How do Programmers Use the Internet? Discovering Domain Knowledge from Browsing and Coding Behaviors

Ko Watanabe[†], Yuki Matsuda[‡], Yugo Nakamura[*], Yutaka Arakawa[*], and Shoya Ishimaru[†]
[†] *Department of Computer Science, University of Kaiserslautern & DFKI GmbH*, Kaiserslautern, Germany,
{ko.watanabe, shoya.ishimaru}@dfki.de
[‡] *Graduate School of Science and Technology, Nara Institute of Science and Technology*, Nara, Japan,
yukimat@is.naist.jp
[*] *Graduate School and Faculty of Information Science and Electrical Engineering, Kyushu University*, Fukuoka, Japan,
{y-nakamura, arakawa}@ait.kyushu-u.ac.jp

***Abstract*—The Internet is an effective tool for learners to gain new knowledge. Often, people use search engines (e.g., Google) rather than accessing websites directly. People have their search techniques to find specific information. In particular, people with domain knowledge tend to search more efficiently than novices. By understanding the gap between people with domain knowledge and novices, the novice can understand the path to becoming an expert. Therefore, in this study, we wanted to know what differences exist in search and programming behavior with and without domain knowledge. In this experiment, we asked a group with and without domain knowledge to solve ten programming problems and collected search logs (input knowledge) and compilation logs (output knowledge). Specifically, the first dataset consisted of 13 participants who had taken a university programming class. The second dataset consisted of 20 participants who had not taken a programming class and had no domain knowledge. We examined differences in search and compilation behavior based on participants' domain knowledge from this data. Since we observed a difference between each group when referring to the correlation coefficient, we performed a binary classification of novice and experienced participants using Random Forest, and achieved an average precision of 0.95, indicating that there were different trends in behavior with and without domain knowledge.**

***Index Terms*—Learning management systems, web browsing, programming, behavioral data, pattern analysis, internet privacy.**

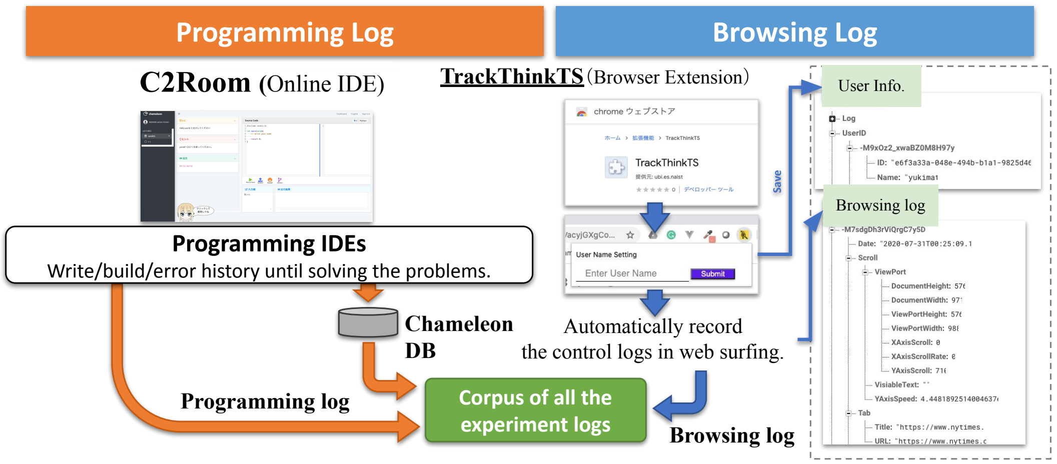
I. Introduction

Search skill is important in many fields. An effective and efficient way of using an internet search engine leads anyone to access new knowledge and skills. In this context, Google has proposed *Google Search Education* [1] which aims to improve students become better searchers. Kim *et al.* has done on analyzing how adults daily use web search [2]. They collected data from 40 participants and discover nine types of search roles. Hence the study has proposed a type of different search behaviors. Hölscher *et al.* investigated how search behavior is different between experts and newbies [3]. They found the significance of domain knowledge in general.

