TYPE Original Research
PUBLISHED 22 July 2024
DOI 10.3389/fcomp.2024.1412458

OPEN ACCESS

EDITED BY
Andrej Košir,
University of Ljubljana, Slovenia

REVIEWED BY
Radoslava Stankova Kraleva,
South-West University "Neofit Rilski," Bulgaria
Daisuke Saito,
Waseda University, Japan

*CORRESPONDENCE
Toyohisa Nakada
nakada@nuis.ac.jp

RECEIVED 23 April 2024
ACCEPTED 09 July 2024
PUBLISHED 22 July 2024

CITATION
Nakada T and Miura M (2024) Extracting typing game keystroke patterns as potential indicators of programming aptitude.
Front. Comput. Sci. 6:1412458.
doi: 10.3389/fcomp.2024.1412458

Extracting typing game keystroke patterns as potential indicators of programming aptitude

Toyohisa Nakada[1]* and Motoki Miura[2]

[1]Department of Information Systems, Faculty of Business and Informatics, Niigata University of International and Information Studies, Niigata, Japan, [2]Department of Information and Communication Systems Engineering, Faculty of Engineering, Chiba Institute of Technology, Chiba, Japan

This study attempted to determine whether individuals possess programming aptitude solely based on keystroke information from typing games where participants type computer programs. The participants were students enrolled in university programming courses. The results indicated that using typing speed alone as an indicator achieved an accuracy of 0.71, while employing a custom machine learning model achieved an accuracy of 0.83. Additionally, it was found that individuals with programming aptitude tended to type the enter key relatively slower compared to other keys.

KEYWORDS

typing game, programming aptitude, machine learning, programming education, keystroke analysis

1 Introduction

A typing game is a game that requires the player to quickly and accurately type a given text or passage. These games have been recognized not only for their entertainment value but also for their practical applications, such as serving as tools for creating corpora related to English word spelling errors (Tachibana and Komachi, 2016), or improving children's fine motor skills (McGlashan et al., 2017). Meanwhile, the extent to which players understand the content of the text they are typing has not been extensively discussed. Therefore, in this study, we investigated how the typing speed and rhythm differ depending on whether players understand the code they are typing in programming language typing games (Nakada and Miura, 2023). In other words, this implies that it is possible to estimate the level of comprehension of the program being typed based solely on the keystroke information.

Research on keyboards traditionally falls within the realm of Human-Computer Interaction (HCI), where studies have focused on aspects such as usability (Wang et al., 2021) and constructing models for human acquisition of keyboard skills (Pinet et al., 2022). However, beyond serving as input devices that transmit keystrokes to computers, there is also research that extends into areas such as estimating user emotions from keystroke rhythms (Yang and Qin, 2021). This study contributes to the applied research in keyboard studies. In typing games, users interact with the same program code, but differences in keystroke movements may arise between those who understand the meaning behind the code and those who do not. This research analyzes whether there are variations in motor actions related to keystrokes between these two groups. Studies treating perception, cognition, and behavior as mutually influencing factors are outlined by Hommel et al. (2001). This research specifically focuses on the cognitive aspect of computer program comprehension within this framework.

This study's novelty lies in developing a machine learning model capable of correctly identifying whether a person understands the computer program they are typing, based solely on typing keystroke information, achieving an accuracy rate of approximately 80%. Furthermore, the analysis of this machine learning model suggests that individuals who understand the program tend to press the Enter key relatively slower. Importantly, these characteristics appear not as a result of learning but potentially as innate traits from the early stages of programming education. These findings were made possible through advanced machine learning techniques, which traditional statistical methods alone would have struggled to achieve. The contribution of this research is enabling the estimation of program comprehension solely from keystrokes, thereby potentially influencing programming education significantly. The summary of the research process and contributions is as shown in Table 1.

In this paper, we first discuss the research methodology in Section 2. We detail the collection of keystroke data from typing games, the test to determine program comprehension, and the development of the machine learning model. In Section 3, raditional statistical methods focusing solely on typing speed are able to explain whether individuals understand programs with only 70% accuracy. Subsequently, we present results showing that applying machine learning can estimate comprehension with approximately 80% accuracy. Moreover, we highlight a significant finding that individuals who understand the program tend to press the Enter key relatively slower. Section 4 interprets these findings, suggesting that the identified characteristics manifest early in programming courses, hinting at innate rather than learned traits. Finally, in Section 5, we summarize the paper's findings.

2 Methods

We collected typing data from students at Niigata University of International and Information Studies who were enrolled in a programming course. The participants were asked to type a series of fixed programming codes as part of a typing game. Furthermore, participants were administered comprehension tests at the end of the programming course period.

2.1 Typing game keystroke data

The programming classes for the beginner and pre-intermediate courses at the university consist of 180 min of instruction per week, spanning 15 weeks. The participants consist of university students attending the class, ranging in age from 18 to 21. Typing games were conducted once each week at the beginning of the class. Understanding of the typed programs is measured through tests administered in the final week. The typing data used for analysis were collected within two weeks of the test administration, and the data closest to the test date were selected.

Table 2 summarizes an overview of the data used for analysis. In addition to absolute typing speed, which measures the overall typing speed, we also employ relative normalized typing speed to analyze variations in typing speed within individual participants. This relative normalized typing speed is normalized between 0 and 1 to eliminate the influence of individual differences in typing speed. Additionally, the data include information on typing errors.

Relative normalized typing speeds were record the milliseconds it takes to type the correct key in a typing game, and in the case of an error, the interval from the mistaken key. This is based on the hypothesis that the timing of typing breaks may reveal characteristics related to code comprehension. For example, when attempting to type a word like "while," a player with a good understanding of the code may consistently strive to type it correctly, even if they make a typing error along the way. In contrast, a player with less code comprehension might stop their typing input midway through typing "while." To make it easier to identify such patterns, the data used for analysis includes intervals from the previous keystrokes, including errors, until the correct key is typed. In other words, this data is used to analyze where typing input is paused or interrupted.

2.2 Code comprehension test

The participants were students enrolled in a university programming course, which consisted of 15 weeks of classes, with each class lasting 180 min per week. A comprehension test was conducted in the final week of the course. The comprehension test is a program that creates software meeting specified specifications, encompassing the most fundamental structures of programming such as loops, conditional statements, input/output operations, and so forth.

