Combining Biometric Data with Focused Document Types Classifies a Success of Program Comprehension

Toyomi Ishida
dept. Advanced Information Engineering,
National Institute of Technology, Nara College
Nara, Japan
a0932@stdmail.nara-k.ac.jp

Hidetake Uwano
dept. Information Engineering,
National Institute of Technology, Nara College
Nara, Japan
uwano@info.nara-k.ac.jp

Yoshiharu Ikutani
Division of Information Science,
Nara Institute of Science and Technology
Nara, Japan
ikutani.yoshiharu.ip8@is.naist.jp

ABSTRACT

Program comprehension is one of the important cognitive processes in software maintenance. The process typically involves diverse mental activities such as understanding of source code, library usages, and requirements. Systematic supports would be improved if the supports can be aware of such fine-grained mental activities during program comprehension. Here we aim to investigate whether biometric data can be varied according to such mental activity classes and conduct an experiment with program comprehension tasks involving multiple documents. As a result, we successfully classified the success/failure of the tasks at 85.2% from electroencephalogram (EEG) combined with focused document types. This result suggests that our metrics based on EEG and focused document types might be beneficial to detect developers' diverse mental activities triggered by different documents.

CCS CONCEPTS

• **Software and its engineering** → **Software evolution**; • **Applied computing** → **Bioinformatics**; • **General and reference** → *Metrics.*

KEYWORDS

program comprehension, EEG, eye movement, machine learning

ACM Reference Format:
Toyomi Ishida, Hidetake Uwano, and Yoshiharu Ikutani. 2020. Combining Biometric Data with Focused Document Types Classifies a Success of Program Comprehension. In *28th International Conference on Program Comprehension (ICPC '20), October 5–6, 2020, Seoul, Republic of Korea.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3387904.3389291

1 INTRODUCTION

Supports for program comprehension make software developers be more productive in their software implementation and debugging processes. Such systematic supports would be improved if the system can be aware of developers' mental states during program comprehension. For example, the support system might become possible to detect a software developer who does not understand the outputs of related source code and then visualize the calculation processes of the output values.

Program comprehension involves two-fold understanding of the target source code: Procedure (how it works) and objective (what it works for) [1, 2]. Understanding of "procedure" consists of various sub-processes such as understanding of control flows, roles of defined variables and functions, usage of API architecture. Furthermore, developers' attentions during program comprehension tend to be distributed on each line, block, method, class, and entire source code. Current systematic supports for program comprehension processes do not take these diverse sub-processes into account. Sensing and classifying developers' mental processes based on their biometric data would be beneficial to improve the quality of these systematic supports.

Our long-term research goal is to provide state-aware supports for developers' program comprehension processes by identifying their mental states based on biometric data. It is difficult to identify developers' mental states during program comprehension based solely on external observations because the processes mainly take place in the brain. Behavior tracking systems capture developers' program comprehension processes from human computer interaction data [3]. However, the tracking systems get no behavioral data if developers spend time to think without any interaction with computers. Another line of studies has investigated change tasks and patterns of documentation usage based on developers' behaviors and eye movements [4, 5]. Combining these context switching information with brain activity data might enable us to reveal developers' fine-grained mental processes during program comprehension. Additionally, this approach would be beneficial to classify the success/failure of program comprehension processes due to its sensitivity on internal cognitive processes.

In this study, we aim to predict successes and failures on program comprehension processes from developers' electroencephalogram (EEG) and eye movement data. Ishida et al. showed that the changes in EEG and eye movement data can be synchronized during program comprehension tasks [7]. Technically, the subjects' α spectrum in EEG and attention rates on source code were increased in early phases if they succeeded to comprehend a given code snippet while the changes have occurred lately if they failed. Here we use the synchronized changes in EEG and eye movement data to classify developers' success and failure on program comprehension. More dynamic and appropriate support might be enabled if the support system can be aware of developers' mental states from biometric data.