People often use web search or browsing while programming to identify an answer to undefined questions. It has been already discovered that developers spend 20% of the time on browsing activity while coding in average [4]. Hence, analysis of web browsing behavior and programming activity is significant for improving skills. Regarding the importance of search skills for developers, Brandt *et al.* investigated how programmers use online resources while coding [4]. They discovered that programmers tend to use a search engine for just-in-time learning of approaches and new skills, clarify and extend their existing knowledge, and remind themselves of details deemed not worth remembering. They also found that depending on different purposes of search, programmers have different styles and duration in queries. This trend appears in both in-the-set conditions and in-the-wild conditions. Li *et al.* has proposed work on analyzing novice and expert programmers [5]. However, their work did not focus together on comparing differences in behavior by combining log data from web browsing and program compiling activity.

Although people acquire knowledge and utilize it at the same time or in parallel, many existing studies have tracked either the input or output of knowledge. The objective of our work is to utilize two systems that record browsing behavior and coding behavior, respectively, and discover comprehensive characteristics from their fusion. In particular, we examined how domain knowledge differs in browsing and coding behavior when solving programming questions. By discovering these differences, it can support novice programmers understand the gap between experts and work on becoming an expert. The research questions addressed in this work are as follows.

- RQ1: Do participants with and without domain knowledge each have similarities on web browsing and programming activities?
- RQ2: Do participants with and without domain knowledge each have differences on web browsing and programming activities?
- RQ3: Can we estimate whether participants have domain knowledge or not by features calculated from web browsing and programming activities?


Programming Log
Browsing Log
C2Room (Online IDE)
TrackThinkTS (Browser Extension)
Programming IDEs
Write/build/error history until solving the problems.
Chameleon DB
Programming log
Corpus of all the experiment logs
Automatically record the control logs in web surfing.
Browsing log
Save
User Name Setting
Enter User Name
Submit
User Info.
Browsing log

```
Log
UserID
  -M9xOz2_xwaBZ0M8H97y
    ID: "e6f3a33a-048e-494b-b1a1-9825d46
    Name: "yukimat
-M7sdgDh3rViQrgC7y5D
  Date: "2020-07-31T00:25:09.1
  Scroll
    ViewPort
      DocumentHeight: 576
      DocumentWidth: 971
      ViewPortHeight: 576
      ViewPortWidth: 988
      XAxisScroll: 0
      XAxisScrollRate: 0
      YAxisScroll: 716
    VisiableText: ""
    YAxisSpeed: 4.448189251400463776
  Tab
    Title: "https://www.nytimes.
    URL: "https://www.nytimes.c
```

Fig. 1: Overview of the system. Store programming activity and web browsing log as a corpus.

II. Related Work

The overall analysis of web browsing behaviors is introduced by Hsieh-Yee *et al.* [6]. Both children's and adults' browsing activity logs are collected. Most of the studies on children described how they interact with the web. Research on a web search of adults focuses on discovering search patterns. Also aim to investigate the effects of selected factors on search behavior, including information organization and presentation, type of search task, web experience, cognitive abilities, and affective states. In other kinds of approaches, Huang *et al.* have proposed a method of reformulating query search [7]. They have identified that adding or removing words, the substitution of words, acronym expansion, and spelling correction as effective.

The researcher also focuses on the developer's web search. Holmes collects 35 months' worth of queries to analyze programmer's search queries [8]. They discovered that programmers use three or more search facts. Rahman *et al.* discover how effective to use code-related queries [9]. They compare participants' search queries into code and non-code-related searches. They discovered that code-related searches often occur in debugging situations where programmers directly copy and paste the error message. However, search by error message often has fewer results on the website. Hence, this research stated that an effective search needs the use of both code and non-code queries to find a proper website.

In addition to web search logs, researchers focus on the behaviors during web browsing for knowing the detailed contexts of users. Liu *et al.* reported scroll behavior (number of scrolls, and their direction) were used to study users browsing strategies [10]. Milisavljevic *et al.* reported there are differences in scroll behaviors depending on the type of web browsing task [11]. It suggests that scroll behaviors reflect the user's purpose of browsing.

Regarding these previous researches, there are not many works that discover both web browsing and programming activities. One of the research groups that pioneered this field implemented a web browser logging system [12], [13]. It is the concept of implementing a system to collect web browsing data and programming compile logs. However, their proposed research question *Is there any difference in web search skills between successful students and the others?* is not yet identified. In this paper, we have succeeded in discovering differences between programmers with and without domain knowledge from web browsing and programming activity. Hence we have found an answer to the research question that has not yet been identified.

III. Approach

To answer the research questions, we utilize two software that records web browsing behavior and programming activities, respectively, and calculate cross-sectional metrics from their logs. This section explains the details of each system and the definitions of the metrics.

