Typing Patterns and Authentication in Practical Programming Exams

Juho Leinonen
University of Helsinki
Finland
juho.leinonen@helsinki.fi

Krista Longi
University of Helsinki
Finland
krista.longi@helsinki.fi

Arto Klami
University of Helsinki
Finland
aklami@cs.helsinki.fi

Alireza Ahadi
University of Technology
Sydney
Australia
alireza.ahadi@uts.edu.au

Arto Vihavainen
University of Helsinki
Finland
avihavai@cs.helsinki.fi

ABSTRACT

In traditional programming courses, students have usually been at least partly graded using pen and paper exams. One of the problems related to such exams is that they only partially connect to the practice conducted within such courses. Testing students in a more practical environment has been constrained due to the limited resources that are needed, for example, for authentication.

In this work, we study whether students in a programming course can be identified in an exam setting based solely on their typing patterns. We replicate an earlier study that indicated that keystroke analysis can be used for identifying programmers. Then, we examine how a controlled machine examination setting affects the identification accuracy, i.e. if students can be identified reliably in a machine exam based on typing profiles built with data from students' programming assignments from a course. Finally, we investigate the identification accuracy in an uncontrolled machine exam, where students can complete the exam at any time using any computer they want.

Our results indicate that even though the identification accuracy deteriorates when identifying students in an exam, the accuracy is high enough to reliably identify students if the identification is not required to be exact, but top k closest matches are regarded as correct.

CCS Concepts

•Information systems → Data mining; •Social and professional topics → Computer science education; CS1; •Computing methodologies → Supervised learning by classification; •Security and privacy → *Biometrics;*

ITiCSE '16, July 09 - 13, 2016, Arequipa, Peru

© 2016 Copyright held by the owner/author(s). Publication rights licensed to ACM. ISBN 978-1-4503-4231-5/16/07...$15.00

DOI: http://dx.doi.org/10.1145/2899415.2899472

Keywords

authentication, machine exam, programmer identification, source code snapshots

1. INTRODUCTION

Hundreds of thousands of students around the world attend an introductory programming course each year, and at the end of the course, many of them take part in an exam. The exam may for example be a traditional written exam, where students answer questions using pen and paper, or, it may be a more practical laboratory exam, where students are expected to construct programs in a lab setting [2,20]. In both cases, students typically take the exams at their own educational institution, where their identity is determined by administrators or course staff.

With the trend towards blended and online course offerings, it is meaningful to consider taking at least parts of the exam by telecommute. Naturally, when such an option is considered, concerns related to e.g. plagiarism [5, 19] and authentication are involved. Plagiarism detection is about determining whether the student is the author of the proposed solution, while authentication is related to determining whether the student is who he or she poses to be.

Our work explores the possibility of automatically authenticating students taking an exam by telecommute using their typing patterns as the authentication mechanism. If authentication using such means is possible, it could lead to reducing costs related to the facilitation of local examinations, increase the flexibility in when the exam is given, and also provide the students with the chance to take the exam at a location where they feel comfortable, possibly helping them perform better during the exam.

While authentication of students based on typing speed and keystroke durations has been previously explored outside the programming domain [11, 18], the task was only recently considered within the domain of programming by Longi et al. [14]. They used programming course data to learn students' typing profiles, and then sought to identify the students in two separate situations: during a future session on the same course and during a future course. With the best approach, over 90% of the students in the dataset were correctly identified.

In this work, we (1) replicate the work previously reported by [14] with separate data sets, (2) explore to which extent the method can be applied to distinguish students with data from a shorter period, i.e. an exam, and (3) study how the accuracy of the method changes when students work using machines that they are accustomed to when compared to machines in a laboratory setting. The main goal of these experiments is to study the robustness of the approach in practical settings.

This article is organized as follows. We provide a brief overview of identification of users using keystroke data in Section 2, after which we outline our research methodology and data in Section 3. The description of results and experiments is given in Section 4, followed by a discussion of the results in Section 5. Finally, in Section 6, we conclude the article and outline future work.