2 RELATED WORK

In the past ten years, biometric data has been attracting increased attention to quantitatively investigate or estimate the mental processes of software developers. Siegmund et al. contrasted brain activities during program output estimations against syntax error searches using functional magnetic resonance imaging (fMRI) and showed that program comprehension processes activated several left-lateralized brain regions [6, 8]. Peitek et al. developed a method to simultaneously measure fMRI signals and eye-movements to obtain a more comprehensive understanding of program comprehension [9]. Although fMRI is a powerful tool to visualize how program comprehension processes take place in the brain due to its high spatial-resolution, it consumes non-negligible monetary cost and unsuitable to measure in practical working environments.

Several recent studies in the software engineering domain have combined more light-weight biometric sensors with machine learning techniques to classify mental states of software developers [10]. Frits et al. trained a Naive Bayes classifier on three biometric sensors involving EEG, electrodermal activity (EDA), and eye movements to predict the difficulty of program comprehension tasks [11]. Müller et al. trained a decision tree classifier on EEG, EDA, eye movements, and heart-related data to classify developers ' mental states (i.e. stuck or inflow, happy or frustrated) and resulted in around 70 percent classification accuracies [12]. Both of their results suggested that combinations of two or more different biometric data could be useful to estimate developers' mental states or subjective difficulties.

Here we investigate whether biometric data can be varied according to fine-grained mental processes triggered by the types of focused documents. In particular, we investigated both the focused document types and EEG data recorded while subjects performed program comprehension tasks with multiple documents. Our combined analysis would be suitable for understanding the sub-processes of program comprehension since the EEG and eye movements can be measured in high temporal resolutions. Further, the devices are easy to apply in real environments such as company working spaces and classrooms for students.

3 EEG AND EYE MOVEMENTS

3.1 EEG

EEG is a method to record the electrical activity of the brain typically using electrodes placed on the scalp. The method measures voltage fluctuations as a potential difference between two electrodes over a period of time [13]. Typical derivation methods are classified as the standard or bipolar method. The standard method is employed when two electrodes need to be placed close to each other, while the bipolar method is used to measure the difference between two specific positions and remove the irrelevant background components. In addition, formal electrode positions are defined by several internationally-acknowledged systems such as the 10-20 systems. In the 10-20 system, the positions of nineteen electrodes except for the one for defining ground potential are determined.

Many studies have investigated the frequency components of EEG data decomposed by FFT (Fast Fourier Transform) or STFT (Short Time Fourier Transform) [13]. The famous classification of these frequency components is following: δ wave in 0.1 - 4 Hz; θ wave in 4 - 8 Hz; α wave in 8 - 13 Hz; β wave in 13 - 30 Hz; and γ wave in 30 - 100 Hz. The signals of each frequency component can differ depending on subjects' behaviors and mental states [11, 14, 15]. For instance, α wave becomes stronger when a subject is relaxed or concentrating on a task while β and *alpha* waves respectively become stronger and weaker when the person is puzzled or stressed. In this study, we use EEG as a biometric to classify programmers who successfully understand source code.

3.2 Eye Movement

Eye movement is recorded as a series of the spatial coordination (x and y) on a display that calculated from his/her eyeball movement [16] and the information allows us to infer where the subject looked at and how long each attention took on the place. Since software documents such as source code and requirement specifications consist of numerous lines, the eye movement analyses in software engineering studies keep playing an important role to enable line-wise analysis of program comprehension processes. Further, eye movement data has been employed to analyze the difference between novice and expert programmers during program comprehension and debugging. For instance, Behroozi et al. distinguished who understands source code via eye movement data [17] and Busjahn et al. demonstrated the differences in the ratio of linear eye movement patterns between experts and novices [18].

Eye movement analyses do not include the internal states of program comprehension. Long gaze-fixation time to a single line generally suggest that the focused line is important for the entire understanding or hard to understand. However, eye movement analyses can not distinguish these two internal situations because the data hardly contains information related to their mental activities. In this study, we used a history of document switching to determine which document a subject focused their attention during the program understanding tasks.