A. Activity-logging software

Figure 1 shows the overview of our data recording architecture. A user solves programming questions on C2Room (online integrated development environment) by installing TrackThinkTS (browser extension).

C2Room – We utilize an online IDE for programming called C2Room [14] to collect participants' programming activities. C2Room is designed for programming courses and has functions to display problems, receive answers, and return outputs of compiling results. After pre-processing raw log data of C2Room, we collect the timing and the validity (success or failure in the compile) while each submission in addition to the submitted answer.

TrackThinkTS – We leverage TrackThinkTS [13] for recording browsing activities. TrackThinkTS has been proposed by Makhlouf *et al.* as a privacy-aware browsing log tracker. It monitors following browsing actions and collects browsing logs. First, information of the visited website (a website title, URL, HTML contents, viewport width/height, and document width/height) are collected. They are stored whenever tab-related actions happen such as tab creation (opening a new tab), tab activation (selecting an existing tab),

tab update (reloading a website, or opening a new page in the same tab), and tab delete (closing the existing tab). Second, users' specific actions during browsing the website such as scrolling logs (scroll speed, scroll length, and visible text after finishing scroll) and clipboard logs (the content of the clipboard) are collected. This is the biggest difference between previous browsing history. Compared to normal browsing history, it is possible to collect information such as which parts of the page the user viewed in detail and how long they stayed on each page. Also, the advantage of TrackThinkTS compared to other browser loggers is that it provides a user-friendly interface for filtering each log. Therefore participants of a study using TrackThinkTS can delete privacy-sensitive information before submitting their log file to an experiment conductor.

B. Evaluation metrics

Domain Knowledge (DK) – Estimating whether a user has domain knowledge or not is one of our main objectives. This ground truth value can be obtained from not two software but an experimental setup such as recruiting two participant groups (with and without DK) or conducting a pre-test.

Score – The score of the programming problems is calculated from the log of C2Room and represents levels of task-specific knowledge or problem-solving. In addition to DK, we also compare this value with other metrics.

Mean and Variance of Browsing Time (MBT, VBT) – We define Browsing Time (BT) as the time difference from *tab activate* until the last *window scroll* on a web page collected by TrackThinkTS. We store this value for each page and calculate their means and variances as metrics.

Compile Search Ratio (CSR) – How often a user search website to solving one problem can be calculated from the combination of C2Room and TrackThinkTS. We define this metric as follows.

$$CSR = \frac{\text{Number of web page access}}{\text{Number of compiles}}$$

Compile Error Search Ratio (CESR) – CESR is an extension of CSR that focuses on only the number of incorrect complies instead of all complies.

$$CESR = \frac{\text{Number of web page access}}{\text{Number of compile errors}}$$

Input Output Ratio (IOR) – We are interested in how much time a user spends on input and output for a given task. In the case of this browsing programming, we define the indicators as follows.

$$IOR = \frac{\text{Time duration of search}}{\text{Time duration of search} + \text{Time duration of coding}}$$

C. Domain knowledge estimation

First, apply random under sampling to balance the number of dataset A and B. Input features are explained in Section III-B. Then apply 10-fold cross validation to do a binary classification. We prepared machine learning models of *logisitic regression*, *linear discrimiinant analysis*, *k-nearest neighbors vote*, *decision tree classifier*, *random forest*, *gaussian naive bayes*, and *support vector machine*.

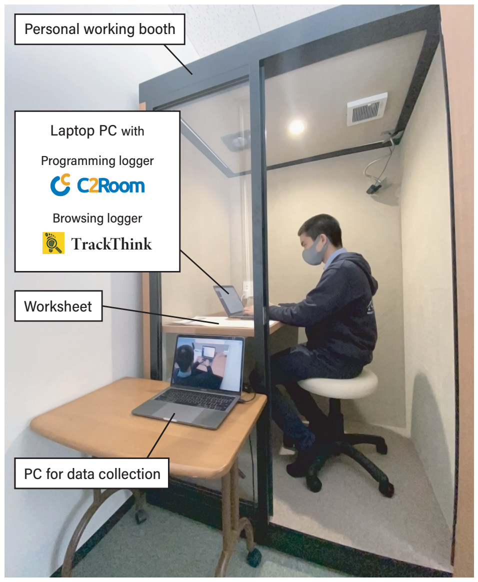


Fig. 2: Experimental settings. Collect programming and browsing log while solving scheme questions.