2. RELATED WORK

Various characteristics can be calculated from typing data. These characteristics include duration of keystrokes, pressure of keystrokes, and keystroke latencies [11], which may vary depending on the situation – typing patterns can be affected by different keyboards, for example, or different types of texts [8]. Out of these characteristics the keystroke latencies, especially *digraph* latencies – the time it takes to move between a pair of keys – have been widely used [6,7,12,17,24]. Generally, digraphs are any two adjacent characters, for example, the word *word* includes three digraphs: *wo*, *or* and *rd*. If the word is mistyped, additional digraphs are also observed – it also is possible to include events such as deletions.

Much of the previous work on typing analysis has been based on transcribed or pre-determined text, such as usernames and passwords [3, 4, 9, 10, 25], but there are studies where the input has been free text instead [3,8,15]. The results vary significantly based on the data used. For example, in a study by Monrose and Rubin with 46 participants, the identification accuracy decreased significantly from 79% with transcribed text to 21% with free text [17]. They suggested that this result could be partially explained by the writer having to think about something to write rather than being just able to type in whatever is given.

There are positive results from using free text as well. For example, in an experiment with 20 participants, Killourhy and Maxion found no significant difference in classification results when using transcribed or free text [12]. In their experiment, the participants were given comparable transcription and free composition tasks, and they used lowercase digraph and key-hold times with two separate classification algorithms. Although the results were not exactly the same, they were very close and neither one was always better [12].

Typing patterns can be affected by different conditions such as the keyboard [24]. In a study by Villani et al. [24], where participants had either only freely typed or transcribed text, identification accuracies of over 98% were achieved when when the subjects only used one type of keyboard. However, when the keyboard was switched between the session that produced the data from which the typing profile was learned and the session where the participant was being identified, the identification accuracy dropped to about 60%. However, at the same time, using both desktop and laptop keyboards in both sessions did not noticeably decrease the accuracy [24].

Keystroke analysis has also been successfully applied in identifying students in online exams [16, 21]. Using data from 30 students taking examinations in a business school, Monaco et al. were able to correctly identify all the students [16]. Similarly, Coursera is collecting typing samples from students who want to acquire a verified certificate [1].

In a programming context, keystroke analysis has been used to infer programming performance [13, 22]. Thomas et al. [22] found that there exist some digraphs that have a moderate negative correlation with course scores. They argue that more knowledgeable programmers type certain digraphs faster than novice programmers.

3. METHODOLOGY

3.1 Context

The data for the experiments has been collected from four programming courses held at the University of Helsinki in 2015 spanning both spring and autumn semesters. Both semesters had a beginner and an advanced course in Java. All four courses had a machine examination. The machine exams consisted of Java programming assignments and the students were allowed to use the Internet, but not communicate with anyone. The machine exams in spring 2015 were conducted at university facilities in a computer lab, where students' identities were verified by course personnel. In autumn 2015, the students were allowed to complete the examination at any location using any computer they wished to use, and their identities were only verified by the system used to submit the solutions to the programming assignments. The students had two and a half hours to complete the exam in both the controlled and the uncontrolled machine exams.

All four courses lasted 7 weeks each. In the beginner courses, the students practice basic programming concepts such as variables, inputs and outputs, loops, and objects, and in the advanced courses, the students learn to use interfaces, inheritance, file handling, and user interfaces. The students complete multiple programming exercises every week. Keystroke data from the exercises is collected using the Test My Code -system, which provides an IDE integration for the purposes of data gathering as well as providing feedback to students as they work on the course assignments [23].

For each keystroke conducted within the environment that changes the code, timestamp, exercise information, course information, student information and the diff-information of the change is collected. As the system only records visible differences in the code, the data does not include special keys like the Shift- and Ctrl-keys. The students can also opt-out of data collection during the weekly exercises, but not in the machine exam.

3.2 Research Questions

Our research questions for this study are as follows:

RQ 1. How does the identification accuracy vary between data sets?

RQ 2. How does a controlled machine examination situation affect the identification accuracy?

RQ 3. How does an uncontrolled machine examination situation affect the identification accuracy?

To answer the first research question, we replicate the study by Longi et al. [14] to determine how identification accuracy varies between data sets. To answer the second and third research questions, we study how the exam situation affects identification accuracy using two different experiments.

