Identification Based on Typing Patterns Between Programming and Free Text

Petrus Peltola, Vilma Kangas, Nea Pirttinen, Henrik Nygren, Juho Leinonen
University of Helsinki
Helsinki, Finland
{petrus.peltola,vilma.l.kangas,nea.pirttinen,henrik.nygren,juho.leinonen}@helsinki.fi

ABSTRACT

Identifying people based on their typing has been studied successfully in multiple different contexts. Previous research has shown that identification is possible based on writing predetermined texts such as typing passwords, free text such as essays, as well based on writing source code. In this work, we study typing pattern based identification when the text format and writing environment change. We replicate two earlier studies which suggested that typing profile identification works with programming data, and that it can be applied to a programming exam circumstances with decent results. Then, we examine how the identification accuracy changes when the user profiles are built using data from programming, and the identification is conducted on data from writing free text. Our results show that the identification accuracy is indeed high within the context of programming data, but drops when identifying essay typists based on typing profiles built from their programming data.

CCS CONCEPTS

• **Security and privacy** → *Biometrics*; *Pseudonymity, anonymity and untraceability*; • **Social and professional topics** → *Computer science education*;

KEYWORDS

keystroke dynamics, cross-context identification, replication, source code snapshots, educational data mining

1 INTRODUCTION

It has been suggested as early as in the 1970s that computer users could be identified by how they use the keyboard [14]. A number of methods for identifying computer users based on their typing patterns have been proposed [5, 12] since. These methods can identify users rather accurately [1–3, 16], but they center mostly on freely typed texts such as writing essays. Additionally, there have been studies on how keystroke based identification performs in the context of programming [7, 8]. The results of those studies show a reasonable accuracy, although identification has been slightly less accurate than in studies within the context of freely typed texts.

With the promising results in multiple different contexts, the question of identifying people from one context to another emerges as an open problem. The importance of identifying users between multiple different contexts becomes apparent when the type of text that the user writes varies greatly. For example, the bulk of a programming course's workload usually consists of programming exercises, but usually at least the exam contains some sort of explanation assignments or other texts written in a natural language. Thus, in order to fully get the advantage of using typing patterns as an authentication method on e.g. MOOCs, it would be beneficial to be able to identify students from one context to another.

This article focuses on the question of whether it is possible to identify users between different contexts. Users' typing patterns are likely to be different depending on whether they are writing free text or programming. They might also differ in the contexts of completing programming exercises at home, completing them in an exam, and writing an essay in an exam in a controlled environment.

This article is organized as follows. First in Section 2 we provide an overview of previous work done on keystroke based identification using free text and programming data. Then in Section 3 we discuss our data and research methodology. In Section 4 we provide our results, which are then examined in Section 5 along with the limitations of our study. Section 6 concludes the article and outlines possible future work on the subject.

2 RELATED WORK

Information recorded from typing has mainly been used for identification and authentication purposes [1, 5, 9, 11, 12]. This information includes for example the duration of keystrokes, pressure of keystrokes, and keystroke latencies. Identifying users via such measures provides more security for systems that traditionally use only usernames and passwords for authentication. They can also monitor the authenticated user and detect if the user changes. It could be said that if username and password combinations are the bouncer at the door of an establishment serving alcoholic beverages, then the continuous examining of keystrokes is the bartender who refuses to serve to those who are too inebriated. Keystroke analysis can also be used in online exams as a way of catching potential plagiarists and cheaters [4].

Digraph latencies are one of the most common features derived from keystroke data. A digraph is a pair of adjacent characters, for example, the word "text" has three digraphs in it: "te", "ex", and "xt". It has been shown that users can be accurately identified using average digraph latencies [1, 2, 11, 16]. Calculating holding times

Koli Calling 2017, November 16–19, 2017, Koli, Finland

ACM ISBN 978-1-4503-5301-4/17/11…$15.00
https://doi.org/10.1145/3141880.3141903

of keystrokes, average keystrokes per minute and number of errors have also been considered for identification purposes [13].