IV. Experimental Design

This section explains the design of our experiment. It is design to evaluate our research questions presented in Section I.

A. Experimental procedure

The experimental settings are shown in Figure 2. Both C2Room and TrackThinkTS were installed into participant's Laptop PC before the experiment. Under this condition, the experiment was conducted with the following procedure.

1) Explanation about the purpose of the study, experimental settings, and tools is provided to the participant. Only the person who is agreed with the conditions can participate this experiment.
2) The participant enters the personal working booth, and both programming and browsing logger will be started to record data.
3) The participant solves given scheme questions on the programming editor of C2Room. We did not restrict the order for answering questions, although supposing the participant will start from the easy part. The result will be recorded by C2Room at every compile execution.
4) When the participant faces unknown syntax and/or compile errors, they will search in the web. This behavior will be recorded by TrackThinkTS.
5) If compile result seems correct, the participant submit the answer and proceed to next question. The participant can correct their answer anytime by returning to previous answered questions.
6) Until finishing to solve all questions or elapsing one hour, the participant repeat procedure 3–5.

TABLE I: Scheme Questions

ID	Question
1	Define variable *PI* as 3.14.
2	Write a scheme to show PI $*5^2$.
3	Write a scheme to show $(-b+\sqrt{b^2-3ac})/3a$.
4	Define function *areaDisk* to calculate circle area from radius *r*.
5	Define function *areaRing* to calculate circle area from outer and inner diameter *D, d*.
6	Define function *d2y* that convert US currency dollar *d* to Japanese currency yen. Note that 1 US dollar is 108.43 yen.
7	Define function *e2d* that convert European currency *e* to United states dollar. Note that 1 euro is 1.1069 US dollar.
8	Define function *p2e* that convert British currency pond *p* to European currency. Note that 1 pond is 1.1632 euro.
9	Define function *p2y* that convert British currency pond *p* to Japanese currency. Use *d2y*, *e2d*, and *p2e* in the previous questions.
10	Define function *c2f* that convert Celsius *C* to Fahrenheit. Note that $f = 1.8c + 32$.

B. Task Setting

In this experiment, we selected *Scheme (Racket)*, which is one of the dialects of LISP languages, as the programming language of the task [15]. There are the following reasons for selection: 1) It is employed in programming lectures of several universities, because of its simple language specification; 2) It is customized for the usage of lectures, hence, its language specification is not usually known by students except for lecture attendees. Table I shows the questions they are asked to answer. We have prepared 10 questions which are sorted by their difficulty. These questions are selected on our own. Easy questions require less code and understanding of the domain knowledge. These questions are simple to answer for participants with the domain knowledge. However, participants without domain knowledge were required to search for the basic rules of the scheme. Therefore, we set these questions for classifying users with and without domain knowledge.

C. Dataset and participants

We conducted experiments with two groups: novice students (Dataset A) and lecture attendees (Dataset B). We have done experiment with two different types of group to discover and compare how the difference will occur in each groups. Here, we explain detail of each dataset. Our experiments do not include EU citizens, hence, General Data Protection Regulation (GDPR) is not applied to our recordings. We define domain knowledge as the participants who took classes related to the Scheme in the university lecture.

Dataset A – group of lecture attendees (With domain knowledge): Data were collected from 13 unique participants (12 males and one female). All participants took lectures on Scheme. Hence they have some domain knowledge of the grammar of the programming language.