In the first experiment, the students complete a mandatory machine examination at university facilities in a computer lab and their identity is verified. In the second experiment, the students were allowed to complete a similar mandatory machine examination at any location using any computer they wish to use and their identity is not verified.

3.3 Preprocessing

In the first experiment where we replicate the study by Longi et al. [14], we used the same filters as outlined in the original study to achieve as reliable replication as possible in Section 4.1. This means that students who typed less than 2000 characters during the first week of the course were not included in the experiments. For the machine exam experiments we only included data from the students who participated in the machine exam.

Following the process described in [14], we filtered out keystroke events that added more than one character to eliminate refactoring and copy-paste events. We used the same limitation on the typing event time; only typing events where the duration from the previous event was in the range of 10ms – 750ms were included.

3.4 Feature selection

In the original study by Longi et al. [14], the typing profiles were constructed using three different types of features: 1. the average latency between any two keys, i.e. the typing speed of the student, 2. single character latencies, i.e. the average latency from any key to a specific key, and 3. digraph latencies, i.e. the average latency from a specific key to a specific key. Since digraph latencies have been shown to work better than the other two types or a combined feature vector that includes all three types [14], we only use digraph latencies in our experiments.

In the replication experiments, we used the 100 most common digraphs for building the typing profiles similar to Longi et al. [14]. The most common digraphs were determined by sorting the digraphs by the median amount of times the students had used them in both the training and the test set. One digraph corresponds to one feature, and for a feature to be included from a specific student, the student had to have at least five instances of the digraph in the data. In the machine exam experiments, we calculated the differences in identification accuracy if the amount of features used to build the typing profile is varied. The results for one data set are presented in Figure 1. The figures for other data sets show similar diminishing returns in the increase of accuracy after around 25 features. The 25 most common digraphs for the same set are presented in Table 1. Since the digraphs are from a programming context, the most common ones include programming-related digraphs such as *i* -> *n* and *n* -> *t* from writing *int*, and the digraph used in creation of Java's code blocks { -> }.

3.5 Identification

The identification is based on comparing typing profiles estimated during the test context against personal typing profiles learned for each student during the course, using Euclidean distance between the profiles for finding the nearest ones. A simple nearest-neighbor classifier would give the best estimate for the identity of the student, but for the purpose of verifying whether the student is who he or she claims to be it is not necessary to solve this full classification problem. Instead, it is sufficient that the correct training profile is within top k closest profiles (Longi et al. [14] called this *acceptance threshold*). The parameter k controls the balance between two types of error: Large k increases the accuracy for recognizing the true identities, but also makes it easier for false identities to pass the identification test. The probability for that is k/N, where N is the number of students. In the following experiments we will vary the choice of k, searching for the smallest k that has sufficiently high accuracy.

4. EXPERIMENTS AND RESULTS

In this section, we present the experiments we conducted to answer our research questions and their results.

4.1 Replication

To answer the question *"How does the identification accuracy vary between data sets?"*, we test identification accuracy with four data sets from different courses. Previous work has tested identification using data sets with around 200 students [14]. We include data sets with fewer students in our replication experiment to see how the accuracy of identification changes when there are fewer students.

The results of our replication experiments are shown in Table 2. There, data set 1 is from an introductory programming course in spring 2015. Data set 2 is from an advanced programming course in spring 2015. Data sets 3 and 4 are from an introductory and an advanced programming courses held during the autumn semester of 2015. In each data set, we used the first six weeks of the course for building the training set typing profiles and the last week of the course for building the test set typing profile. We achieve similar results having above 90% accuracy with every data set regardless of the acceptance threshold. The results indicate that identification can be conducted reliably even with smaller data sets.

4.2 Controlled Machine Exam Identification Accuracy

Next, we answer the question *"How does a controlled machine examination situation affect the identification accuracy?"*. Here, we use data from the spring of 2015 courses at the University of Helsinki, where the students had to complete a mandatory machine examination in a controlled environment at university premises. The results are presented in Table 3.

The students can be identified with quite high accuracy, even though the identification accuracy is significantly lower than the accuracy achieved in Section 4.1. Especially precise identification, i.e. identifying a student with a threshold of one, does not work as well in the exam as it does for a larger data set that consists of one week's worth of programming.