Behavioral features such as typing patterns can vary depending on the context. For example, different types of text have been shown to have different identification accuracies [11]. Especially the difference between transcribed or predetermined text, such as the username and password, and free text, such as essays, has been studied. In a study by Monrose and Rubin, identifying users with transcribed text reached an accuracy of 79% whereas using free text the accuracy decreased significantly to 21% [11]. A possible explanation of how the context of the experiment can affect the result, Monrose and Rubin speculate that the decrease might be caused by the writer having to spend time on coming up with what to write during essays.

A study by Longi et al. [8] shows that the identity of programmers can be detected from keystroke data recorded during programming sessions. They used data from two separate programming courses and noted that linking the students from one course to another could be done with an accuracy of 98.6%. This indicates that keystroke identification could be a considerable way of authentication in, for example, MOOCs.

Keystroke based identification has been successfully used for students participating in online exams [7, 15]. Leinonen et al. [7] were able to identify a large portion of the students who took a programming exam. Their results showed that students can be identified quite accurately in both controlled and uncontrolled exam environments. In the controlled exam, the students were in a computer lab at the university, whereas in the uncontrolled exam they could choose where they wanted to take the exam.

Typing patterns can also differ when using different keyboards and changing other factors in the writing environment, such as exam setting. In their study, Villani et al. [16] used both freely typed and transcribed text. They could identify 98% of the users who used only one type of keyboard, but when they were identifying users from one setup to another, i.e. when the users had different keyboards in their training and test sets, the identification accuracy fell to around 60%. When users stayed on the same computer but switched between the different types of texts, the identification accuracy decreased by 10%-20%. Thus their results suggest that changing the writing environment affects the accuracy more than changing the assignment.

3 METHODOLOGY

3.1 Context

The data for this study comes from a CS1 Java programming course conducted as a massive open online course (MOOC) that was organized by University of Helsinki in the spring of 2017. The course first covered the basics of programming and then delved into object-oriented programming. The course was targeted at beginners, and therefore many of the participants had not programmed at all before the course. The course material consisted of written theory sections which each covered a certain topic and exercises which were interspersed with the theory sections. The material was split into fourteen separate weeks which all had different themes, exercises and deadlines.

The course included a final exam that was held at university premises with a basic computer lab setup. The exam was compulsory for those who wished to earn credits for the course. In the three-hour exam, the first 30 minutes were reserved for three essay questions, and for the last two and a half hours the participants were able to complete programming assignment questions or revise their essay answers. Majority of the participants were native in Finnish, the language the course was held in.

Since the course suffers from a high drop-out rate that generally affects most MOOCs, only the people who completed both the course and the final exam at the university premises were included in this study. The data sets used in this study had 113 students after preprocessing, except for the last two weeks' set that had 112 students.

3.2 Research questions

- RQ1. How does the identification accuracy vary between contexts, i.e. when identifying people in..
  - RQ1.1. ..the last two weeks?
  - RQ1.2. ..the programming exam?
  - RQ1.3. ..the essay questions?
- RQ2. Can you identify students writing an essay based on programming exam data and vice versa?

To have a large set of data that we can use to build reliable typing profiles from, we use the programming data of the first 12 weeks of assignments. This data set is used as a training set for RQ1.

To answer Research Question 1.1, a study by Longi et al. [8] is replicated. We call these replicated results our "control set" for the rest of the article. The study determines how identification accuracy varies between data sets in a relaxed, at-home environment.

For Research Question 1.2, we replicate a study by Leinonen et al. [7] to determine the effects to identification accuracy when identifying people in a supervised exam programming environment based on data from a relaxed at-home programming environment. We will refer to this set of results as the "exam set".

With Research Question 1.3, we examine whether people writing a free text essay assignment in a supervised exam environment can be identified based on data from a relaxed at-home programming environment. We will refer to the essay assignment data set as the "essay set".

For Research Question 2, the identification accuracy in an exam environment between different textual contexts, i.e. free text and programming, is investigated. Questions 1.3 and 2, both of which concern identifying people between different textual contexts, are previously unexplored and thus form a novel contribution to the field of keystroke-based identification.

3.3 Data collection

The programming assignment related data was collected during the course through a custom IDE plugin. By installing the plugin to their programming environment, the students gave permission both to gathering and using their data. The plugin was not required to be able to partake in the course, but it was obligatory if the student wished to earn credits for completing the course.