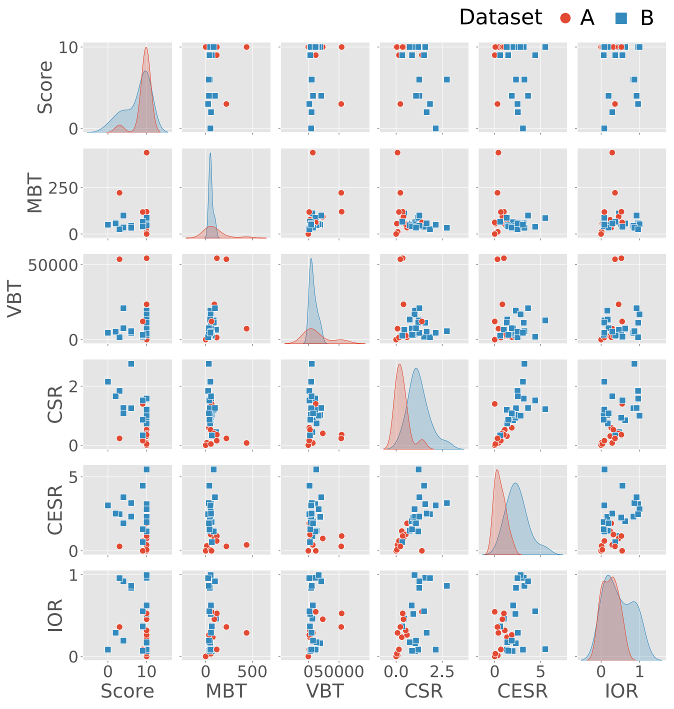


Fig. 3: Pair plot shows semantic mapping of dataset A and B. Dataset A is a group with domain knowledge and dataset B is a group without domain knowledge. The point to note in this plot is that CSR and CESR have unique peaks in the histogram of each dataset.

Dataset B – group of novice students (Without domain knowledge): Data were collected from 20 unique participants (20 males). All participants did not take lectures on Scheme. Hence they do not have domain knowledge and required to search for the grammar of the programming language.

To compare datasets A and B, we asked both participants to solve question ID from 1 to 10 from Table I.

V. Results

This section explains the result of the experiment. Figure 3 shows the pair plot group by each dataset A and B. Dataset A is lecture attendees who have domain knowledge and dataset B is the novice students. The calculation methods for all metrics are shown in Section III-B. From the result, we confirm that CSR and CESR have different peaks in the histogram. Looking at the histogram of MBT and VBT, there is no significant peak for dataset A. In dataset B, with IOR, the peak values indicate two characteristic groups in the histogram. Lastly, looking at the pair plot of CSR and CESR, datasets A and B have different groups. Dataset A is grouped in low CSR and CESR, for dataset B is in high CSR and CESR.

Next, the correlation of each metric in detail. Figure 4 shows the correlation heatmap including all participants. The significant correlation with DK is CESR with -0.69 and CSR with -0.68. Also, the score and the CSR have a negative correlation

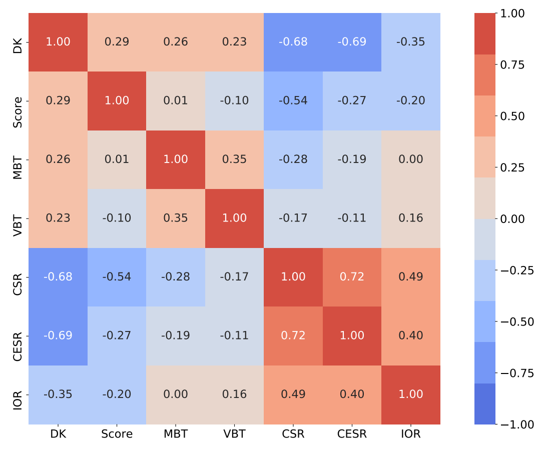
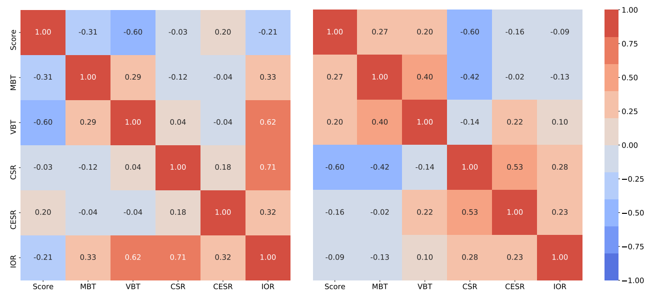


Fig. 4: Correlation heatmap of all dataset.



(a) Dataset A (b) Dataset B

Fig. 5: Correlation heatmap of each dataset A and B.

of −0.54. CESR and CSR was a significant indicator for the domain knowledge and score. Figure 5 is the heatmap for each dataset A and B. For dataset A, score and VBT have a negative correlation of −0.60. For dataset B, score and CSR have a negative correlation of −0.60. For each dataset, the key metric for obtaining a high score was different.

Lastly, the result of predicting the existence of the domain knowledge. Table II shows the prediction rate of the domain knowledge. The highest mean accuracy was 0.95 recorded by Random Forest with a standard deviation of 0.15. The parameter setting of the model is hyper parameters *the number of trees: 100*, *Criterion: Gini impurity*, and *the number of max features: 7 (square root of the number of features)* are used for classification. The lowest mean accuracy was Support Vector Machine. Table III shows the result of feature importance for the model using Random Forest. The most important input was CESR and the least was IOR.

