largest differences were observed in the galvanic skin response both in valence($\Delta accuracy$ = 5.12%, $\Delta F1score$ = 0.043) and arousal ($\Delta accuracy$ = 7.27%, $\Delta F1score$ = 0.051).

6. Discussion

Affect in collaborative tasks in general, and in code review in particular, have important consequences for team performance; indeed, happy programmers have been proven to be more productive (Graziotin et al., 2018). Negative affect, on the other hand, may create an obstacle towards professional conduct, or performance of a task (Gachechiladze et al., 2017). With the prevalence of computer mediated communication during code review, textual comments do not effectively transmit the subtle but significant social and behavioral cues, necessary for correct affect recognition.

Currently the best performance in affect recognition from nonverbal signals is obtained from facial and

prosodic expressions (D'mello and Kory, 2015). Being directly observed, recorded, and analyzed, however, these are unsuited in long-term and daily applications in software development teams. Conventional behavioral signals and indirect physiological expressions, as galvanic skin response, eye gaze, or touch pressure, present more suitable candidates for affect recognition, since they do not require participant's effort, cannot be easily controlled by the users, cannot be directly interpreted by an external observer and thus do not violate the sense of participant's privacy.

The first question of this work was whether reviewers' comments contain any recognizable affect at all. When two independent raters manually annotated the comment base, comments with affect represented a minority of the comment base (bellow 10%) which was expected and in line with prior research. As noted by Lutchyn et al. (2015), corporate rules of professional conduct inhibit certain emotions as inappropriate or undesirable in the workplace and workers are expected to manage their affective expressions. Although overt expressions can be voluntarily suppressed, involuntary behaviors, such as typing, still may communicate affect. As reported in this work, results suggest that comments with affect required significantly more time, independently on the comment length or task order.

The second question considered the extend to which indirect physiological signals correspond to genuine affective states in the in situ code review tasks. In recognition of valence and arousal, fusion of three modalities delivered performance high above baseline, more in favor of recognition of participant's valence. While recognition of high and low arousal were fairly balanced, recognition of positive and negative valence was skewed towards the positive class. One explanation could be that genuine expressions of valence in real-world tasks are more subtle compared to acted and cued expressions in the lab studies (Elfenbein and Ambady, 2002), and, thus certain aspects of affect are harder to detect.

When comparing performance of individual modalities, eye gaze delivered the highest recognition performance overall. Touch pressure delivered equal recognition for high and low arousal in the recognition of arousal, which corresponds with findings of Hernandez et al. (2014). Interestingly, models utilizing galvanic skin response scored the lowest out of the three sensors.

Several factors could contribute to these results. Readings of GSR are influenced by factors such as environmental temperature, physical activity, and individual differences in physiology (Braithwaite et al., 2013). In our study, the well air-conditioned offices, seated and focused work present realistic conditions that are challenging for GSR sensors. Also, code review tasks do not elicit extreme changes in affective states, therefore, we can conclude that in this case, GSR was more insensitive to subtle changes in valence and arousal than the other sensors. Another explanation proposed by Shu et al. (2018) is that while GSR can inform on the increase of the affect, it cannot differentiate the affect polarity.

6.1. Future work

The leading challenge in computer mediated asynchronous communication arises when affective information is undetected or misinterpreted by the other party (Ebert et al., 2019). The results presented here lay foundations for numerous investigations in real-life professional software development, as they in detail report on the feasibility of three suitable modalities for affect recognition. With respect to future research, we investigated how indirect physiology is linked to affect in code review with the aims of developing intelligent affect-aware code review.

In this study we focused on modeling the affective states of the reviewer (sender), and set grounds for models to identify how author developers (receivers of the review) perceive the comments with respect to the underlying affective tone. Also, next modeling approaches can extend our findings to other factors occurring in professional software development, such as confusion, misinterpretation, to the role of culture (Elfenbein and Ambady, 2002), and their relationship to the productivity of remote teams.