The participants target both the beginner course, designed for those encountering programming for the first time, and the pre-intermediate course, aimed at students who have completed the beginner course. The learning objectives for the beginner course include attaining a level of understanding where participants comprehend all programs typed in the typing game.

2.3 Machine learning for analysis

The machine learning used in this study is known as supervised learning, which involves providing correct data to the model to train it on a certain input data. Specifically, typing keystroke information is set as input data, and the output indicates whether the program understands it or not (Burkart and Huber, 2020). While supervised learning models are expected to achieve high classification accuracy, it has been traditionally acknowledged that understanding why the model succeeds or fails in classification is challenging, as the model is often considered a black box. In this study, we first develop this machine learning model independently to verify the extent of classification accuracy achieved. We do not delve deeply into explaining the classification in this study.

The developed model is illustrated in Figure 1. The convolutional layer denoted as Conv1D is commonly used in image recognition tasks, as it helps analyze images by breaking them down into smaller parts. When dealing with input data

TABLE 1 The overview of the research process and contributions.

Research Question	Can keystroke dynamics alone differentiate between individuals who comprehend the program code they are typing in typing games and those who do not?
Data collection	1. Typing game keystroke data indicating when and which keys were pressed.
	2. Results from the program comprehension test.
	We collected data from 144 participants and analyzed the data from 112 participants who could clearly be identified as understanding the code or not.
Research methodology 1	Statistically analyze keystroke dynamics to determine if the typed code is understood. Estimating program comprehension solely based on average typing speed. It was explainable with approximately 70%.
Research methodology 2	Using machine learning techniques to determine if the typed code is understood. According to our developed machine learning model, we were able to estimate comprehension with approximately 80% accuracy. Furthermore, individuals who understood the program were found to press the Enter key relatively slower compared to other keys.
Contributions of the study	1. Typing speed has been shown to correlate with the understanding of programming.
	2. A machine learning model has been developed that can estimate programming understanding based on keystroke dynamics with approximately 80% accuracy.
	3. It has been shown that individuals who understand programming tend to press the Enter key relatively slower compared to other keys.
	4. Keystroke characteristics suggest that they reflect inherent abilities individuals possess rather than something acquired through learning.

TABLE 2 Features of typing data used for analysis.

Absolute typing speed	Information on when and which keys were pressed is acquired in milliseconds.
Relative normalized typing speed	The delay time is the time it takes to press a specific key from the previous Relative latency in my typing data key. The delay time is normalized using the maximum time in a typing game as 1 and the minimum time as 0.
Mistyped keys	It is data that records which key was intended to be pressed when a typing error occurred.

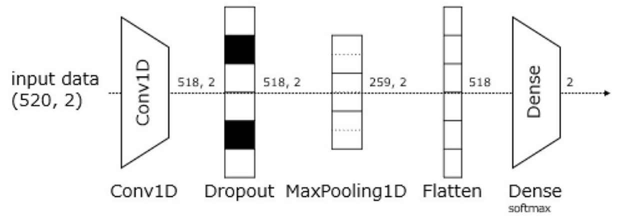


FIGURE 1
Machine learning model.

from images, multiple layers of convolutional layers are stacked to enable each layer to recognize features of different sizes simultaneously, such as large and small objects within the image. In contrast, for typing data, where the size of words is typically fixed (e.g., while, for), a single layer is considered sufficient. Additionally, while images have two axes (width and height), typing data is one-dimensional, only along the time axis. Subsequent to these convolutional layers, conventional techniques used in image recognition are applied. Dropout is utilized to intentionally reduce the amount of data, thus improving the generalization performance of classification. This is followed by a MaxPooling1D layer to extract the maximum value and a Flatten layer to connect to the final fully connected layer. The output consists of two nodes: one node outputs a large value when the program understands the input, while the other outputs a large value when the program fails to understand it.

3 Results

3.1 Comprehension test and absolute typing speed

The participants consist of students enrolled in both the beginner course, where understanding programs typed in the typing game serves as an educational objective, and the pre-intermediate course, which builds upon the beginner course by introducing more advanced programs. While both courses' comprehension tests are scored out of 30 points, the scoring for pre-intermediate course students is designed such that achieving half of the total points (15 points) on the comprehension test indicates understanding of programs typed in the typing game. Therefore, the analysis focuses on the score obtained by adding 15 points to the raw score of pre-intermediate course students.

Furthermore, in order to determine whether participants understand the programs typed in the typing game based on the results of the comprehension test, scores ranging from 20 to 30 points will be excluded from the analysis. Participants scoring below 20 points will be considered as not understanding, while those scoring above will be considered as understanding. Additionally, anticipating the implementation of machine learning, it is necessary to avoid artificially inflating the classification accuracy by creating multiple sets of learning data for each participant. Therefore, only the data acquired initially for each participant will be retained, while subsequent data will be discarded.

Within 2 weeks of taking the comprehension test, a total of 144 participants had recorded their typing

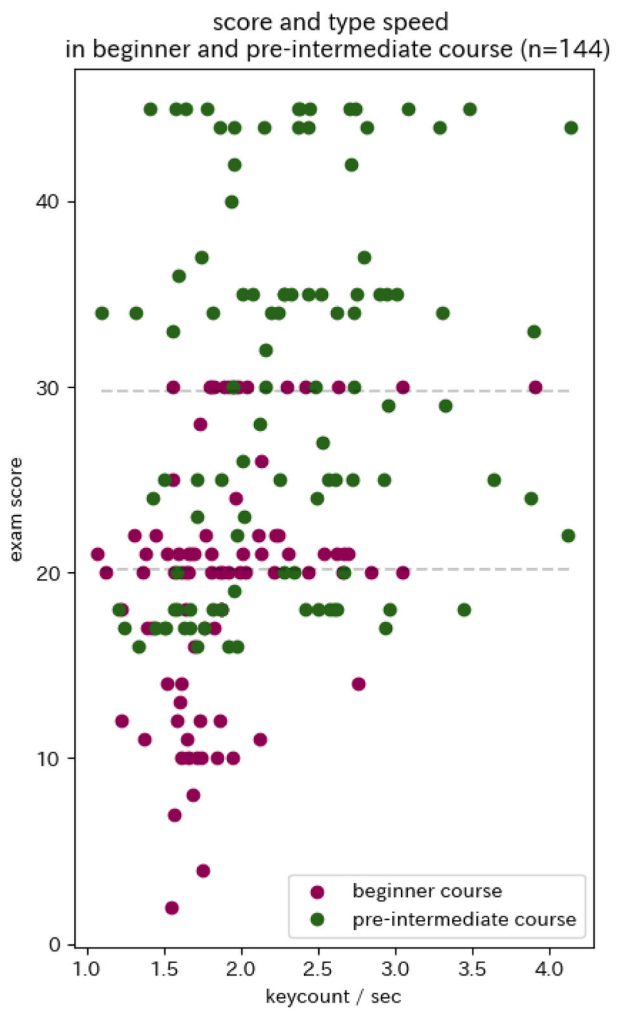


FIGURE 2
Absolute typing speed and comprehension test.