VI. Discussion

In this section, we discuss the result proposed in Section V. We answer the research questions proposed in Section I.

A. RQ1: Do participants with and without domain knowledge each have similarities in web browsing and programming activities?

Figure 3 shows the pair plot with a semantic mapping of dataset A (with domain knowledge) and B (without domain knowledge). Looking at CSR and CESR in red square,

TABLE II: 10-fold cross validation result of binary prediction of lecture attendees and novice student.

Machine Learning Model	Mean Accuracy	Standard Deviation
Logistic Regression	0.90	0.20
Linear Discriminant Analysis	0.70	0.40
K-nearest Neighbors Vote	0.70	0.33
Decision Tree Classifier	0.85	0.32
Random Forest Classifier	**0.95**	**0.15**
Gaussian Naive Bayes	0.85	0.23
Support Vector Machine	0.60	0.49

TABLE III: Feature importance of Random Forest Classifier.

Rank	1	2	3	4	5
Feature	CESR	CSR	MBT	VBT	IOR
Weight	0.50	0.24	0.13	0.07	0.05

there are similarities in pattern for each dataset A and B according to the histogram. Also, with a combination of both CSR and CESR as features, the classification of with and without domain knowledge can be done. For IOR we could find similarities for dataset A. This is regarding the initial knowledge that participants can produce code without web browsing. However, it is interesting that the IOR of dataset B can be separated into two groups. One group of participants try to discover answers by increasing the duration of input knowledge time. Whereas another group of participants has low IOR, meaning they are not browsing at all. Assuming that participants with low IOR and high scores have confident domain knowledge. Also by looking at Figure 4 in DK (domain knowledge) column, it is discovered that CSR, CESR, and IOR have negative correlations. These results imply that amount of web page access increase for a novice programmer. As a result, participants with and without domain knowledge showed similarities in CSR and CESR for each group, but not in Score, MBT, VBT, and IOR. Also, a negative correlation was found between domain knowledge and CSR, CESR, and IOR.

B. RQ2: Do participants with and without domain knowledge each have differences in web browsing and programming activities?

In order to answer this question, we will discuss by looking at Figure 5. For dataset A, we found a strong negative correlation between VBT and score. Whereas for dataset B, a strong negative correlation can be seen with CSR. So for dataset A, as the score increases, participants tend to search for less time duration on each web page access. For dataset B, as the score increases, the number of compile increases or searches decreases. This heatmap presents that dataset A and B each correlate with different features. As a result, we have found differences in participant domain knowledge.

C. RQ3: Can we estimate whether participants have domain knowledge or not by features calculated from web browsing and programming activities?

Regarding the first research question, CSR and CESR can role as a feature for classifying the group of lecture attendees and novice students. Table II shows the accuracy of binary classification. As a result, the Random Forest classification scored the highest mean prediction accuracy of 0.95. Table III shows the feature importance of the Random Forest classifier. According to the result, the CESR is the most important feature. As a result, participants with lecture attendees and novice students can be predicted.

VII. Limitation

In this section, we discuss the limitation of our work. First, the limitation of our work is the setting of the programming questions. We come up with the language that is not common for the participants and hence remove domain knowledge. However, the questions were relatively easy for the participants and hence variance in scores did not appear. The majority of participants received perfect scores, regardless of their domain knowledge. If we could set the questions that make the variance score, we could also try to predict a programming score from web browsing and programming activities.

Second, eye gaze information during web browsing is not tracked. TrackThinkTS can track web browsing activity as participants access and scroll through web pages. However, it does not track where participants were looking on the website. Using techniques such as eye-tracking can improve the quality of the web browsing log corpus. If we can obtain participants' gaze while looking at a website, the flow of domain knowledge acquisition will be more clearly visualized.

Third, the functionality of the IDE (C2Room) could be improved. For example, when used in this experiment, the timing of when participants saw a new question was not tracked. If recording were done, it would be possible to calculate, for example, the time it takes a participant to see a question and submit an answer. It would then be possible to estimate the difficulty of the task based on the time it took the participant to solve the problem, and to calculate correlations with the presence or absence of domain knowledge.

Fourthly, the size of the dataset is relatively small. In our experiment, we conduct a programming class at a Japanese university. Hence, our dataset does not include non-Japanese participants. For our future work, we need to increase the number of participants, especially from a variety of nationalities.