Our experimental environment presented several documents in a time and subjects can switch them through a tab interface. Because each tab window only contained a single document, the switching history directly indicates which document was focused at each period of time.

3.3 Potential Relationship between EEG and Eye Movement

Our key idea is to utilize the biometric pattern of successful program comprehension that can be summarized as "concentration of visual attentions to source code is followed by an increase in powers of EEG frequency components". Program comprehension is consisted of understanding various documents, such as source code and specifications, and programmers focus their attention on source code when they engage in program comprehension processes. An earlier work [7] showed that programmer's α spectrum tends to rapidly increase when he/she succeeded in program comprehension tasks, while the spectrum slowly increases if they failed the tasks. Therefore, the concentration of visual attentions on source code followed by the considerable increase of α spectrum would be a useful feature to determine his/her program comprehension processes are succeeded or failed. In this study, we hypothesize that an increase of α wave in the succeeded trials is observed earlier than those in the failure trials. To validate this hypothesis, we employ the EEG data and focused document type on each time point as a proxy of internal cognitive process and context switching. Our experiment investigates whether the success or failure of a program comprehension task can be classified by the combined metrics.

4 EXPERIMENT

For this study, we recruited five subjects with a major in computer science from National Institute of Technology, Nara College (all males, aged between 19 and 20 years). All had normal or corrected-to-normal vision and understood basic-level Java grammars.

4.1 Task Design

The experiment consisted of 16 tasks and in each task subjects were presented with three documents; (1) a Java code snippet with one class implementation, (2) a requirement specification that explained the purpose of the program, and (3) a question statement, e.g. the question "What will be the value of variable a at the sixth line in the second loop?" and the answer "a is 12". For each task, subjects were given 2.5 minutes to comprehend the documents and were required to answer the question verbally within the time. The experimenters manually judged subjects' answers and labeled each task as *success* or $failure$. Technically, we labeled *success* if a subject answered the question correctly within the given time and labeled $failure$ if the answer was incorrect or no answer available.

We prepared two difficulty levels (easy or difficult) in each task so that we could obtain similar sample numbers of *success* and $failure$ trials. For easy tasks the given code snippets consisted of a main method, single loop block, and single conditional branch, while the snippets in difficult tasks included a complex algorithm that had multiple methods or a recursive structure. Note that we randomized the presentation order of stimuli across subjects to minimize the potential order effects.

4.2 Data Collection

We used the NeXus-10 MARK II manufactured by Nanotech Image Ltd. to collect EEG data from all subjects while they performed the experimental tasks. We measured subjects' EEG with 256Hz sampling frequency on a single electrode position Pz based on the 10-20 system [19]. The ground electrode was located on right ear (A2), standard electrode was located on left ear (A1). In addition, we recorded all tab-switching histories while they performed the tasks and used them as an indicator of their focused document type. The experiments were performed in a quiet room with a subject and two experimenters. All subjects sit on a chair with arm- and foot-rests and were instructed to keep their body stable as possible to suppress potential artifacts due to body movements.

4.3 Data Analysis

We first calculated power spectrum of α waves from EEG data in each task, independently for each subject. Technically we divided EEG data on each task into 200 ms length and then calculated the power spectrum using STFT [13]. The calculated power spectrum was averaged over each five seconds to simplify the analysis. To mitigate the individual difference in EEG data, we normalized the averaged power spectrum based on the median value of each task. Next we quantified the fixation ratio of each document type (i.e. 'code', 'specification', 'question') from the tab switching histories recorded during the tasks. This procedure enabled us to identify which document the subject focused on because the tab interface used in this experiment displayed only one document in a time. The fixation ratio was finally averaged over every five seconds as percentage of the fixation time to each document type.

