Measuring Young Learners' Open-ended Agent-based Programming Practices with Learning Analytics *

John Chen Uri Wilensky

Abstract

Agent-based modeling (ABM) has been recognized as an important component of computational thinking and literacy. Agent-based programming (ABP) is the computational foundation of ABM. In this study, we propose a novel method to programmatically measure young learners' ABP practices in open-ended programming contexts with a 4-stage model. We present results from the online community of Turtle Universe, a new version of NetLogo designed for mobile platforms and younger learners in informal contexts. We draw on a dataset of 2,300 block-based and text-based projects shared by out-of-school, unsupervised learners. Our approach generally succeeded in measuring learners' open-ended ABP practices. Differences were found among projects of different stages and modalities. We discuss the implications of our approach and design of block-based ABP environments.
Keywords: *Agent-based Programming; Agent-based Modeling; Learning Analytics; Open-ended Learning Contexts; Online Learning Communities*

1 Introduction

Computational modeling, especially agent-based modeling (ABM), has been recognized as a key component of computational thinking and literacy (Wilensky, Brady, & Horn, 2014; Wilensky & Resnick, 1999). In recent years, agent-based modeling and programming practices have been extensively studied. Since ABM naturally combines with scientific disciplines, it has been widely studied, accepted, and implemented in school and non-school contexts.

All agent-based models are implemented by agent-based programming (ABP), or agent-oriented programming (Shoham, 1993). As such, the learning of ABP becomes an essential part of ABM learning (Gilbert & Terna, 2000). While many studies of ABM in learning have focused on measuring the disciplinary or mathematical aspects of learning (e.g., Blikstein (2011); Wilkerson-Jerde and Wilensky (2015)), some studies also emphasized the programming aspect of ABM (e.g., Berland, Martin, Benton, Petrick Smith, and Davis (2013); Blikstein (2011); Weintrop et al. (2016)). They made important progress in helping researchers and educators understand, support, and scale the learning of ABM or ABP in a variety of learning contexts.

Although the implementation of ABM in learning would naturally involve ABP, many previous studies only used questionnaires or interviews to measure ABM learning (e.g., Musaeus and Musaeus (2019)). When program code is included in the analysis, the analysis either relies on a specific disciplinary learning context (e.g. Wagh, Cook-Whitt, and Wilensky (2017)) or relates to general programming instead of ABP (e.g. Blikstein (2011)). Hence, an approach is needed to understand learners' ABP practices in open-ended programming contexts.

In this paper, we introduce a novel learning analytics method for achieving this goal. We were informed by extensive previous research on learning analytics in constructionist learning environments (Berland, Baker, & Blikstein, 2014) and studies that programmatically measure learning through coding processes or results (e.g., Berland, Davis, and Smith (2015); Tissenbaum and Kumar (2019)). Instead of focusing on basic computational concepts or coding processes, we designed our approach specific enough

*Submission to the 2023 American Educational Research Association Annual Meeting

to focus on ABP, but general enough to cover different programming contexts including learner-driven open-ended ones. To achieve this, we propose an automated 4-stage model that could identify ABP practices from any NetLogo model. Using this model, we answer the following research question:

> *What are some characteristics of young learners' out-of-school, unsupervised ABP practices in Turtle Universe's online community?*

Turtle Universe (previously NetLogo Mobile) is a new incarnation of NetLogo that aims to bring ABP into out-of-school, unsupervised learning contexts (Chen & Wilensky, 2020, 2022). It supports both text-based and block-based variants of NetLogo (Horn, Brady, Hjorth, Wagh, & Wilensky, 2014; Wilensky, 1999). In the past 12 months, 56,000 young learners started to use Turtle Universe and shared 2,300 projects based on their own goals and interests. We present empirical findings based on learners' original projects from the online community of Turtle Universe. Then, we discuss the implications for learning analytics and the design of learning environments.