We envision a novel form of implicit affect-sensing system for code review that continuously improves awareness of changing levels of affect with respect to the reviewed code. Such sensing can be embedded into the development and review environments. The methods and the system present a needed tool to further the understanding of the link between emotions and work in software development teams.

Future research will focus on the questions how to meaningfully communicate affective states (Picard, 1997; Barral et al., 2016), but also on evaluations of how the affect-enhanced code review improve the communication and confusion among remote software development teams. In the daily work, intelligent multimodal affect recognition applied on comments could enable reviewers and developers to better communicate the meaning of the comment and assist in setting of the importance of written messages.

6.2. Threats to validity

As in any in-situ study, our work is not exempt from limitations. In this work, we purposefully employed affordable sensors and situated the study in the daily code review with low to none experimental control, which inherently introduced several limitations.

The reported results on affect in comments were limited by the sample size and class imbalance. Further work using even a larger set of annotated comments would be required to validate the results. Obtaining manually annotated database of code comments, preferably project and language specific, however, presents one of the current challenges in the sentiment analysis research (Islam

and Zibran, 2018; Basile et al., 2018). Although beneficial, obtaining such a database would require considerable resources of multiple project knowledgeable raters.

The study also contains a trade-off between the data quality, affordability of the sensors, and optimal data collection conditions. Since we visited each participant's on site, we could not ensure even illumination of the offices, nor the optimal distance from the eye-tracker, as recommended for eye-tracking experiments (Holmqvist et al., 2011). Similarly, we did not increase the temperature in the office nor enforce recommended physical exercise prior to the experiment to increase the accuracy of the GSR sensor (Braithwaite et al., 2013). The in-situ characteristic of the experiment took the toll in form of missing data in the recordings. Due to unrestricted settings and nature of the sensors, we expected challenges with evaluations and compensated them in form of careful and robust data processing, filtering, and selection.

7. Conclusion

In code review, the reviewer argues internally about the validity of the code: why the particular piece of code was written in the particular way and fitted into the particular position in the current project hierarchy, whether it is suited to the project best practices, or whether it does not violate software efficiency, to name a few. Reviewer's internal states related to code review, however, remain hidden to the author of the code and rarely propagate to the reviewer's feedback, as we observed in the current study.

In corporate software development, code review is a beneficial practice to improve code quality, share best practices among colleagues, and lower resources needed in product testing. However, when the code is reviewed in computer-mediated way, the reviews are lacking important social-cognitive cues that are crucial for efficient team functioning.

In this work, we investigated potentials of unobtrusive affect sensing using biometric sensors for purposes of enhancing code review. We ground our investigation using Linear Mixed Effect models and machine learning to capture affect during source code reviews in a real-life, in-situ data collection. With minimal interference to the professional source-code review practice, we collected physiological signals related to affective states and perform modeling and analysis towards automatic detection of reviewer's affect.

Authentic affect in the written reviews was significantly associated with increased typing duration of the comment. Genuine affect after the task was recognizable from employed biometric sensors that were installed on site.

Intelligent multimodal affect recognition in code review opens up to new research directions and applications. The next generation of code review tools can utilize affect recognition to better communicate detected affect in the code review. Future research on computer-mediated team collaboration could extend the present study and investigate the affective information received by author developers, the discrepancy between the reviewer's genuine affect and developer's perceived affect from the written reviews, and the extend to which intelligent affect-aware embedded in the code review can remedy understanding and communication challenges.

Acknowledgment

The work was supported by a Microsoft internship and by the Academy of Finland grant No. 305199.

References

Ahmed, T., Bosu, A., Iqbal, A., Rahimi, S., 2017. Senticr: a customized sentiment analysis tool for code review interactions, in: Proceedings of the 32nd ieee/acm international conference on automated software engineering, IEEE Press. pp. 106–111.

Bacchelli, A., Bird, C., 2013. Expectations, outcomes, and challenges of modern code review, in: Proceedings of the 2013 International Conference on Software Engineering, IEEE Press, Piscataway, NJ, USA. pp. 712–721.