game performance. Figure 2 illustrates the relationship between typing speed and comprehension test results. The horizontal axis represents typing speed, measured as the number of keys typed per second in the typing game. Additionally, the vertical axis displays comprehension test results, adjusted to include an additional 15 points for participants in the pre-intermediate course. Subsequent analysis will categorize scores below 20 as indicative of a lack of understanding of the programs typed in the typing game, while scores of 30 or above will be considered as indicating understanding.

Since normality was not confirmed by the Shapiro-Wilk test (p <0.05), Spearman's rank correlation coefficients were computed. The results showed that for the combined data from both courses, r = 0.41, p <0.001. When analyzed separately by course, the correlation coefficients were r = 0.35, p <0.001 for the beginner course, and r = 0.25, p = 0.011 for the pre-intermediate course.

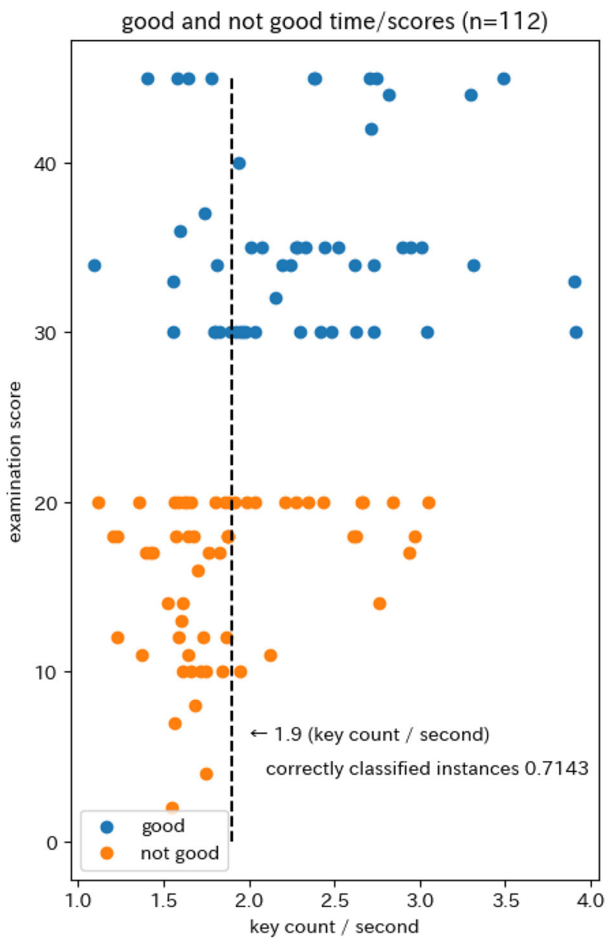


FIGURE 3
Classification of comprehension based on absolute typing speed.

3.2 Comprehension assessment based on absolute typing speed

Figure 3 illustrates the results of linear classification regarding whether participants understand the program based on their typing speed. Participant data excludes scores between 20 and <30, where understanding is ambiguous, and further, anticipating comparison with machine learning models to be conducted later, only the initial comprehension test results for participants enrolled in both courses are retained, with subsequent data being discarded.

The results showed that people who understand how the program works tend to type fixed programming codes faster. However, the accuracy was about 0.7143, and there were about 30% of people who could type quickly without understanding the program, or vice versa.

3.3 Analysis of characteristics by key type

Thomas et al. (2005) collected all the keylogs of students typing during regular programming exercises, not typing games, and analyzed the differences in keystrokes for each key between those

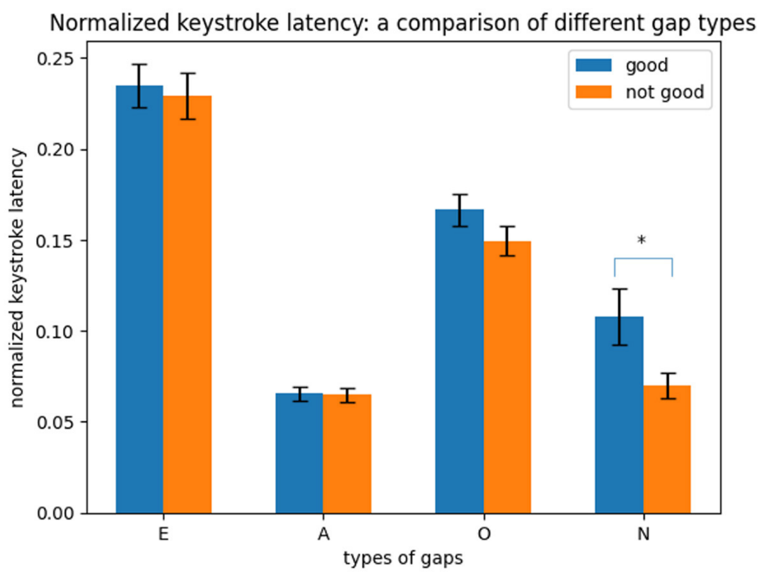
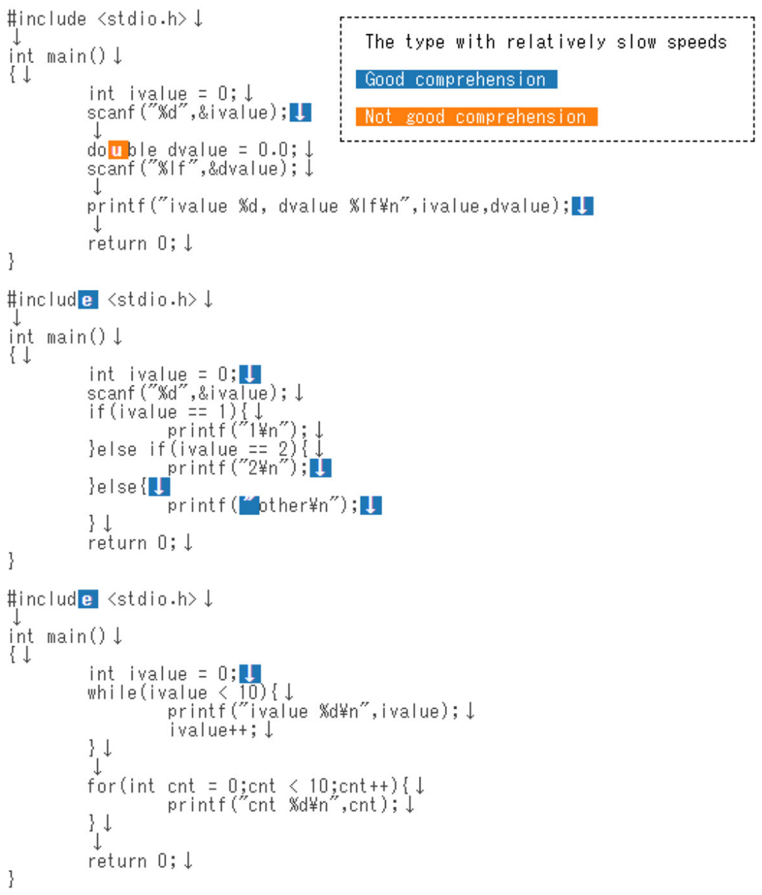