2 Methodology

The online community of Turtle Universe supports young learners' sharing and remixing of NetLogo projects. As we did not prescribe concrete learning or programming goals, learners shared a wide range of NetLogo projects with different goals, programming styles, and levels of complexity. Different from other NetLogo versions, most young learners used Turtle Universe in informal learning contexts. Thus, our dataset from Turtle Universe community's shared projects provides a diverse sample set to apply our novel approach. Similar to the Scratch community, many learners remixed models from the models library or other learners' projects and shared the remixed projects with the community. While our approach could also work for those projects, in this study, we first focus on original projects. We defined original projects as projects that are completely new or made substantial changes to their parent project.

We calculated each project's distance from its parent project and produced histograms for text-based and block-based projects. Based on the histogram (Fig. 1), we used 50% as the cut-off value for text-based projects. Block-based projects are all remixes of several base models, which are either domain-specific or provide a subset of NetLogo features. Since block-based projects often include chunks of code from the base model, we lowered the threshold for original projects to 20%. We ended up with 1,055 original text-based projects from 175 authors and 447 original block-based projects from 166 authors.

Based on previous studies of ABP in computer science, we categorized all NetLogo projects into four conceptual stages (Table. 1), from without any agent (stage 0), only using single agents (stage 1), to simple (stage 2), branching (stage 3), and communicative ABP (stage 4). To programmatically apply our conceptual model, we parsed each project's code into a simplified abstract syntax tree (AST) to understand the logical structure of the NetLogo code. Then, we transformed our definition into programmatic rules and applied on ASTs with Javascript.

To verify our approach, we first applied our programmatic rules to 83 models from Turtle Universe's models library. Most models come from NetLogo's models library, with a few additional models focusing on introducing TU's unique features. As expected, ABP practices were identified in all models, and the majority of models were in stages 3 (39%) and 4 (45%). Then, we manually verified the results with randomly sampled projects and models. In all but one case, our automatic learning analytics approach matches manual categorization. The only exception happened when a learner initialized the modeling world with agents, but then used non-agent-based approaches for most other parts of her project.

To understand the learners' purposes and ways learners programmed in different stages, we randomly sampled cases from each stage and thematically coded them until reaching theoretical saturation. To examine whether different stages have implications on code readability and complexity, we measured each project using metrics from previous studies (e.g. Vendome, Rao, and Giabbanelli (2020)). We measured the readability of code using average line length, average procedure length, and the number of comments; and complexity of code using numbers of procedures, conditionals, loops, and unique token. To compare projects of different stages, we use Kruskal-Wallis (non-parametric) one-way ANOVA and Dunn's pairwise test to examine if each metric is significantly different between groups.

Below, we first report the distribution of stages among block-based and text-based projects, then report the differences in readability and complexity metrics between block-based and text-based projects of different stages.

3 Results

3.1 Distribution of Stages

In this section, we report the distribution of stages among block-based and text-based projects and the learners who authored and shared them.

Block-based Projects. We found that every block-based project included some ABP practices. Distributions of both projects (Fig. 2) and authors (Fig. 3) show that more learners stayed at the earlier stages and less at later stages. Here, the technological design of Turtle Universe may have played an important role. While all block-based models in TU support ABP practices, a few of learner projects used ones (e.g. Martin, Bain, Swanson, Horn, and Wilensky (2020)]) that only support practices up to stage 3, leaving learners unable to engage with stage 4 practices.

Text-based Projects. However, text-based projects show a very different pattern. We found that more projects and authors were at later stages than at earlier stages. In addition, we saw projects without ABP practices and found that they were creatively coded for learners' design goals. In Fig. 2 and Fig 3, we could see the same increasing trend for projects and authors of text-based programming.

As expected, most stage 2, 3, 4 projects were for creating agent-based animations, pictures, and games based on emergence with different levels of complexity. Surprisingly, we found that 14% of projects belonging to stage 0, i.e., used no agents at all. We closely examined those projects and found that they were used:

- As a social space. For example, in the "A question about movement of balls" project, the learner only used 'user-message' to show a question to anyone who would enter the project.
- To demonstrate non-agent-based media. In the "Walnut Rock (GIF+Music)" (Fig. 4) project, the learner used a loop to display a sequence of drawings and played music alongside the animation.
- As a container for obfuscated code that we cannot parse. Some learners were protective of their intellectual property and decided to hide the (sometimes encrypted) code in global variables, which could be shared together with code in projects.