Barral, O., Kosunen, I., Ruotsalo, T., Spapé, M.M., Eugster, M.J., Ravaja, N., Kaski, S., Jacucci, G., 2016. Extracting relevance and affect information from physiological text annotation. User Modeling and User-Adapted Interaction 26, 493–520.

Basile, V., Novielli, N., Croce, D., Barbieri, F., Nissim, M., Patti, V., 2018. Sentiment polarity classification at evalita: Lessons learned and open challenges. IEEE Transactions on Affective Computing , 1–1doi:10.1109/TAFFC.2018.2884015.

Bates, D., Mächler, M., Bolker, B., Walker, S., 2015. Fitting linear mixed-effects models using lme4. Journal of Statistical Software 67, 1–48. doi:10.18637/jss.v067.i01.

Braithwaite, J.J., Watson, D.G., Jones, R., Rowe, M., 2013. A guide for analysing electrodermal activity (eda) & skin conductance responses (scrs) for psychological experiments. Psychophysiology 49, 1017–1034.

Calefato, F., Lanubile, F., Maiorano, F., Novielli, N., 2018. Sentiment polarity detection for software development. Empirical Software Engineering 23, 1352–1382.

Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P., 2002. Smote: synthetic minority over-sampling technique. Journal of artificial intelligence research 16, 321–357.

Corporation, M., 2016. Touch mouse specification. https://www.microsoft.com/hardware/en-us/touch-technology#lightbox-touch-technology.

De Choudhury, M., Counts, S., 2013. Understanding affect in the workplace via social media, in: Proceedings of the 2013 conference on Computer supported cooperative work, ACM. pp. 303–316.

Dewan, P., 2015. Towards emotion-based collaborative software engineering, in: Cooperative and Human Aspects of Software Engineering (CHASE), 2015 IEEE/ACM 8th International Workshop on, IEEE. pp. 109–112.

D'mello, S.K., Kory, J., 2015. A review and meta-analysis of multimodal affect detection systems. ACM Computing Surveys (CSUR) 47, 43.

Ebert, F., Castor, F., Novielli, N., Serebrenik, A., 2018. Communicative intention in code review questions, in: 2018 IEEE International Conference on Software Maintenance and Evolution (ICSME), IEEE. pp. 519–523.

Ebert, F., Castor, F., Novielli, N., Serebrenik, A., 2019. Confusion in code reviews: reasons, impacts and coping strategies. IEEE International Conference on Software Analysis, Evolution, and Reengineering, SANER ; Conference date: 24-02-2019 Through 27-02-2019.

Ekman, P.E., Davidson, R.J., 1994. The nature of emotion: Fundamental questions. Oxford University Press.

Elfenbein, H.A., Ambady, N., 2002. On the universality and cultural specificity of emotion recognition: a meta-analysis. Psychological bulletin 128, 203.

Ericsson, K.A., Simon, H.A., 1993. Protocol analysis. MIT press Cambridge, MA.

Fagan, M.E., 1999. Design and code inspections to reduce errors in program development. IBM Systems Journal 38, 258.

Fritz, T., Begel, A., Müller, S.C., Yigit-Elliott, S., Züger, M., 2014. Using Psycho-physiological Measures to Assess Task Difficulty in Software Development. Proceedings of the 36th International Conference on Software Engineering , 402–413.

Gachechiladze, D., Lanubile, F., Novielli, N., Serebrenik, A., 2017. Anger and its direction in collaborative software development, in: Software Engineering: New Ideas and Emerging Technologies Results Track (ICSE-NIER), 2017 IEEE/ACM 39th International Conference on, IEEE. pp. 11–14.

Graziotin, D., Fagerholm, F., Wang, X., Abrahamsson, P., 2018. What happens when software developers are (un) happy. Journal of Systems and Software 140, 32–47.

Graziotin, D., Wang, X., Abrahamsson, P., 2013. Are happy developers more productive?, in: International Conference on Product Focused Software Process Improvement, Springer. pp. 50–64.

Hancock, J.T., Landrigan, C., Silver, C., 2007. Expressing emotion in text-based communication, in: Proceedings of the SIGCHI conference on Human factors in computing systems, ACM. pp. 929–932.