FIGURE 4
Normalized keystroke latency a comparison of different gap types. *indicates significant differences ($p < 0.05$) observed in means of two independent samples.



```
#include <stdio.h>↓
↓
int main()↓
{↓
          int ivalue = 0;↓
          scanf("%d",&ivalue);↓
          ↓
          double dvalue = 0.0;↓
          scanf("%lf",&dvalue);↓
          ↓
          printf("ivalue %d, dvalue %lf¥n",ivalue,dvalue);↓
          ↓
          return 0;↓
}

#include <stdio.h>↓
↓
int main()↓
{↓
          int ivalue = 0;↓
          scanf("%d",&ivalue);↓
          if(ivalue == 1){↓
                    printf("1¥n");↓
          }else if(ivalue == 2){↓
                    printf("2¥n");↓
          }else{↓
                    printf("other¥n");↓
          }↓
          return 0;↓
}

#include <stdio.h>↓
↓
int main()↓
{↓
          int ivalue = 0;↓
          while(ivalue < 10){↓
                    printf("ivalue %d¥n",ivalue);↓
                    ivalue++;↓
          }↓
          ↓
          for(int cnt = 0;cnt < 10;cnt++){↓
                    printf("cnt %d¥n",cnt);↓
          }↓
          ↓
          return 0;↓
}
```

FIGURE 5
Code heatmap: distribution of characteristics between individuals who understand the typed program based on relative typing speed and those who do not.

who understood programming and those who did not. First, we define the key types as follows:

- B: Keys for browsing within the editor
- A: Alphabets
- N: Numbers
- O: Others

Next, Thomas et al. (2005) analyze the points where the key type changes.

- E: Keys that change the key type from the previous key
- Symbol: Keys that do not change the key type

The results showed that the delay time of E, where the type changes, is shorter for people who understand the program. For example, when trying to type "ab = 10," it is encoded as AEEO according to Thomas's definition. The fact that E is typed quickly means that "ab = 10" is treated as a single chunk. In other words, people who understand the program have a larger chunk size when typing than those who do not.

In the context of typing a fixed program like a typing game, the results of this study aligned with those depicted in Figure 4. Statistically significant differences in keystroke gaps were found for N. Particularly, there were no notable differences observed for E as observed by Thomas et al. (2005).

3.4 Comprehension assessment based on relative normalized typing speed

The "Relative normalized typing speed" is an indicator that shows the fast and slow aspects of typing within an individual subject, and it represents data normalized between 0 and 1. When comparing this relative typing speed between groups of individuals who understand the program and those who do not through a mean comparison test, significant differences were observed in the area highlighted in yellow in Figure 5. The yellow highlights indicate areas where individuals who understand the program type slower. Additionally, there is one instance of green text (u) in Figure 5, indicating areas where individuals who do not understand the program type slower. The downward arrow represents the Enter key. Upon reviewing these results, it can be observed that individuals who understand the program tend to press the Enter key relatively slower compared to other keys.

3.5 Comprehension assessment based on machine learning

In Section 2.3, we trained a custom-developed machine learning model where the input consisted of the delay times of all keys represented as Relative normalized typing speed, and the output was whether the individual understood the program or not.

The results evaluated through 3-hold cross-validation are presented in Table 3. We achieved a classification accuracy of 0.83 on average. This value surpasses the baseline of 0.7143, which is simply linearly separable based on typing speed alone, by more than 0.1. This indicates that the fast and slow aspects of an individual's typing, akin to the rhythm of typing games, serve as valuable indicators for determining whether the individual understands the program they are typing.

TABLE 3 Evaluation of the machine learning model.

Test set number	Number of data	Accuracy rate
1	38	0.8421
2	37	0.8108
3	37	0.8378
Average	37.33	0.8304

3.6 Comprehension and typing errors

There might be different characteristics in the information pertaining to typing mistakes between individuals who understand the program and those who do not. For instance, it's conceivable that individuals who do not understand the program make more mistakes compared to those who do. Figure 6 illustrates the frequency distribution of typing mistakes. The horizontal axis represents the number of keys with mistakes made within a single typing game, while the vertical axis represents the number of participants with that frequency of mistakes. No significant differences are particularly evident.

Next, the keys with typing mistakes are shown in Table 4. For example, if the missed key with a score of "good" is "n," it indicates the number of times the key "n" was attempted but failed to be typed correctly. Since the same keys often appear in the top 10 rankings, albeit with different positions, it is suggested that there is no significant difference in the major characteristics of typing mistakes between the two groups.

4 Discussion

4.1 Keystroke and comprehension of the typing text

In the era when there was a profession called a typist, there were studies conducted to devise models for analyzing these typists (Card et al., 1980; Salthouse, 1986). During that era's research, it was believed that the performance of typists did not depend on their comprehension of the text they were typing (Salthouse, 1984).