We here used a specific pattern in biometric data as a prime feature to classify a success of program comprehension. The pattern was summarized as 'the concentration of visual attentions on source code followed by the considerable increase of α spectrum' [7]. Both a fixation ratio on code snippets and an increase in α spectrum were calculated from the differences between adjacent time periods. Technically, let $ALPHA_{p,t}$ be a power spectrum of α wave in subject p at task t and let $EYE_{p,t}$ be a fixation ratio on the code snippet. Both metrics are divided into N periods ($n = 1, 2, ..., N$) for every five seconds where α_n is the averaged power of α spectrum on the period and eye_n is a ratio of time when the code snippet was displayed. Note that the last periods that lasted less than five seconds were excluded from our analysis.

$$\begin{aligned} ALPHA_{p,t} &= \{\alpha_1, \alpha_2, ..., \alpha_n\} \\ EYE_{p,t} &= \{eye_1, eye_2, ..., eye_n\} \end{aligned}$$

We then calculate $\alpha\text{-}inc_n$ and $eye\text{-}inc_n$ each corresponds to an increase/decrease in the metric at period n.

$$\begin{aligned} \alpha\text{-}inc_n &= \{\alpha_{n+1} - \alpha_n \mid N > n \geq 1\} \\ eye\text{-}inc_n &= \{eye_{n+1} - eye_n \mid N > n \geq 1\} \end{aligned}$$

We finally extracted the temporal difference between xth largest values of $\alpha\text{-}inc_n$ and $eye\text{-}inc_n$. Let Tx_α be the xth largest increase in the alpha spectrum observed at period t and let Tx_{eye} be the xth largest fixation ratio on source code. The temporal difference between Tx_α and Tx_{eye} is calculated as the following:

$$Ti_{eye}\text{-}Tj_\alpha = \{Ti_{eye} - Tj_\alpha \mid 1 \leq i \leq x, 1 \leq j \leq x\}$$

For instance, $eye_{T1}\text{-}\alpha_{T5}$ means the temporal distance between the first largest increase of the fixation ratio and the fifth largest increase of α spectrum. Positive values mean an increase of the fixation ratio appeared slower than α spectrum increase while negatives indicate the opposite. Here we classified a success of the task using Random Forest and both $x = 3$ and $x = 5$ parameters to examine the effect of x on the classification accuracies. The classification accuracies were evaluated by a leave-one-out cross validation procedure.

5 RESULTS AND DISCUSSION

We collected 80 data (16 tasks $\times$ 5 subjects) from 47 *success* and 33 $failure$ trials. We first evaluated classification accuracies on $x = 3$ using 70 data (success:39, failure:31); 10 data were excluded because of short measurement duration less than 15 seconds ($N < 4$). Table.1 shows the results of classification accuracies. The result

Table 1: Classification Accuracy ($x = 3$)

	Accuracy	Task time (s)	Success	Time up
All	0.843	75.4	0.557	0.243
Success	0.897	68.0	-	-
Failure	0.774	96.1	-	0.548
Easy	0.839	56.1	0.871	-
Difficult	0.846	102.7	0.308	0.436
Subject 1	0.923	66.6	0.615	0.231
Subject 2	0.692	79.7	0.308	0.231
Subject 3	0.813	75.0	0.563	0.188
Subject 4	0.867	71.6	0.667	0.333
Subject 5	0.923	84.4	0.615	0.231

Table 2: Classification Accuracy ($x = 5$)

	Accuracy	Task time (s)	Success	Time up
All	0.852	88.8	0.463	0.315
Success	0.880	81.1	-	-
Failure	0.828	104.8	-	0.586
Easy	0.813	66.7	0.875	-
Difficult	0.868	105.6	0.289	0.447
Subject 1	0.875	92.8	0.375	0.375
Subject 2	0.727	85.1	0.273	0.273
Subject 3	0.636	94.1	0.455	0.273
Subject 4	1.000	79.1	0.615	0.385
Subject 5	1.000	94.4	0.545	0.273

demonstrated the classification accuracy on all data was 0.843, indicating that the accuracy was clearly higher than chance (0.5). This suggests that our proposed metrics could be useful to predict a success of program comprehension. Interestingly, the accuracies for *success* (0.897) and *failure* (0.774) trials were different. This difference might be affected by the structure of *failure* data as a mixture of two failure types; failures due to the incorrect answer (14 data) and failures when subjects spent all of the given time without answering (17 data). Subjects might not perform well program comprehension processes when they met the time limit without a clear increase in fixation ratio on source code.