In addition to studying the identification accuracy in a controlled machine examination, we study the effect of feature quantity on identification accuracy to see how the feature count influences the identification accuracy. The results show that for identifying students precisely, i.e. with an acceptance threshold of one, 50 features seem to be enough as the increase in accuracy with 100 features is not signi-

Table 1: The 25 most common digraphs in a programming course which were used in building the typing profiles for that course.

from key	space	t	n	l	a	a	=	r	j	t	{	l	h	i	s	o	backspace	k	i	t	v	u	t	e	u
to key	=	u	t	u	r	t	space	i	a	h	}	i	i	n	t	u	backspace	u	s	a	a	k	space	t	t

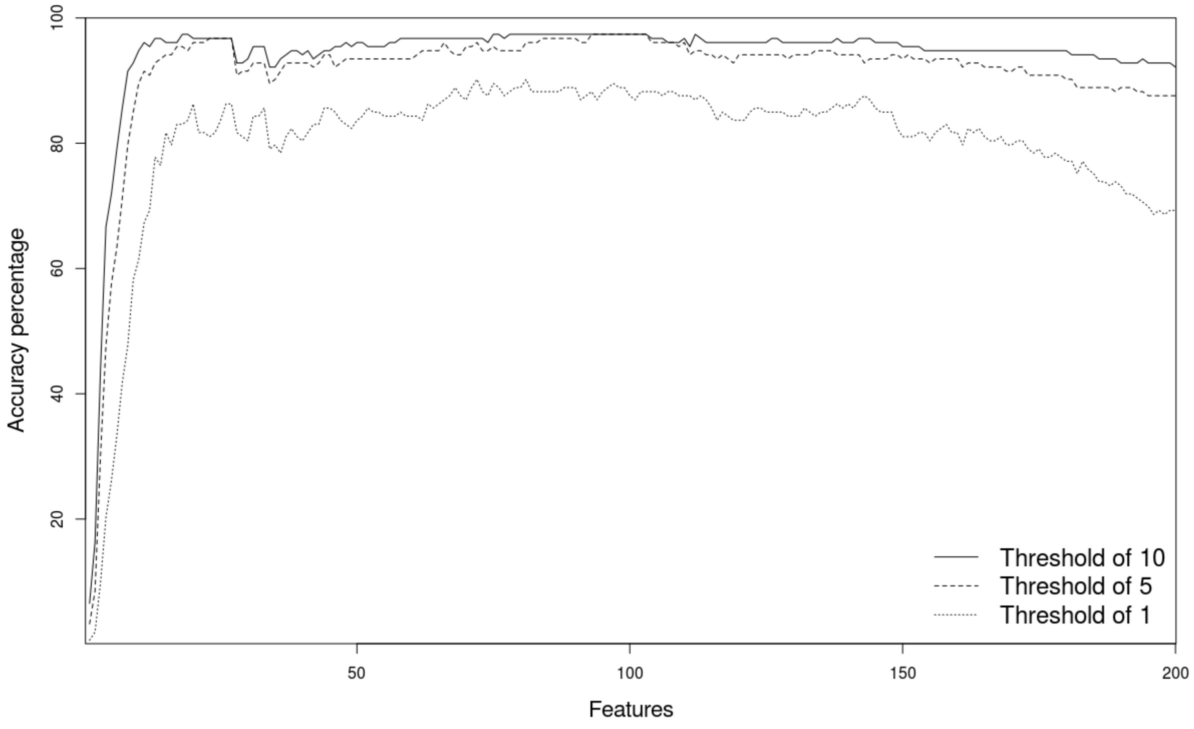

Figure 1: Smoothed identification accuracy plotted against the number of features. The threshold specifies the number of students that are considered to be correct for identification purposes. The figure shows that using around 25 features provides a good identification accuracy with all three thresholds and that the accuracy starts to deteriorate after around 150 features.

Table 2: Identification Accuracy with Different Data Sets

Data set	Students	Threshold 1	Threshold 5	Threshold 10
1	69	94.03%	97.02%	97.02%
2	61	93.22%	100%	100%
3	153	91.37%	97.84%	99.28%
4	128	94.64%	100%	100%

ficant enough to warrant increased complexity. If we allow the student to be within an acceptance threshold of 5 or 10, 25 features seem to suffice. In the advanced course, using 10 features seems to achieve a quite reliable identification accuracy of 95% with an acceptance threshold of five.