On the other hand, recent research has reported differences in keystroke logs between tasks that involve simply copying text, such as typing games, and tasks that require constructing text, such as composing emails (Conijn et al., 2019). However, it has not been investigated whether understanding the text being typed in tasks that involve copying text affects keystrokes.

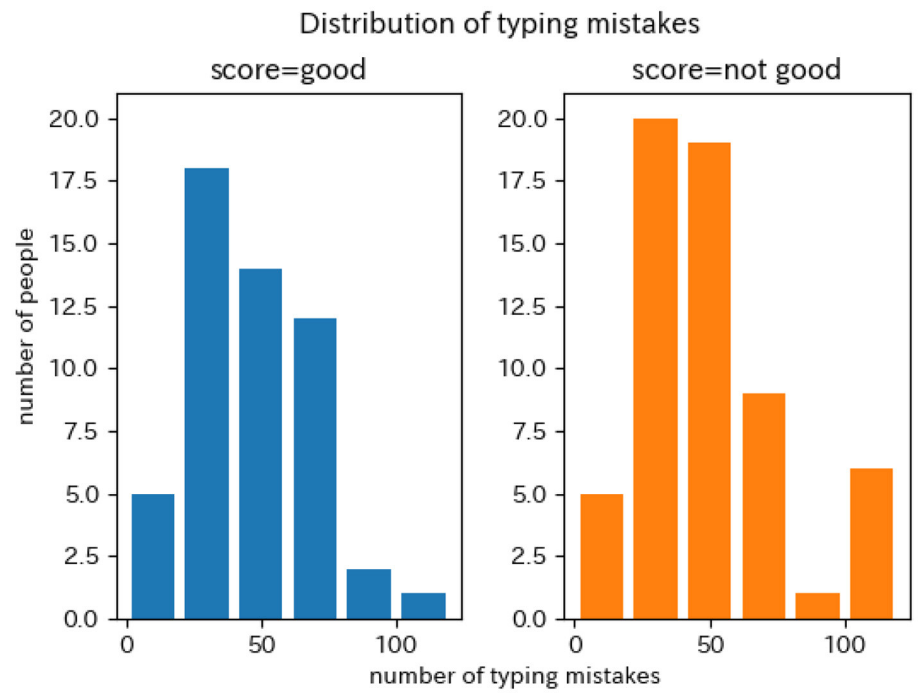


FIGURE 6
Distribution of typing mistakes.

4.2 Personal authentication using keystroke dynamics

Authentication is the process of determining who the person operating the computer is. While passwords are the most commonly used method, creating and managing strong passwords can be challenging. Therefore, methods utilizing personalized biometric information or characteristics of movement exist for authentication. Keystroke dynamics (Shanmugapriya and Padmavathi, 2009) is a field of study that focuses on how individuals type rather than what they type, utilizing these dynamics for authentication purposes. Gedikli and Efe (2019) has reported an identification accuracy of 94.7% using mechanisms of deep learning. These studies operate under the assumption that there are individual differences in typing movements, akin to identifying whether a program understands keystrokes based solely on typing patterns, which is the premise of our study.

TABLE 4 Ranking of the 10 most frequently mistyped keys along with their average error rates.

score=good		score=not good	
Missed key	Average count	Missed key	Average count
n	2.85	i	3.77
(space)	2.73	l	2.7
l	2.62	.	2.33
;	2.37	;	2.27
i	2.31	}	2.27
v	2.04	n	2.25
(	1.85	(	2.22
a	1.77	(space)	2
e	1.75	t	1.67
d	1.73	#	1.52

4.3 Keystroke and cognitive abilities

Wetherell et al. (2023) reported that there are changes in typing rhythm under stress. This suggests that keystroke dynamics not only rely on static characteristics of individuals but also on dynamically changing movements. There is also research aimed at detecting long-term changes from keystrokes. Hossain et al. (2021) and Holmes et al. (2022) are involved in studying cognitive impairment detection through keystroke behavior. According to Alfalahi et al. (2022), these studies are numerous, suggesting that keystrokes are a promising indicator for detecting cognitive impairments.

There is also research attempting to infer innate individual traits from keystrokes. Kovačević et al. (2023) endeavors to estimate personality traits, such as those described by the Big Five model (Goldberg, 1992), through keystroke dynamics. Additionally, Pinet (2024) demonstrated in a comparative experiment between groups capable and incapable of touch typing that the former group not only typed words displayed on the screen faster but also exhibited shorter response times in tasks involving repeated movements, such as repeating verbally heard words. This suggests that typing proficiency may correlate with language processing abilities.

4.4 Programming learning

Despite the multitude of research, practical examples, and continually evolving effective learning materials in programming education, it is still recognized as a challenging task for learning (Cheah, 2020; Kadar et al., 2021). There is a study that suggests that a necessary ability for mastering programming is fluid reasoning (Prat et al., 2020). Fluid reasoning, proposed alongside crystallized reasoning by Cattell (1943), is considered to be the capacity to identify relationships, while crystallized reasoning is thought to be the capacity that results from the habituation of abilities discovered through fluid reasoning. However, the definition of fluid reasoning is not fixed, and according to Kievit et al. (2016), fluid reasoning is defined as the ability to solve new abstract tasks, irrespective of task-specific knowledge. This definition is considered important for acquiring programming skills.

Analysis of keystrokes during programming, as revealed by Thomas et al. (2005), has shown that individuals with better programming performance tend to type a greater number of words as a single chunk compared to those with poorer performance. However, the relationship between keystrokes in copy tasks like typing games and programming performance remains unclear. In recent years, extensive research has been conducted using various methods, including keystroke analysis, to measure student performance in programming courses (Choi et al., 2023). Edwards et al. (2020) analyzed keystrokes of students during programming courses and discovered a consistent correlation between typing speed and grades, albeit with variations across different programming languages. They also reported that a machine learning model using random forests can predict student grades from keystrokes with an accuracy rate of approximately 60%–70%. Furthermore, some studies aim not only to directly estimate grades but also to identify students needing assistance during programming courses based on keystroke behavior. Zhao et al. (2021) reported the ability to estimate with 94% accuracy when students are facing difficulties, while Shrestha et al. (2022) highlighted the importance of pause times in keystrokes as a critical factor in grade estimation.