The raw data is collected at the keystroke level and includes information such as a student identifier, an assignment identifier, a

timestamp, the type of interaction, and any visible change to the source code in the IDE. Though the interactions included information such as when the student changes from the IDE to another window or returns to IDE, only text inserts and text removals were relevant for our study.

Data was also collected during the final exam. The data from the programming exercises was collected with the same plugin that was used during the course, while the essay data was gathered with a JavaScript plugin. All the essays were written into a text area in the webpage that contained a monitoring script which captured all the changes to the text at the keystroke level.

3.4 Typing profiles

The raw data was converted into typing profiles that include the users' average digraph latencies for the 25 most common digraphs of the test set. We were interested specifically in average digraph latencies, i.e. how long it takes the user to go from one key to another on average. Digraphs were chosen as the form of identification due to the results of a study done by Longi et al. [8]. The study concluded that digraph latencies were distinctly the most accurate way of identification when compared to the average time the programmer needs to press any key while typing or to the average time for each key individually.

The raw data is a full snapshot of the programming environment at the moment of its timestamp. Thus, it contains unnecessary information that was excluded for not being relevant to our study. For the analysis, all the events that consisted of more than a single keypress were removed. These events can be caused, for example, by the programming environment's autocomplete feature that matches quotes and parentheses. Copy-pasting also shows up as an event with multiple keypresses. We also excluded two or more text remove events consecutively, since that usually means that the writer has simply held the backspace button down, and has not actually made separate keypresses.

Participants were mapped so that for each participant there was a map of digraph IDs, their total counts – the number of times the digraph was written – and sums – how long it has taken to write the digraph in total. We excluded event time differences that were either below 10 ms or above 750 ms from the collected results, in order to not include pauses or single character auto inserts by the IDE in the typing profiles. To get more accurate results, we only took into account the 25 most common digraphs from each test set, as previous research has shown that 25 digraphs are enough for identification and reduce the chance of overfitting to the data [6, 7].

These most common digraphs were then compared to those in the training data set. The same digraphs were searched from the set, the total counts and sums were calculated for each participating user and the results were normalized to a range from 0 to 1. However, the essay set's 25 most common digraphs differ a lot from the exam set's most common digraphs due to very different text contexts. We recognized that this could have possibly distorted the results. To compensate for that we also decided to study identification accuracies using the 25 most used digraphs from the intersection of the exam and essay sets. For the calculation, we first determined which data set contained less usages for each mutual digraph, and used that usage count as that digraph's count when selecting the 25 most used mutual digraphs. This was done so that the 25 most common mutual digraphs would not include digraphs that were used several times in one data set and only a few times in the other. We will refer to these digraphs as the "mutual digraphs", which are discussed further in Section 4.

3.5 Data analysis

To identify the students across the different datasets, the Euclidean distance between the students' typing profiles was calculated similar to previous studies by Longi et al. [8] and Leinonen et al. [7].

A typing profile vector has 25 values – the average time it takes for the typist to type each of the 25 digraphs. For each typing profile in the test set, we iterated over all the profiles in the training set and calculated the distances between the two profiles, leaving us with an ordered list of the closest profiles in the training set for each test set profile.

Then, to get data about how correctly the student was identified, we calculated how close their training set profile was to their test set profile. We iterated over every test set user's list of closest training set profiles and looked at the index of the user's profile in the training set. The index denotes the relative distance between the user's two profiles. For example, a user's distance could be 3, which would mean that their training profile was the 3rd closest to their test profile.

To measure the identification accuracy we defined three different acceptance thresholds similar to Longi et al. [8] for how close the user's training profile is to their test profile: 1, 5 and 10. The thresholds indicate that the training profile is in the closest k estimates. For example, the threshold of one indicates exact matches, i.e. the subject's training set profile was the closest match of all the people in the training set. Similarly, the threshold of five indicates that the user is in the five closest training set profiles.

4 RESULTS

The most common digraphs from programming context include for example *i -> n* and *n -> t*, from writing *int*, and *{ -> SPACE*. These are much more typical for a programming language than for a natural language.

The essays were written in Finnish, a language with vast linguistic differences to English, so the digraphs are bound to be different. The essay set was the smallest, with total digraph usage counts ranging from about 13 000 to 3000 in the 25 most common digraphs, while the total usage counts in the exam set range from 15 000 to just over 3000, and in control set from 25 000 to 10 000. The peculiarity of a natural language compared to a programming language can also be seen in the number of digraphs containing keypresses to or from the spacebar. Out of 25, six digraphs contain this event in the essay set, while in the exam set, *SPACE* only appears once.