4.3 Uncontrolled Machine Exam Identification Accuracy

Last, we answer the question *How does an uncontrolled machine examination situation affect the identification accuracy?*. In this experiment, we use data from the programming courses held at the University of Helsinki in the autumn of 2015. The results are presented in Table 4.

The results show that identification accuracy in an uncontrolled machine exam is slightly worse than in a controlled one. Regardless, the identification accuracy is still sufficiently high especially if we allow for an acceptance threshold of 10 and use at least 25 features.

Contrary to the results in 4.2, increasing the feature amount from 25 to 50 does not improve the identification accuracy significantly when the acceptance threshold is one. 25 features seem to suffice with any threshold to achieve a high identification accuracy. In the beginner course, having 50 features instead of 25 has a detrimental effect on accuracy, which is lower with all three threshold levels when using 50 features compared to using 25.

5. DISCUSSION

Previous work by Bennedsen and Caspersen [2] argues strongly for having machine examination on introductory

Table 3: Identification Accuracy in a Controlled Machine Exam

Beginner course with 69 students

Features	Threshold 1	Threshold 5	Threshold 10
10	44.93%	78.26%	89.86%
25	75.36%	92.75%	97.10%
50	86.89%	100%	100%
100	86.89%	100%	100%

Advanced course with 61 students

Features	Threshold 1	Threshold 5	Threshold 10
10	73.77%	95.08%	98.36%
25	85.25%	96.72%	98.36%
50	91.37%	97.84%	99.28%
100	94.64%	100%	100%

Table 4: Identification Accuracy in an Uncontrolled Machine Exam

Beginner course with 153 students

Features	Threshold 1	Threshold 5	Threshold 10
10	67.32%	91.50%	96.08%
25	86.28%	96.73%	96.73%
50	84.31%	93.46%	96.08%
100	86.93%	97.39%	97.39%

Advanced course with 128 students

Features	Threshold 1	Threshold 5	Threshold 10
10	68.75%	86.72%	91.41%
25	86.72%	92.97%	96.09%
50	87.50%	94.53%	96.09%
100	89.06%	94.53%	96.09%

programming courses. However, a big limitation for having machine examinations is the cost of overseeing students taking the exam. Our experiments show that it is possible to identify students in a machine examination based on their typing profiles, which means that the cost of machine examinations could be alleviated by having the students complete the exam remotely on their own devices, since cheating students could be identified based on their typing patterns. However, condemning a student for cheating solely based on their typing profile is not advisable, since there could be other factors that affect typing such as exam stress or a broken arm. Nevertheless, what could be done is that a flag could be raised in situations where the student is suspected of cheating based on their typing, and further analysis is performed manually. A limitation of our approach is that keystroke analysis can only identify cases where a student has someone else complete the exam for them, but not cases where the whole course is taken by someone else than the student. Since we can only observe typing, a student could cheat by having a friend help them during the exam, but do all the typing himself. However, at the same time, such behavior might likely also influence the typing patterns in the same way as changing from transcribed to free-text does [17].

In most of our experiments, an acceptance threshold of 5 seems to have about as good performance as a threshold of 10, and both perform significantly better than exact identification, i.e. using a threshold of one. There are a few exceptions though, which suggests that to be certain, a threshold of 10 should be used. With only 25 features and a threshold of 10, we can get over 95% identification accuracy with all data sets in our experiments. This means that in a real-world scenario, only 5% of the cases would be false positives, i.e. identifying a "cheating" student where there is no cheating. The acceptance threshold is still small enough to guarantee reasonably low false negative rate; for all contexts the probability of an impostor passing the identification test is below 10%.

The fact that no major difference in the identification accuracies between the uncontrolled examination and the controlled examination could indicate that there is some cheating, since hypothetically, the results should be better if the students are allowed to complete the exam on the same computer that they have used during the exercises – previous work has shown that changing keyboards had an adverse effect on identification [8]. However, there could be other unknown factors that influence the results. For example, the fairly low-dimensional typing profiles and relatively high acceptance threshold might make the identification algorithm robust for such changes.