4.5 Characteristics of typing: student growth or programming aptitude

Lindemann et al. (2007) demonstrated that there is a correlation between the cognitive recognition of numerical magnitudes and manual movement. This suggests that internal cognitive processes such as knowing and understanding manifest as observable external phenomena, such as bodily movements. For instance, in the case of typing games, it is believed that understanding a program leads to changes in typing behavior, indicating growth in the participant.

If typing rhythm characteristics represent student growth, we should observe changes between the beginning and the end of the lecture course. Of course, some students may have already learned about programming to some extent before the start of the course. However, it is unlikely that all students will have understood programming at the start of the course. Therefore, we will conduct a time series comparison between the group of students who are eventually judged to have understood the program and the group who are judged not to have understood it, from the beginning to the end of the course.

Figures 7–9 depict the temporal changes in typing characteristics that distinguish individuals who have demonstrated understanding of the programming being typed vs. those who have not, as revealed in our study. Figure 7 represents the absolute typing speed, Figure 8 illustrates the average latency in pressing the return key, and Figure 9 shows the results of understanding judgments by the machine learning model. The horizontal axis of each graph represents the normalized date within the 15-week lecture course, with the first day of class designated as 0 and the last day of class designated as 1. The translucent band around the fitted regression line represents a 95% confidence interval.

From these results, it is suggested that students who achieve high scores on comprehension tests administered at the end of the course, regardless of which indicators are difficult to conclusively determine due to large data variance, possess these characteristics from the beginning of the course. If these characteristics represent student growth, then the difference between the group eventually understanding the program and the group not understanding it at the beginning of the course should not be clearly distinguished until closer to the end of the course.

It might indicate a tendency to treat a typing game, or any program being typed, not as a mere sequence of meaningless characters, but rather, to carefully perceive it as meaningful line by line. This attentiveness correlates with fluid reasoning ability (Cochrane et al., 2019), and fluid reasoning is a necessary skill for acquiring programming knowledge (Prat et al., 2020).

Furthermore, the fact that the characteristics extracted by the machine learning model appear from the early stage of the class period suggests that they may represent innate aptitude rather than student growth. This is an important finding, as it suggests that it may be possible to identify students who have a natural aptitude for programming early on and provide them with the resources they need to succeed.

However, it must be noted that the data for this typing game were obtained within the context of a programming class. This implies that participants were typing what they were learning in the programming course as part of the typing game. Therefore, unlike conventional typing games that assess speed and accuracy, there is a possibility that participants perceived this activity as a part of their learning process, hence attributing it with some degree of learning time. Due to the lack of data regarding this aspect, further analysis cannot be conducted. This stands as a limitation of the present study.

5 Conclusion

This study sheds light on the intriguing relationship between typing rhythm and programming aptitude among university students enrolled in programming courses. The analysis of typing

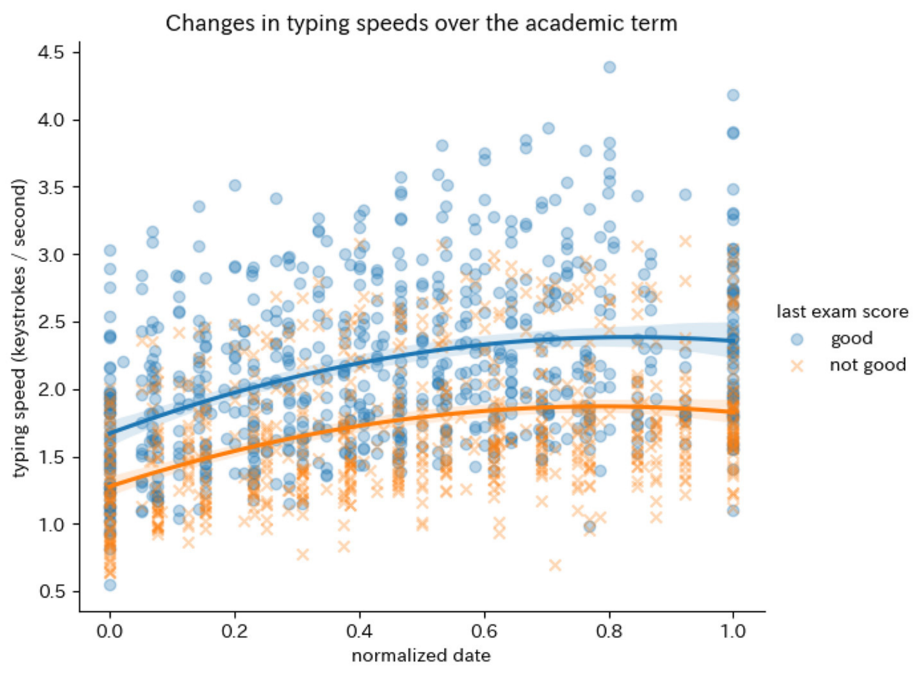


FIGURE 7
Changes in typing speeds over the academic term.

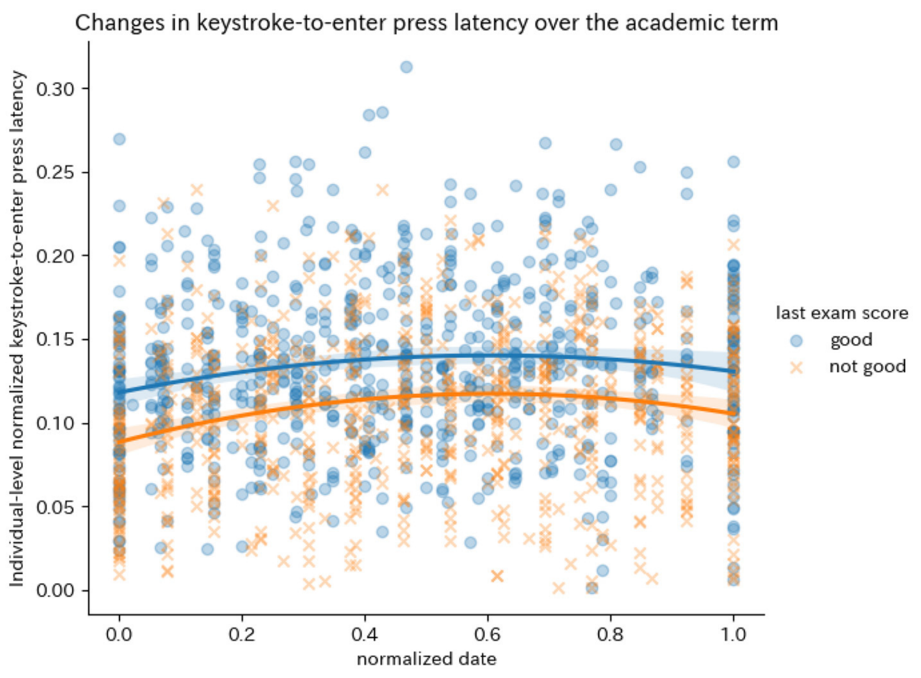


FIGURE 8
Changes in keystroke-to-enter press latency over the academic term.