The mutual digraphs show features from both the essay and exam sets. For example, *, -> SPACE* is a digraph that is very prominent in natural languages, but not so much in programming. On the other hand, the digraph *i -> n* that is common in source code is included in the mutual most common digraphs too. The mutual digraphs are somewhat biased to the exam set, as some digraphs that are common in the essay set are rarely found in the exam set, which results in them being excluded.

Table 1: Identification accuracies for the data sets with varying thresholds. Students in the test set were identified based on typing profiles built from the training set. There were 113 students in all the data sets except the 2 weeks set that had 112.

Index	Train set	Test set	Features	Threshold 1	Threshold 5	Threshold 10
1	**12 weeks**	**2 weeks**	2 weeks top 25 digraphs	96%	100%	100%
2	**12 weeks**	**Exam**	Exam top 25 digraphs	73%	92%	95%
3	**12 weeks**	**Essay**	Essay top 25 digraphs	50%	83%	89%
4	**Exam**	**Essay**	Essay top 25 digraphs	25%	54%	68%
5	**Essay**	**Exam**	Exam top 25 digraphs	19%	47%	62%
6	**Exam**	**Essay**	Mutual top 25 digraphs	27%	56%	68%
7	**Essay**	**Exam**	Mutual top 25 digraphs	27%	63%	76%

To answer Research Question 1.1 we calculated the identification accuracy when identifying between the training set and the control set, for which the results are shown in Table 1, row 1. The identification of students between the training and the control set is very accurate, with about 96% of users being exact matches. The identification is in around 98% accuracy when examining the two closest users, leaving only two students out of 112 unidentified. The number jumps up to a perfect 100% from the four closest users and beyond. The identification accuracy is generally in line with the previous findings on the subject [7, 8].

For Research Question 1.2 the programming exam set was used instead of the control one, for which the results are shown in Table 1, row 2. Identifying students between the training and the exam set, a drop in the accuracy compared to the control set is observed, which was predictable and corresponds to the findings of Leinonen et al. [7]. About 73% of the users were exact matches, around 92% belonged in the five closest profiles and nearly 95% were found in the top 10.

Calculations were done similarly for Research Question 1.3, now using the essay set as the test data. The results are reported in Table 1, row 3. While the accuracy noticeably drops when moving from assignments to the exam, the fall is even greater with the essay data. The percentage of exact matches fell to about 50%, top 5 matches dropped to 83% and top 10 matches to about 89% accuracy.

For Research Question 2, we used the same methods to calculate accuracy as in the previous sections. First, the essay set was used as a test set and the exam set as a training set, then vice versa. The results of the first experiment are in Table 1, row 4. The identification accuracy drops drastically compared to any other combination of sets, with only about 25% being exact matches. Almost 54% of the students were identified with a threshold of 5, and about 68% with a threshold of 10.

The results of the second experiment, with the exam set as a test set and the essay set as a training set, are in Table 1, row 5. The accuracy is still better than random guessing, though it has fallen to a fraction of the control set accuracy. Around 19% are exact matches, nearly 47% belong to the five closest profiles and almost 62% to ten closest profiles.

When using the essay data as a test set and the exam data as a training set for mutual digraphs (Table 1, row 6), approximately 27% of the people were identified with threshold 1, about 56% with threshold 5 and about 68% with threshold 10. The accuracy increased also when using the exam data as a test set and the essay data as a training set (Table 1, row 7), compared to when using the most common digraphs of the exam data instead of mutual digraphs. Now the accuracy was around 27% with a threshold of 1, nearly 63% with a threshold of 5 and approximately 76% with a threshold of 10.

5 DISCUSSION

As has been concluded in previous studies [7, 8], identifying students within the programming data collected on a course is very accurate. Since all the exercises are programming tasks and likely completed within a single context, the resulting typing pattern data is consistent throughout the whole course. Therefore identification is feasible, unless there is a drastic change in the user's setup or ability to type. Our results aligned with this hypothesis, with our identification accuracy being even higher than that of previous studies, possibly due to an increased amount of data.