Hart, S.G., Staveland, L.E., 1988. Development of nasa-tlx (task load index): Results of empirical and theoretical research, in: Advances in psychology. Elsevier. volume 52, pp. 139–183.

Herbsleb, J.D., Klein, H., Olson, G.M., Brunner, H., Olson, J.S., Harding, J., 1995. Object-oriented analysis and design in software project teams. Human–Computer Interaction 10, 249–292.

Herbsleb, J.D., Mockus, A., 2003. An empirical study of speed and communication in globally distributed software development. IEEE Transactions on software engineering 29, 481–494.

Hernandez, J., Paredes, P., Roseway, A., Czerwinski, M., 2014. Under pressure: sensing stress of computer users, in: Proceedings of the SIGCHI conference on Human factors in computing systems, ACM. pp. 51–60.

Holmqvist, K., Nyström, M., Andersson, R., Dewhurst, R., Jarodzka, H., Van de Weijer, J., 2011. Eye tracking: A comprehensive guide to methods and measures. Oxford University Press, London.

Islam, M.R., Zibran, M.F., 2018. Sentistrength-se: Exploiting domain specificity for improved sentiment analysis in software engineering text. Journal of Systems and Software 145, 125–146.

Lutchyn, Y., Johns, P., Roseway, A., Czerwinski, M., 2015. Moodtracker: Monitoring collective emotions in the workplace, in: Affective Computing and Intelligent Interaction (ACII), 2015 International Conference on, IEEE. pp. 295–301.

Mäntylä, M.V., Graziotin, D., Kuutila, M., 2018. The evolution of sentiment analysisa review of research topics, venues, and top cited papers. Computer Science Review 27, 16–32.

McKinney, W., 2010. Data structures for statistical computing in python, in: van der Walt, S., Millman, J. (Eds.), Proceedings of the 9th Python in Science Conference, pp. 51 – 56.

Müller, S.C., Fritz, T., 2015. Stuck and frustrated or in flow and happy: Sensing developers' emotions and progress. Proceedings - International Conference on Software Engineering 1, 688–699.

Novielli, N., Girardi, D., Lanubile, F., 2018. A benchmark study on sentiment analysis for software engineering research, in: 2018 IEEE/ACM 15th International Conference on Mining Software Repositories (MSR), IEEE. pp. 364–375.

Paul, R., Bosu, A., Sultana, K.Z., 2019. Expressions of sentiments during code reviews: Male vs. female, in: 2019 IEEE 26th International Conference on Software Analysis, Evolution and Reengineering (SANER), IEEE. pp. 26–37.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12, 2825–2830.

Peng, F., Li, C., Song, X., Hu, W., Feng, G., 2016. An eye tracking research on debugging strategies towards different types of bugs, in: 2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC), pp. 130–134. doi:10.1109/COMPSAC.2016.57.

Picard, R.W., 1997. Affective Computing. MIT Press, Cambridge, MA, USA.

Picard, R.W., 1999. Affective computing for hci. Procs. 8th HCI International on Human-Computer Interaction: Ergonomics and User Interfaces , 829–833.

Pinheiro, J., Bates, D., DebRoy, S., Sarkar, D., R Core Team, 2017. nlme: Linear and Nonlinear Mixed Effects Models. URL: https://CRAN.R-project.org/package=nlme. r package version 3.1-131.

Pollak, J., Adams, P., Gay, G., 2011. Pam: A photographic affect meter for frequent, in situ measurement of affect, in: Proceedings of CHI, pp. 725–734.

R Core Team, 2015. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing. Vienna, Austria. URL: http://www.R-project.org/.

Riordan, M.A., Trichtinger, L.A., 2017. Overconfidence at the keyboard: Confidence and accuracy in interpreting affect in e-mail exchanges. Human Communication Research 43, 1–24.

Sano, A., Johns, P., Czerwinski, M., 2017. Designing opportune stress intervention delivery timing using multi-modal data, in: Affective Computing and Intelligent Interaction (ACII), 2017 Seventh International Conference on, IEEE. pp. 346–353.