data revealed a significant correlation between understanding of programming concepts and typing proficiency, indicating that individuals with a deeper comprehension of program functionality tend to exhibit faster typing speeds for fixed programming codes. However, it is noteworthy that a considerable portion of participants demonstrated high typing speeds without a corresponding understanding of the program, and vice versa, highlighting the complexity of this relationship. To address this challenge, a machine learning model leveraging typing rhythm was developed, achieving an impressive accuracy rate of approximately 83.0% in distinguishing individuals based on their comprehension of the program. Importantly, the characteristics identified by

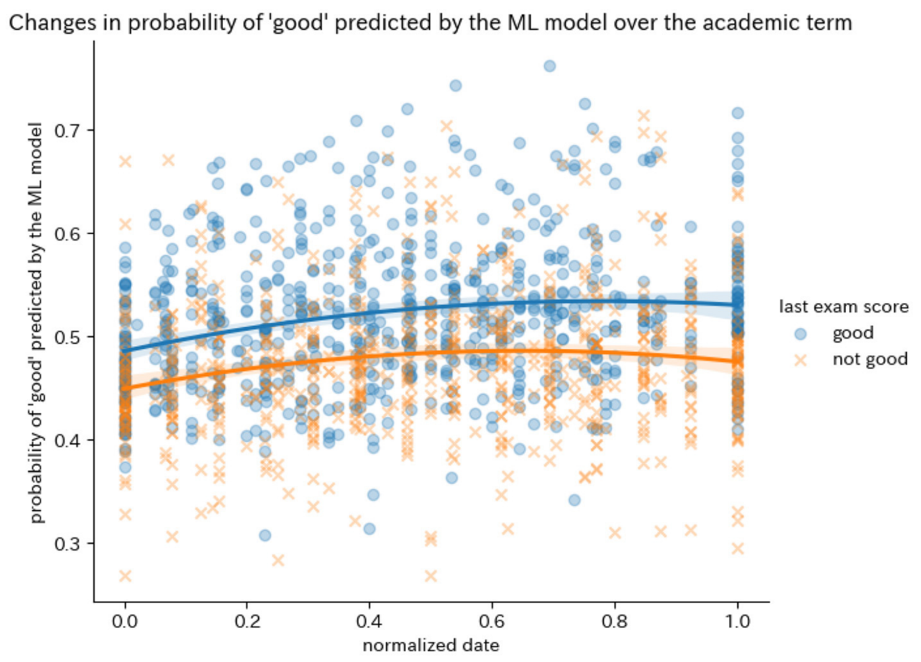


FIGURE 9
Changes in probability of good predicted by the ML model over the academic term.

this model emerged early in the course, suggesting the potential presence of innate aptitude rather than solely reflecting student progression. These findings underscore the multifaceted nature of programming aptitude and emphasize the importance of further research in understanding and harnessing individual differences in programming proficiency.

Data availability statement

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

Ethics statement

Ethical review and approval was not required for the study on human participants in accordance with the local legislation and institutional requirements. Written informed consent from the patients/ participants OR patients/participants legal guardian/next of kin was not required to participate in this study in accordance with the national legislation and the institutional requirements.

Author contributions

TN: Conceptualization, Data curation, Formal analysis, Methodology, Project administration, Resources, Software, Supervision, Visualization, Writing – original draft. MM: Funding acquisition, Resources, Validation, Writing – review & editing.

Funding

The author(s) declare financial support was received for the research, authorship, and/or publication of this article. This work was supported by JSPS KAKENHI Grant Number JP22K12319.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

References

Alfalahi, H., Khandoker, A. H., Chowdhury, N., Iakovakis, D., Dias, S. B., Chaudhuri, K. R., et al. (2022). Diagnostic accuracy of keystroke dynamics as digital biomarkers for fine motor decline in neuropsychiatric disorders: a systematic review and meta-analysis. *Sci. Rep*. 12:7690. doi: 10.1038/s41598-022-11865-7

Burkart, N., and Huber, M. F. (2020). A survey on the explainability of supervised machine learning. *arXiv, abs/2011.07876*.

Card, S., Moran, T., and Newell, A. (1980). The keystroke-level model for user performance time with interactive systems. *Commun. ACM* 23, 396–410. doi: 10.1145/358886.358895

Cattell, R. B. (1943). The measurement of adult intelligence. *Psychol. Bull.* 40, 153–193. doi: 10.1037/h0059973

Cheah, C. S. (2020). Factors contributing to the difficulties in teaching and learning of computer programming: a literature review. *Contemp. Educ. Technol.* 12:ep272. doi: 10.30935/cedtech/8247

Choi, W., Lam, C., and Mendes, A. (2023). "A systematic literature review on performance prediction in learning programming using educational data mining," in *2023 IEEE Frontiers in Education Conference (FIE), pages 1-9, Los Alamitos, CA, USA* (IEEE Computer Society). doi: 10.1109/FIE58773.2023.10343346

Cochrane, A., Simmering, V., and Green, C. S. (2019). Fluid intelligence is related to capacity in memory as well as attention: Evidence from middle childhood and adulthood. *PLoS ONE* 14:e0221353. doi: 10.1371/journal.pone.0221353

Conijn, R., Roeser, J., and van Zaanen, M. (2019). Understanding the keystroke log: the effect of writing task on keystroke features. *Read. Writ.* 32, 2353–2374. doi: 10.1007/s11145-019-09953-8