However, identification gets more difficult when the setup and conditions change, as seen in the results of Leinonen et al. [7]. This can be due to various reasons, for example the stress of the exam situation, a different keyboard or other equipment, and the absence of the course material [3]. In reality, probably all of the mentioned factors affect the accuracy, along with many things we have not considered in this study. To actually study just the effect of a single element changing, e.g. switching a user's keyboard, further study needs to be done. Our results were still considerably good, especially when we raise the identification threshold from exact matches to five or ten closest matches.

As expected, typing pattern based identification is less accurate when the data for the training and the test set come from different contexts, or more specifically, when identifying someone writing a free text essay based on their typing profile built from programming assignment data. Our results indicate that especially the accuracy of exactly identifying someone decreases. As the essay was written in an exam, the same reasons that we hypothesize reduce identification accuracy in an exam scenario apply. Additionally, the essays are written in Finnish, while Java's keywords are in English, or abbreviations resembling English. Thus, the results would possibly be better if the essays were written in English, as the digraphs are most likely very different between languages. The most common digraphs for Java include e.g. the digraph *{ -> SPACE*. Thus, at least some of the reduction in accuracy would remain even if the essays were written in English. The Finnish language contains some characters that are not found in English. The course material instructs

not to use these letters in variable or method names, so even if students use variable or method names in their native language, the special letters are rarely present in any of the programming data. These letters occur consistently in the essay data, some even in the 25 most common digraphs. This is also one of the reasons we decided to calculate the mutual digraphs for the essay and exam sets, as their most common digraphs separately are quite different. The special letters get excluded from the mutual digraphs as their occurrence in the exam data is very rare.

When comparing essay data to exam data and vice versa, the environment is exactly the same, but the text content is very different. It is likely that the digraphs that are common in free text are barely found in programming data, and thus, identification is harder. Also, the data sets are relatively small, at least compared to the full twelve weeks of the course, so the count of total occurrences of any digraph is much smaller. Even so, the results are noticeably better than completely random matching.

Previous research has excluded users with low event counts [8]. Our data sets were already quite small and the event counts were reasonable on their lowest so we decided against this, but with larger datasets it would undoubtedly increase accuracy. The number of digraphs studied could be adjusted according to the data being analyzed, since the limit of 25 most common digraphs is an arbitrary number that has had success in previous studies done on the same kind and amount of data we have in our study. In this study we chose the digraphs with the highest total counts, which could result in large disparities between the usages of particular digraphs between users. For example, in a hypothetical scenario where a single student types an obscure digraph for a number of times, it could be included in the most common digraphs. Previous work on the topic has avoided this issue by using the median number of times students use a particular digraph instead of using the total count [7]. However, our results are in line with those studies, and thus this does not seem to be an issue with the data sets used in this study.

Achieving high identification accuracies also poses risks as it means that the data used to build typing profiles contains personal, identifiable information. For example, if the data sets used in this study were released openly, someone who had similar data could possibly identify people in our data based on their data. This could be used e.g. for targeted advertising – for example, the context of our data reveals that the users have attended a MOOC, which could be used in personalizing advertisements. Recently, approaches to prevent identification have been proposed [6, 10] to alleviate this issue, but more research on the topic is needed.

6 CONCLUSIONS

In this study, we successfully replicated two studies concerning identifying programmers from their typing patterns and identified students across different textual contexts, from free text to programming and vice versa. We conclude that even if cross-context identification is less accurate than within one context, it still is possible. This indicates that typing pattern based identification could be a reliable authentication method even on courses where the content of the assignments differs: for example, in programming courses with both coding assignments as well as essays.

For future work, we are interested in studying the limits of typing pattern based authentication with a larger data set. For example, relying on a fixed threshold will not work if the number of students is increased to thousands. Additionally, we are interested in examining how different machine learning methods could be used for typing pattern based identification in addition to the methods used in this study. Future work should also alter the different environment and assignment elements more exclusively to get a better understanding of just how much a change of environment or going from one text content to another affects the identification accuracy.

$$d(p,q) = \sqrt{\sum_{i=1}^{25}(p_i - q_i)^2}$$

REFERENCES

[1] Paul S. Dowland and Steven M. Furnell. 2004. *A Long-Term Trial of Keystroke Profiling Using Digraph, Trigraph and Keyword Latencies.* Springer US, Boston, MA, USA, 275–289.