Schmidt, P., Reiss, A., Dürichen, R., Van Laerhoven, K., 2018. Labelling affective states in the wild: Practical guidelines and lessons learned, in: Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers, ACM. pp. 654–659.

Schneider, K., Klünder, J., Kortum, F., Handke, L., Straube, J., Kauffeld, S., 2018. Positive affect through interactions in meetings: The role of proactive and supportive statements. Journal of Systems and Software 143, 59–70.

Schulze, J., Krumm, S., 2017. The virtual team player a review and initial model of knowledge, skills, abilities, and other characteristics for virtual collaboration. Organizational Psychology Review 7, 66–95.

Shimmer, 2016. Multi sensor management shimmer. http://www.shimmersensing.com/images/uploads/docs/Consensys_Spec_Sheet_v1.2_final.pdf.

Shu, L., Xie, J., Yang, M., Li, Z., Li, Z., Liao, D., Xu, X., Yang, X., 2018. A review of emotion recognition using physiological signals. Sensors 18, 2074.

Storey, M.A., Zagalsky, A., Singer, L., German, D., et al., 2017. How social and communication channels shape and challenge a participatory culture in software development. IEEE Transactions on Software Engineering , 185–204.

Tang, H., Tan, S., Cheng, X., 2009. A survey on sentiment detection of reviews. Expert Systems with Applications 36, 10760–10773.

Uwano, H., Nakamura, M., Monden, A., Matsumoto, K.i., 2006. Analyzing Individual Performance of Source Code Review Using Reviewers' Eye Movement. Eye tracking research & applications (ETRA) , 133–140.

van der Walt, S., Colbert, S.C., Varoquaux, G., 2011. The numpy array: A structure for efficient numerical computation. Computing in Science Engineering 13, 22–30. doi:10.1109/MCSE.2011.37.

Watson, D., Clark, L., Tellegan, A., 1988. Development and validation of brief measures of positive and negative affect. Journal of Personality and Social Psychology 54, 1063–1070.

Wrobel, M.R., 2013. Emotions in the software development process, in: 2013 6th International Conference on Human System Interactions (HSI), IEEE. pp. 518–523.

Appendix

Inter-rater agreement

Model diagnostic plots.

Table 6: Confusion matrix of raters' coding of comments emotionality.

	R2 (positive)	R2 (neutral)
R1 (positive)	17	82
R1 (neutral)	4	156

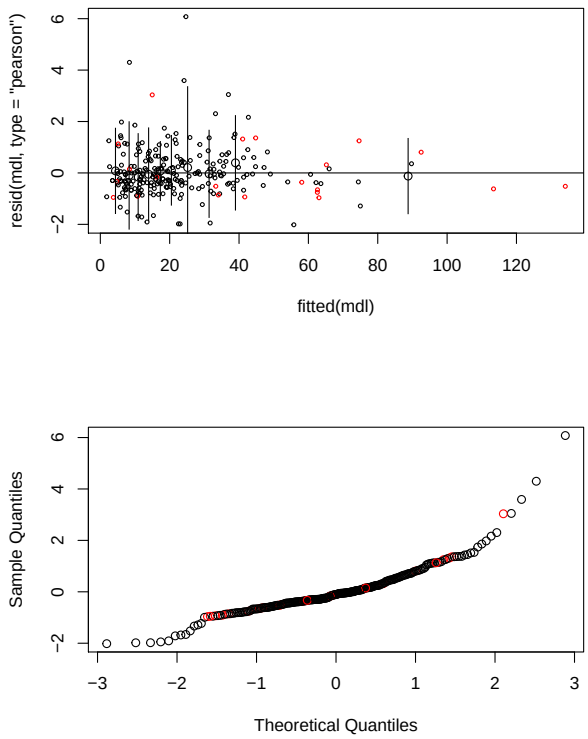

Figure 5: Diagnostic plots of the final model. The residual and Q-Q plot are based on Pearson residuals. Neutral comments are in black, comments with emotionality are illustrated in red.