To achieve optimal identification accuracy, at least 25 features should be used. This is less than used by Longi et al. [14], who used 100 features. In our experiments the accuracy gain for going from 25 to 50 features was only marginal, and that around 150 to 200 features the accuracy already drops visibly (see Figure 1). This discrepancy compared to the earlier study is to be expected since the training and test context differ more in our setup; the most frequent features are likely to be more robust against changes caused by different keyboard or other external factors. For highly controlled setups one could use also features that would be too fragile for the less controlled test situations in our experiments.

6. CONCLUSIONS AND FUTURE WORK

In this work, we explored how students' typing patterns could be used to authenticate them in a computer-based examination. We were able to identify programmers from their typing patterns. This replicates the study by Longi et al. [14] as well as shows that identification is possible with data sets containing a smaller number of students. We also showed that it is possible to identify students in a machine exam based on typing profiles built with exercise data. Identification in an exam setting does not work as well as identification of students on the last week of introductory programming courses, but shows promise that even though the accuracy is lower, it might be enough to catch cheaters in computerized exams.

We also observed that in all of our data sets, the optimal amount of features was around 25. After 25 features, identification accuracy did not improve significantly, and even deteriorated when more than about 150 features were included in the typing profiles. We therefore argue that should keystroke analysis be used in identifying students in a machine examination, the typing profiles should be built considering the 25 most common digraphs to avoid considering features that are too fragile to work in different test settings. To get reliable identification accuracy, we furthermore suggest that students should be identified using the acceptance threshold method [14] and that the acceptance threshold should be 10, i.e. a student is not considered to have cheated if his or her profile is in the top ten closest samples in the training set. This suggestion is valid for student populations of roughly 50-150 students; for smaller courses the threshold needs to be lowered.

For future work, we are interested in researching in more detail how participant and feature quantities affect identification. Here, we only showed that identification is possible in data sets with fewer than a hundred students. We hope to examine how identification accuracy changes when there are hundreds or thousands of students. We are also hopeful that keystroke analysis is included in cheating prevention in machine exams since the results presented here indicate that it is possible to identify students in a machine examination situation. We are also looking for approaches to anonymize the data so that datasets such as ours could be published without privacy concerns.

Acknowledgements

The research was supported in part by the Academy of Finland (project 1266969 and COIN Centre of Excellence) and the Finnish Funding Agency for Innovation (under project Re:Know).

7. REFERENCES

[1] Coursera signature track. https://www.coursera.org/signature/. Accessed: 2015-07-31.

[2] J. Bennedsen and M. E. Caspersen. Assessing process and product: A practical lab exam for an introductory programming course 1. *Innovation in Teaching and Learning in Information and Computer Sciences*, 6(4):183–202, 2007.

[3] F. Bergadano, D. Gunetti, and C. Picardi. User authentication through keystroke dynamics. *ACM Trans. Inf. Syst. Secur.*, 5(4):367–397, Nov. 2002.

[4] S. Cho, C. Han, D. H. Han, and H.-I. Kim. Web-based keystroke dynamics identity verification using neural network. *Journal of organizational computing and electronic commerce*, 10(4):295–307, 2000.

[5] M. Dick, J. Sheard, C. Bareiss, J. Carter, D. Joyce, T. Harding, and C. Laxer. Addressing student cheating: Definitions and solutions. *SIGCSE Bull.*, 35(2):172–184, June 2002.

[6] P. Dowland and S. Furnell. A long-term trial of keystroke profiling using digraph, trigraph and keyword latencies. In Y. Deswarte, F. Cuppens, S. Jajodia, and L. Wang, editors, *Security and Protection in Information Processing Systems*, volume 147 of *IFIP - The International Federation for Information Processing*, pages 275–289. Springer, 2004.

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

[8] D. Gunetti and C. Picardi. Keystroke analysis of free text. *ACM Trans. Inf. Syst. Secur.*, 8(3):312–347, Aug. 2005.

[9] S. Haider, A. Abbas, and A. Zaidi. A multi-technique approach for user identification through keystroke dynamics. In *Systems, Man, and Cybernetics, 2000 IEEE International Conference on*, volume 2, pages 1336–1341 vol.2, 2000.