We evaluated the difference in classification accuracies across subjects. The difference between the highest (subject 1 and 5) and lowest (subject 2) accuracies was 0.231. This might be caused by the ratio of success trials in subject 2. The amount of data was not enough to learn the biometric pattern in success trials because the success ratio was lowest in subject 2. Another possible interpretation could be the variations in time duration, which calculated from task start to making the first answer, because the time on failure trials was longer than those on success trials (see Table 1). However, the correlation analysis showed no significant correlation between the task time and classification accuracy on each subjects ($r = -0.266, p = 0.665$).

Table 2 shows the classification accuracies using parameter $x = 5$. We used 54 data (success:25, failure:29) and 26 data were excluded because their lengths were not enough to calculate $T5_{\alpha}$ and $eye3_{T5}$. As a result the overall classification accuracy (0.852) was similar to the value on $x = 3$. In addition, the classification accuracies on success trials (0.880) were higher than failure trials (0.828) showing a similar tendency on $x = 3$. In comparison with the results on $x = 3$, the classification accuracies of subject 1 and 3 were decreased because insufficient training data. The number of samples decreased from 13 to 8 (38.5%) in subject 1 and 16 to 11 (31.3%) in subject 3, respectively. Correlation analysis between task time and classification accuracy on each subject in $x = 5$ shows no significant correlation ($r = 0.028, p = 0.965$).

We additionally investigated the mean decrease in Gini coefficient as a measure of how each variable contributes to the resulted classification accuracies. Figure 1 indicated that $T1_{eye} - T3_{\alpha}$ and $T3_{eye} - T1_{\alpha}$ were higher than other variables. In both variables the absolute values of failure were higher than success. The value of success/failure were -7.9 and -15.9 at $T1_{eye} - T3_{\alpha}$; 5.6 and 9.8 at $T3_{eye} - T1_{\alpha}$ respectively. This analysis suggest that subjects failed to understand source code had a longer period of time between the increases in fixation ratio and α spectrum, supporting our hypothesis to a success of program comprehension that 'the concentration of visual attentions on source code followed by the considerable increase of α spectrum'.

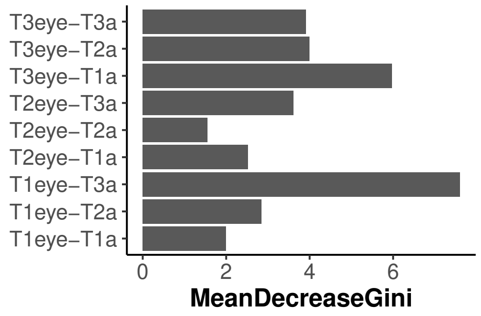

Figure 1: Mean Decrease of Gini Coefficients ($x = 3$)

6 CONCLUSION

In the present study, we classified a success of program comprehension tasks based on α spectrum powers and focused document types. Specifically, we used the temporal distance between increases in α spectrum power and fixation ratio on source code. Although several inter-subject differences were observed, the resulted classification accuracy (0.852) was clearly higher than chance. The results indicate that our proposed metric is useful to classify success/failure of program comprehension processes. This encourages us to develop a real-time support system that detect fine-grained mental processes conducted by a software developer. The proposed metric might be improved by taking the potential effects of gender, age, and individual programming expertise into account. In addition, other biometrics such as flickers and β power spectrum could be beneficial to classify the success/failure of a program comprehension process.

ACKNOWLEDGMENT

This work was supported by JSPS KAKENHI Grant Number JP16-K00114 and JP18J22957.

REFERENCES

[1] Mayrhauser, A. V. and Vans, A. M., "Program comprehension during software maintenance and evolution," in Computer, Vol.28, No.8, pp. 44-55, 1995.

[2] Xu, S., "A cognitive model for program comprehension," In Proceedings of the Third ACIS International Conference on Software Engineering Research, Management and Applications (SERA), pp.392-398, 2005.

