Block-Based Learning Analytics Repository and Dashboard: Towards an Interface between Researcher and Educator

Max Kesselbacher
max.kesselbacher@aau.at
Universität Klagenfurt
Klagenfurt, Austria

Kevin Wiltschnig
kevin.wiltschnig@aau.at
Universität Klagenfurt
Klagenfurt, Austria

Andreas Bollin
andreas.bollin@aau.at
Universität Klagenfurt
Klagenfurt, Austria

ABSTRACT

The collection of programming session data is a cornerstone of programming learning analytics research. For text-based programming, there are data collection projects, like BlueJ's Blackbox, which provide access to the data and thereby facilitate additional research as well as verification. For block-based programming, only data sets of finished projects but not of programming sessions are available. We introduce a data repository that is extendable by implementing instrumentation plugins for various IDEs. The currently supported features are: data collection of text-based and block-based programming sessions, curated user self-registration and filtered data download. This then enables us to implement an educator dashboard in the future, making use of live programming session data to incorporate educators and students into learning analytics research.

CCS CONCEPTS

• **Social and professional topics → Student assessment**; **K-12 education**.

KEYWORDS

learning analytics, data repository, programming education

ACM Reference Format:
Max Kesselbacher, Kevin Wiltschnig, and Andreas Bollin. 2020. Block-Based Learning Analytics Repository and Dashboard: Towards an Interface between Researcher and Educator. In *Workshop in Primary and Secondary Computing Education (WiPSCE '20), October 28–30, 2020, Virtual Event, Germany.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3421590.3421662

1 INTRODUCTION AND RELATED WORK

Learning analytics (LA) research uses various data from educational processes to aid in evaluation and assessment of students as well as design of courses and tools, and thereby aims to improve education. In programming, LA research specifically uses data that originate from programming sessions.

Seminal publications related to LA in programming stress the importance of re-analysis, replication and reproduction to independently verify experimental findings [6], and propose a process model and research agenda for IDE-based LA scenarios [5]. Both points are facilitated by data collection projects like BlueJ's Blackbox [2], collecting and sharing data on a large scale; systematic data collection clearly provides benefits to the research field.

Many young learners make their first contact with programming through the use of block-based programming environments like Scratch. This led to block-based programming processes being researched from multiple viewpoints, illustrated with the following examples. Filva et al. [3] analyze clickstreams in order to assess students' behavioural patterns. Frädrich et al. [4] use static code analysis to detect bug patterns, and evaluate their severeness. Kesselbacher and Bollin [7] detect struggling students based on block type usage and programming behaviour. Troiano et al. [8] assess successive program snapshots to evaluate the development of learners' computational thinking skills.

Despite these research efforts, there is, to the best of our knowledge, no project that systematically collects block-based programming session data of learners. Aivaloglou et al. [1] published a Scratch data set consisting of about 250k scraped Scratch 2 projects. However, each Scratch project only consists of one code snapshot from the time when it was scraped.

In this contribution, we introduce a data repository to collect and share incremental snapshots of block-based programming sessions. We also propose avenues of future research, making use of live programming session data in an educator dashboard in order to approach new kinds of LA research questions centered on educators and students.

2 DESIGN AND IMPLEMENTATION

Our block-based programming data repository consists of two parts[1]. The first part is a headless data collection websocket endpoint which can dynamically be extended by implementing instrumentation plugins for various IDEs. Currently supported IDEs include: an instrumented Scratch 3 server running at our department, plugins for IDEA IDEs (IntelliJ, Android Studio, PyCharm), and Arduino IDE. The second part is a web-based user interface which handles access to the data repository. The main functionalities are user self-registration and data download.

Users can register for three roles: block-based and text-based research role, which enables data download for corresponding supported IDEs, and educator role, which is to be implemented in the future and described in Section 3.1. Registering for specific roles serves the purpose of tracking usage statistics and, ultimately, research interest in the different types of data. Any researcher can register for roles they do not currently have. Role assignment is curated by our department[2].

WiPSCE '20, October 28–30, 2020, Virtual Event, Germany
© 2020 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-8759-0/20/10.
https://doi.org/10.1145/3421590.3421662

[1] Link to the source code repository: https://gitlab-iid.aau.at/seqtrex

[2] The web-interface is available here: https://seqtrex-dashboard.aau.at/

We implemented several filter options to refine and limit the data when downloading programming session data from the repository. The filter options are:

- Start and end date of programming sessions
- Platforms that represent the supported IDEs, dependent on the user's roles (text-based and/or block-based)
- Change granularity, dependent on the selected platforms. The currently implemented granularities range from key strokes to compilations [6]
- Three free text fields, which support full text inclusion search of source file names, user names, and error texts (like compilation errors)

These filter options are designed to accommodate different study designs, and different teaching contexts. For example, filtering for source file names can help in recognizing those related to a given task when downloading data for a longitudinal study.

Programming data is downloaded as a `.zip` file, with individual programming sessions comprising directories containing incremental program snapshots (program source code or serialized blocks) and change information (e.g. changed blocks or key strokes), ordered by creation time stamp.