Lastly, our limitation is that the definition of domain knowledge was binary classified. We focus on participants with and without domain knowledge in this study, but it could be separated into smaller groups. Such as the group with domain knowledge might also be separated by how long they have been experienced. On the other hand, we also see the programming problem setting as positive. This is because we have confirmed that the presence or absence of domain knowledge can be classified by search and programming activities, which the scores did not reveal. However, we did not consider the experiences in detail. It could also be possible that participants may have experienced different programming knowledge, which may have resulted in higher search skills than pure novices. Future work may include a clearer definition of what constitutes a participant's novice status.

VIII. Conclusion

In conclusion, we identified how participants with domain knowledge perform web browsing and programming activities in problem-solving. From web browsing and programming activity logs, we predict the existence of domain knowledge with 0.95 accuracy by Random Forest. The activity of web browsing and programming changes with domain knowledge.

Acknowledgment

This work was supported by JST, CREST Grant Number JPMJCR16E1, Japan and JSPS Fostering Joint International Research (B) Grant Number 20KK0235.

References

[1] G. Inc., "Google search education," 2020. [Online]. Available: https://www.google.com/insidesearch/searcheducation/

[2] J. Kim, B. McNally, L. Norooz, and A. Druin, *Internet Search Roles of Adults in Their Homes*. New York, NY, USA: Association for Computing Machinery, 2017, p. 4948–4959. [Online]. Available: https://doi.org/10.1145/3025453.3025572

[3] C. Hölscher and G. Strube, "Web search behavior of internet experts and newbies," *Computer networks*, vol. 33, no. 1-6, pp. 337–346, 2000.

[4] J. Brandt, P. J. Guo, J. Lewenstein, M. Dontcheva, and S. R. Klemmer, *Two Studies of Opportunistic Programming: Interleaving Web Foraging, Learning, and Writing Code*. New York, NY, USA: Association for Computing Machinery, 2009, p. 1589–1598. [Online]. Available: https://doi.org/10.1145/1518701.1518944

[5] A. Li, M. Endres, and W. Weimer, "Debugging with stack overflow: Web search behavior in novice and expert programmers," 2022.

[6] I. Hsieh-Yee, "Research on web search behavior," *Library & Information Science Research*, vol. 23, no. 2, pp. 167–185, 2001.

[7] J. Huang and E. N. Efthimiadis, "Analyzing and evaluating query reformulation strategies in web search logs," in *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, ser. CIKM '09. New York, NY, USA: Association for Computing Machinery, 2009, p. 77–86. [Online]. Available: https://doi.org/10.1145/1645953.1645966

[8] R. Holmes, "Do developers search for source code examples using multiple facts?" in *2009 ICSE Workshop on Search-Driven Development-Users, Infrastructure, Tools and Evaluation*, 2009, pp. 13–16.

[9] M. M. Rahman, J. Barson, S. Paul, J. Kayani, F. A. Lois, S. F. Quezada, C. Parnin, K. T. Stolee, and B. Ray, "Evaluating how developers use general-purpose web-search for code retrieval," in *Proceedings of the 15th International Conference on Mining Software Repositories*, ser. MSR '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 465–475. [Online]. Available: https://doi.org/10.1145/3196398.3196425

[10] C. Liu, J. Liu, and Y. Wei, "Scroll up or down? using wheel activity as an indicator of browsing strategy across different contextual factors," in *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval*, ser. CHIIR '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 333–336. [Online]. Available: https://doi.org/10.1145/3020165.3022146

[11] A. Milisavljevic, F. Abate, T. Le Bras, B. Gosselin, M. Mancas, and K. Doré-Mazars, "Similarities and differences between eye and mouse dynamics during web pages exploration," *Frontiers in Psychology*, vol. 12, 2021.

[12] K. Nagano, Y. Arakawa, and K. Yasumoto, "Trackthink: a tool for tracking a thought process on web search," in *Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2017 ACM International Symposium on Wearable Computers*, 2017, pp. 681–687.

[13] J. Makhlouf, Y. Arakawa, and K. Watanabe, "A privacy-aware browser extension to track user search behavior for programming course supplement," in *International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services*. Springer, 2021, pp. 783–796.

[14] dTosh Inc., "Product | c2room," 2020. [Online]. Available: https://c2room.jp/

[15] S. Adams and C. Hanson, "Mit scheme user's manual," *Massachusetts Institute of Technology*, 1995.