[10] R. Joyce and G. Gupta. Identity authentication based on keystroke latencies. *Communications of the ACM*, 33(2):168–176, 1990.

[11] M. Karnan, M. Akila, and N. Krishnaraj. Biometric personal authentication using keystroke dynamics: A review. *Applied Soft Computing*, 11(2):1565 – 1573, 2011. The Impact of Soft Computing for the Progress of Artificial Intelligence.

[12] K. S. Killourhy and R. A. Maxion. Free vs. transcribed text for keystroke-dynamics evaluations. In *Proceedings of the 2012 Workshop on Learning from Authoritative Security Experiment Results*, LASER '12, pages 1–8, New York, NY, USA, 2012. ACM.

[13] J. Leinonen, K. Longi, A. Klami, and A. Vihavainen. Automatic inference of programming performance and experience from typing patterns. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*, SIGCSE '16, pages 132–137, New York, NY, USA, 2016. ACM.

[14] K. Longi, J. Leinonen, H. Nygren, J. Salmi, A. Klami, and A. Vihavainen. Identification of programmers from typing patterns. In *Proceedings of the 15th Koli Calling Conference on Computing Education Research*, pages 60–67. ACM, 2015.

[15] J. Monaco, N. Bakelman, S.-H. Cha, and C. Tappert. Recent advances in the development of a long-text-input keystroke biometric authentication system for arbitrary text input. In *Intelligence and Security Informatics Conference (EISIC), 2013 European*, pages 60–66, Aug 2013.

[16] J. Monaco, J. Stewart, S.-H. Cha, and C. Tappert. Behavioral biometric verification of student identity in online course assessment and authentication of authors in literary works. In *Biometrics: Theory, Applications and Systems (BTAS), 2013 IEEE Sixth International Conference on*, pages 1–8, Sept 2013.

[17] F. Monrose and A. Rubin. Authentication via keystroke dynamics. In *Proceedings of the 4th ACM Conference on Computer and Communications Security*, CCS '97, pages 48–56, New York, NY, USA, 1997. ACM.

[18] A. Peacock, X. Ke, and M. Wilkerson. Typing patterns: A key to user identification. *IEEE Security & Privacy*, 2(5):40–47, 2004.

[19] J. Sheard, M. Dick, S. Markham, I. Macdonald, and M. Walsh. Cheating and plagiarism: perceptions and practices of first year it students. In *ACM SIGCSE Bulletin*, volume 34, pages 183–187. ACM, 2002.

[20] J. Sheard, Simon, A. Carbone, D. D'Souza, and M. Hamilton. Assessment of programming: Pedagogical foundations of exams. In *Proceedings of the 18th ACM Conference on Innovation and Technology in Computer Science Education*, ITiCSE '13, pages 141–146, New York, NY, USA, 2013. ACM.

[21] J. Stewart, J. Monaco, S.-H. Cha, and C. Tappert. An investigation of keystroke and stylometry traits for authenticating online test takers. In *Biometrics (IJCB), 2011 International Joint Conference on*, pages 1–7, Oct 2011.

[22] R. C. Thomas, A. Karahasanovic, and G. E. Kennedy. 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, pages 127–134, Darlinghurst, Australia, Australia, 2005. Australian Computer Society, Inc.

[23] A. Vihavainen, T. Vikberg, M. Luukkainen, and M. Pärtel. Scaffolding students' learning using test my code. In *Proceedings of the 18th ACM Conference on Innovation and Technology in Computer Science Education*, ITiCSE '13, pages 117–122, New York, NY, USA, 2013. ACM.

[24] M. Villani, C. Tappert, G. Ngo, J. Simone, H. Fort, and S.-H. Cha. Keystroke biometric recognition studies on long-text input under ideal and application-oriented conditions. In *Computer Vision and Pattern Recognition Workshop, 2006. CVPRW '06. Conference on*, pages 39–39, June 2006.

[25] E. Yu and S. Cho. Ga-svm wrapper approach for feature subset selection in keystroke dynamics identity verification. In *Neural Networks, 2003. Proceedings of the International Joint Conference on*, volume 3, pages 2253–2257, July 2003.