3 PROSPECTIVE FUTURE RESEARCH

3.1 Educator Dashboard

Establishing a programming data repository to collectively share data from programming sessions alone does not empower fundamentally new research. For example, the lack of information regarding the learning and teaching context is identified as a major drawback of the Blackbox dataset [2]. To remedy this drawback, we plan to add another functionality to the web-based user interface, in the form of an educator dashboard.

The educator dashboard should make it possible for researchers as well as teachers to plan programming lessons. This includes:

- Providing information of the learning and teaching context (demographic information, lesson information, supposed learner age, etc.)
- Adding supplemental learning and teaching material
- Creating data collection groups in order to link programming sessions to programming lessons
- Taking notes on programming sessions that could be used for feedback and grading

These features all serve the purpose of adding contextual information to programming sessions.

3.2 Incorporating Educators into LA Research

The educator dashboard makes it possible to directly incorporate educators into research processes of LA in programming. For example, different depictions and evaluations of programming sessions can be shown to educators in real time, while students complete their tasks. Questions that could be answered by incorporating educators include:

RQ_{e1} Which depictions and evaluations of program sessions help educators with student assessment?

RQ_{e2} Which depictions and evaluations are best suited to identify struggling student?

Answers to these questions could help understanding the use and benefit of LA for educators.

3.3 Incorporating Students into LA Research

In turn, an extension of the dashboard could also incorporate students into research processes of LA in programming. Such a research avenue could serve as a pre-stage to evaluate depictions and evaluations of programming sessions before integrating them in IDEs. Questions that could be answered by incorporating students include:

RQ_{s1} Which depictions and evaluations of program sessions help students with programming skill acquisition?

RQ_{s2} How is the usefulness of different depictions and evaluations related to students' programming skills?

Answers to these questions could help shaping the integration of LA data in (block-based) IDEs, improving education by empowering students to use data from their own processes.

4 CONCLUSION

In this contribution, we introduce an extendable data repository for learning analytics (LA) in programming which currently supports text-based but also block-based data collection, the latter in the form of an instrumented Scratch 3 server running at our department. The data repository also supports curated user self-registration and filtered data download.

Moreover, we propose ideas for future research to incorporate educators and students into LA research by implementing an educator dashboard that makes use of live programming session data. In the light of the process model of Hundhausen et al [5], this uniquely addresses the intersection of *analyzing/processing data* and *delivering interventions*.

REFERENCES

[1] Efthimia Aivaloglou, Felienne Hermans, Jesús Moreno-León, and Gregorio Robles. 2017. A Dataset of Scratch Programs: Scraped, Shaped and Scored. In *2017 IEEE/ACM 14th MSR*. 511–514.

[2] Neil C.C. Brown, Amjad Altadmri, Sue Sentance, and Michael Kölling. 2018. Blackbox, five years on: An evaluation of a large-scale programming data collection project. *Proceedings of the 2018 ACM Conference ICER* (2018), 196–204.

[3] Daniel A. Filvà, Marc A. Forment, Francisco J. García-Peñalvo, David F. Escudero, and María J. Casañ. 2019. Clickstream for learning analytics to assess students' behavior with Scratch. *Future Generation Computer Systems* 93 (2019), 673–686.

[4] Christoph Frädrich, Florian Obermüller, Nina Körber, Ute Heuer, and Gordon Fraser. 2020. Common Bugs in Scratch Programs. In *Proceedings of the 2020 ACM Conference ITiCSE* (Trondheim, Norway) *(ITiCSE '20)*. Association for Computing Machinery, New York, NY, USA, 89–95.

[5] Christopher D. Hundhausen, Daniel M. Olivares, and Adam S. Carter. 2017. IDE-Based Learning Analytics for Computing Education: A Process Model, Critical Review, and Research Agenda. *ACM Trans. Comput. Educ* 17, 26 (2017), 1–26.

[6] Petri Ihantola, Arto Vihavainen, Alireza Ahadi, Matthew Butler, Jürgen Börstler, Stephen H. Edwards, Essi Isohanni, Ari Korhonen, Andrew Petersen, Kelly Rivers, Miguel Rubio, Judy Sheard, Bronius Skupas, Jaime Spacco, Claudia Szabo, and Daniel Toll. 2015. Educational Data Mining and Learning Analytics in Programming: Literature Review and Case Studies. *ITiCSE WGR'16* (2015), 41–63.

[7] Max Kesselbacher and Andreas Bollin. 2019. Discriminating Programming Strategies in Scratch – Making the Difference between Novice and Experienced Programmers. In *14th Workshop in Primary and Secondary Computing Education (WiPSCE 19), October 23–25, 2019* (Glasgow, Scotland UK). ACM New York, 10.

[8] Giovanni M. Troiano, Sam Snodgrass, Erinc Argimak, Gregorio Robles, Gillian Smith, Michael Cassidy, Eli Tucker-Raymond, Gillian Puttick, and Casper Harteveld. 2019. Is My Game OK Dr. Scratch?: Exploring Programming and Computational Thinking Development via Metrics in Student-Designed Serious Games for STEM. In *Proceedings IDC '19* (Boise, ID, USA). ACM, New York, NY, USA, 208–219.