[3] Xia, X., Bao, L., Lo, D., Xing, Z., Hassan, A. E., Li, S., "Measuring program comprehension: A large-scale field study with professionals," IEEE Transactions on Software Engineering, pp.951-976, 2017.

[4] Kevic, K., Walters, B., Shaffer, T., Sharif, B., Shepherd, C., D., Fritz, T., "Eye gaze and interaction contexts for change tasks - Observations and potential," J. Syst. Softw. 128: pp.252-266, 2017.

[5] Bavota, G., Canfora, G., Penta D., M., Oliveto, R., Panichella, S., "An empirical investigation on documentation usage patterns in maintenance tasks", ICSM, pp.210-219, 2013.

[6] Siegmund, J., Peitek, N., Parnin, C., Apel, S., Hofmeister, J., Kastner, C., Begel, A., Bethmann, A., and Brechmann, A., "Measuring neural efficiency of program comprehension," In Proceedings of the 11th Joint Meeting on Foundations of Software Engineering(ESEC/FSE), pp.140-150, 2017.

[7] Ishida, T. and Uwano, H., "Synchronized analysis of eye movement and EEG during program comprehension," In Proc. 6th International Workshop on Eye Movements in Programming (EMIP), 2019.

[8] Siegmund, J., A. Brechmann, Apel, S., Kastner, C., Liebig, J., Leich, T., and Saake, G., "Toward measuring program comprehension with functional magnetic resonance imaging," In Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering (FSE), No.24, 2012.

[9] Peitek, N., Siegmund, J., Parnin, C., Apel, S., Hofmeister, J. C., Brechmann, A., "Simultaneous measurement of program comprehension with fMRI and eye tracking: a case study," The ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM), pp.1-10, 2018.

[10] Lee, S., Hooshyar, D., Ji, H., Nam, K., Lim, H., "Mining biometric data to predict programmer expertise and task difficulty," Cluster Computing, pp.1097-1107, 2018.

[11] Fritz, T., Begel, A., Müller, S. C., Yigit-Elliott, S., Züger, M., "Using psychophysiological measures to assess task difficulty in software development," In Proceedings of the International Conference on Software Engineering (ICSE), pp.402-413, 2014.

[12] Muller, S. C., and Fritz. T., "Stuck and frustrated or in flow and happy: Sensing developers' emotions and progress," In Proceedings of 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering, Vol.1, pp.688-699, 2015.

[13] Satheesh, Kumar, J., Bhuvaneswari, P., "Analysis of electroencephalography (EEG) signals and its categorization - A study," In Proceedings of the International Conference on Modeling, Optimization and Computing (ICMOC), Vol.38, pp.2525-2536, 2012.

[14] Birbaumer, N., Elbert, T., G M Canavan, A., Rockstroh, B., "Slow potentials of the cerebral cortex and behavior," Physiological Reviews, Vol.70, pp.1-41, 1990.

[15] Zuger, M. and Fritz, T., "Interruptibility of software developers and its prediction using psycho-physiological sensors," In Proceedings of 33rd Annual ACM Conference on Human Factors in Computing Systems, pp.2981-2990, 2015.

[16] Duchowski, A. T., "Eye tracking methodology," Springer 2006.

[17] Behroozi, M., Lui, A., Moore, I., Ford, D., and Parnin, C., "Dazed: measuring the cognitive load of solving technical interview problems at the whiteboard," In Proceedings of the International Conference on Software Engineering (ICSE), pp.93-96, 2018.

[18] Busjahn, T., Bednarik, R., Begel, A., Crosby, M., Paterson, J. H., Schulte, C., Sharif, B., Tamm, S., "Eye movements in code reading: Relaxing the linear order," In Proceedings of the 23rd International Conference on Program Comprehension (ICPC), pp.255-265, 2015.

[19] Jasper, H. H., "The ten-twenty electrode system of the International Federation", Electroencephalogr. Clin. Neurophysiol., pp.370-375, 1958.