[2] R. S. Gaines, W. Lisowski, S. J. Press, and N. Shapiro. 1980. *Authentication by keystroke timing: Some preliminary results.* Technical Report.

[3] Daniele Gunetti and Claudia Picardi. 2005. Keystroke Analysis of Free Text. *ACM Trans. Inf. Syst. Secur.* 8, 3 (Aug. 2005), 312–347.

[4] Arto Hellas, Juho Leinonen, and Petri Ihantola. 2017. Plagiarism in Take-home Exams: Help-seeking, Collaboration, and Systematic Cheating. In *Proc. of the 2017 ACM Conf. on Innovation and Technology in Computer Science Education (ITiCSE '17).* ACM, New York, NY, USA, 238–243.

[5] M. Karnan, M. Akila, and N. Krishnaraj. 2011. Biometric Personal Authentication Using Keystroke Dynamics: A Review. *Appl. Soft Comput.* 11, 2 (March 2011), 1565–1573.

[6] Juho Leinonen, Petri Ihantola, and Arto Hellas. 2017. Preventing Keystroke Based Identification in Open Data Sets. In *Proc. of the Fourth (2017) ACM Conf. on Learning @ Scale (L@S '17).* ACM, New York, NY, USA, 101–109.

[7] Juho Leinonen, Krista Longi, Arto Klami, Alireza Ahadi, and Arto Vihavainen. 2016. Typing Patterns and Authentication in Practical Programming Exams. In *Proc. of the 2016 ACM Conf. on Innovation and Technology in Computer Science Education (ITiCSE '16).* ACM, New York, NY, USA, 160–165.

[8] Krista Longi, Juho Leinonen, Henrik Nygren, Joni Salmi, Arto Klami, and Arto Vihavainen. 2015. Identification of Programmers from Typing Patterns. In *Proc. of the 15th Koli Calling Conf. on Computing Education Research (Koli Calling '15).* ACM, New York, NY, USA, 60–67.

[9] J Monaco, J Stewart, S.-H Cha, and C Tappert. 2013. Behavioral biometric verification of student identity in online course assessment and authentication of authors in literary works in 2013. (01 2013), 1–8.

[10] John V Monaco and Charles C Tappert. 2016. Obfuscating Keystroke Time Intervals to Avoid Identification and Impersonation. *arXiv preprint arXiv:1609.07612* (2016).

[11] Fabian Monrose and Aviel Rubin. 1997. Authentication via Keystroke Dynamics. In *Proc. of the 4th ACM Conf. on Computer and Communications Security (CCS '97).* ACM, New York, NY, USA, 48–56.

[12] Alen Peacock, Xian Ke, and Matthew Wilkerson. 2004. Typing Patterns: A Key to User Identification. *IEEE Security and Privacy* 2, 5 (Sept. 2004), 40–47.

[13] Mariusz Rybnik, Marek Tabedzki, and Khalid Saeed. 2008. A Keystroke Dynamics Based System for User Identification. In *Proc. of the 2008 7th Computer Information Systems and Industrial Management Applications (CISIM '08).* IEEE Computer Society, Washington, DC, USA, 225–230.

[14] R. Spillane. 1975. Keyboard apparatus for personal identification. *IBM Technical Disclosure Bulletin* 17, 3346 (1975).

[15] John C. Stewart, John V. Monaco, Sung-Hyuk Cha, and Charles C. Tappert. 2011. An Investigation of Keystroke and Stylometry Traits for Authenticating Online Test Takers. In *Proc. of the 2011 International Joint Conf. on Biometrics (IJCB '11).* IEEE Computer Society, Washington, DC, USA, 1–7.

[16] Mary Villani, Charles Tappert, Giang Ngo, Justin Simone, Huguens St. Fort, and Sung-Hyuk Cha. 2006. Keystroke Biometric Recognition Studies on Long-Text Input Under Ideal and Application-Oriented Conditions. In *Proc. of the 2006 Conf. on Computer Vision and Pattern Recognition Workshop (CVPRW '06).* IEEE Computer Society, Washington, DC, USA, 39–.