Edwards, J., Leinonen, J., and Hellas, A. (2020). "A study of keystroke data in two contexts: written language and programming language influence predictability of learning outcomes," in *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, 413–419. doi: 10.1145/3328778.3366863

Gedikli, A. M., and Efe, M., Ö. (2019). "A simple authentication method with multilayer feedforward neural network using keystroke dynamics," in *Mediterranean Conference on Pattern Recognition and Artificial Intelligence*. doi: 10.1007/978-3-030-37548-5_2

Goldberg, L. R. (1992). The development of markers for the big-five factor structure. *Psychol. Assess*. 4:26. doi: 10.1037//1040-3590.4.1.26

Holmes, A. A., Tripathi, S., Katz, E., Mondesire-Crump, I., Mahajan, R., Ritter, A., et al. (2022). A novel framework to estimate cognitive impairment via finger interaction with digital devices. *Brain Commun.* 4:fcac194. doi: 10.1093/braincomms/fcac194

Hommel, B., Müsseler, J., Aschersleben, G., and Prinz, W. (2001). The theory of event coding (tec): A framework for perception and action planning. *Behav. Brain Sci.* 24, 849–878. doi: 10.1017/S0140525X01000103

Hossain, M. N., Uddin, M. H., Thapa, K., Al Zubaer, M. A., Islam, M. S., Lee, J., et al. (2021). Detecting cognitive impairment status using keystroke patterns and physical activity data among the older adults: a machine learning approach. *J. Healthc. Eng.* 2021:1302989. doi: 10.1155/2021/1302989

Kadar, R., Wahab, N. A., Othman, J., Shamsuddin, M., and Mahlan, S. B. (2021). A study of difficulties in teaching and learning programming: a systematic literature review. *Int. J. Acad. Res. Progr. Educ. Dev*. 10, 591–605. doi: 10.6007/IJARPED/v10-i3/11100

Kievit, R. A., Davis, S. W., Griffiths, J., Correia, M. M., Cam,-C. A. N., and Henson, R. N. (2016). A watershed model of individual differences in fluid intelligence. *Neuropsychologia* 91, 186–198. doi: 10.1016/j.neuropsychologia.2016.08.008

Kovačević, N., Holz, C., Günther, T., Gross, M., and Wampfler, R. (2023). Personality trait recognition based on smartphone typing characteristics in the wild. *IEEE Trans. Affect. Comput.* 14, 3207–3217. doi: 10.1109/TAFFC.2023.3253202

Lindemann, O., Abolafia, J. M., Girardi, G., and Bekkering, H. (2007). Getting a grip on numbers: Numerical magnitude priming in object grasping. *J. Exper. Psychol.* 33, 1400–1409. doi: 10.1037/0096-1523.33.6.1400

McGlashan, H. L., Blanchard, C. C., Sycamore, N. J., Lee, R., French, B., and Holmes, N. P. (2017). Improvement in children's fine motor skills following a computerized typing intervention. *Hum. Mov. Sci.* 56, 29–36. doi: 10.1016/j.humov.2017.10.013

Nakada, T., and Miura, M. (2023). "Correlation analysis between keystroke and code understanding in programming language typing game," in *Proceedings of 2023 Summer Conference, Digital Games Research Association JAPAN*, 29–34.

Pinet, S. (2024). "What are you looking at? Beyond typing speed and formal training for assessing typing expertise," in *Proceedings of the Annual Meeting of the Cognitive Science Society*.

Pinet, S., Zielinski, C., Alario, F.-X., and Longcamp, M. (2022). Typing expertise in a large student population. *Cogn. Res.* 7:77. doi: 10.1186/s41235-022-00424-3

Prat, C. S., Madhyastha, T. M., Mottarella, M. J., and Kuo, C.-H. (2020). Relating natural language aptitude to individual differences in learning programming languages. *Sci. Rep*. 10:3817. doi: 10.1038/s41598-020-60661-8

Salthouse, T. (1984). Effects of age and skill in typing. *J. Exper. Psychol. General* 113, 345–371. doi: 10.1037/0096-3445.113.3.345

Salthouse, T. A. (1986). Perceptual, cognitive, and motoric aspects of transcription typing. *Psychol. Bull.* 99:303. doi: 10.1037//0033-2909.99.3.303

Shanmugapriya, D., and Padmavathi, G. (2009). A survey of biometric keystroke dynamics: approaches, security and challenges. *arXiv preprint arXiv:0910.0817*.

Shrestha, R., Leinonen, J., Zavgorodniaia, A., Hellas, A., and Edwards, J. (2022). "Pausing while programming: Insights from keystroke analysis," in *2022 IEEE/ACM 44th International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*, 187–198. doi: 10.1109/ICSE-SEET55299.2022.9794163

Tachibana, R., and Komachi, M. (2016). "Analysis of english spelling errors in a word-typing game," in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, 385–390.

Thomas, R. C., Karahasanovic, A., and Kennedy, G. E. (2005). "An investigation into keystroke latency metrics as an indicator of programming performance," in *Proceedings of the 7th Australasian Conference on Computing Education - Volume 42, ACE'05* (AUS. Australian Computer Society, Inc.), 127–134.

Wang, Y., Lin, T., Yu, J., Wang, L., He, J., and Ke, L. (2021). "Current status of user experience of the keyboard on smartphones: An overall questionnaire analysis," in *HCI International 2021 - Late Breaking Papers: Multimodality, eXtended Reality, and Artificial Intelligence*, eds. C. Stephanidis, M. Kurosu, J. Y. C. Chen, G. Fragomeni, N. Streitz, S. Konomi, et al. (Cham: Springer International Publishing), 168–182. doi: 10.1007/978-3-030-90963-5_14

Wetherell, M. A., Lau, S.-H., and Maxion, R. A. (2023). The effect of socially evaluated multitasking stress on typing rhythms. *Psychophysiology* 60:e14293. doi: 10.1111/psyp.14293

Yang, L., and Qin, S.-F. (2021). A review of emotion recognition methods from keystroke, mouse, and touchscreen dynamics. *IEEE Access* 9, 162197–162213. doi: 10.1109/ACCESS.2021.3132233

Zhao, H., Li, M., Lin, T., Wang, R., and Wu, Z. (2021). Prolog2vec: Detecting novices' difficulty in programming using deep learning. *IEEE Access* 9, 53243–53254. doi: 10.1109/ACCESS.2021.3067505