Another 3% of projects belonged to stage 1, i.e., their code talked to agents but only individually. We found that those projects were used to create animations or pictures in approaches more in line with Turtle Geometry. For example, the "Special Light for Disco" draws a changing pattern that reacts to touch

interaction. The "Transforming a Circle into a Square" (Fig. 5) project first calculates a list of parameters and then uses several turtles, each to draw a segment of the curve.

3.2 Complexity and Readability

Due to space constraints, we only report the complexity and readability metrics of text-based projects in this proposal. We found that all complexity metrics of code remarkably increased in each stage between 2-4, yet stage 1 non-agent-based projects were an outlier. For readability, while we did not find a trend between stages, comparisons were made between learners' projects and experts' models.

Complexity. Fig. 6 shows the different complexity metrics between stages of text-based projects. All metrics (number of procedures, conditionals, loops, and unique tokens) significantly increased between stage 2-4 projects (pairwise $p<0.05$). To our surprise, except for loops where the difference is not significant, stage 1 projects were significantly more complex than stage 2 ($p<0.05$). In addition, we found that all stage 1 project authors (n=8) also shared projects of stages 2, 3, 4, implying that they could be more advanced learners.

Readability. Fig. 7 demonstrates the different readability metrics between learners' projects and experts' models. While we did not find significant differences between the average length of line and procedure, learners' projects had significantly fewer comments than experts' models ($p<0.05$). The same results were found across every stage. Learners' projects were also much more diverse than experts' models in all three metrics, showing a variety of personal programming styles.

4 Discussion

While learners were sharing projects for their personal goals without prescribed curricular goals, our novel approach generally succeeded in measuring their open-ended ABP practices. However, researchers and practitioners should be cautious about the implications of our stage-based model. First, stage 0 projects come as a surprise. In an open-ended setting, it is surprising but also predictable that children will leverage constructionist learning environments with purposes that designers would never envision. In our case, we found that children mostly used for social and media purposes.

Second, while the four stages were designed as a linear scale, we found that learners with stage 1 projects were likely to be more advanced learners. One possible explanation is the design of the programming language or space: both text-based NetLogo language and block-based NetTango programming spaces prioritize multi-agent programming over programming of individual agents. Also, while we noticed the statistical trend between stages, individual projects differ greatly from each other. As a result, stages do not strictly correspond with code complexity. This further necessitates our future work to understand trajectories, in addition to a deeper analysis of the end results, of learners' ABP practices.

Finally, our results point to the design of learning environments. We found that more block-based projects stayed at the periphery stages than text-based projects. One reason is that while text-based and block-based NetLogo programming both strive to be low threshold and high ceiling, the position of threshold and ceiling could be conceptually different. While block-based ABP environments tend to be more simple than text-based counterparts and focus more on domain-specific contents, it is still important for researchers and educators to think about which stages are made possible through each design. While certain disciplinary contexts might only necessitate stage 2 or 3, it could still be beneficial to design rooms for learners to venture into higher ABP stages as well.

For researchers who are interested in the dataset we used, please reach out to civitas@u.northwestern.edu.

References

Berland, M., Baker, R. S., & Blikstein, P. (2014). Educational data mining and learning analytics: Applications to constructionist research. *Technology, Knowledge and Learning*, *19*(1), 205–220.

Berland, M., Davis, D., & Smith, C. P. (2015). AMOEBA: Designing for collaboration in computer science classrooms through live learning analytics. *International Journal of Computer-Supported Collaborative Learning*, *10*(4), 425–447.

Berland, M., Martin, T., Benton, T., Petrick Smith, C., & Davis, D. (2013). Using learning analytics to understand the learning pathways of novice programmers. *Journal of the Learning Sciences*, *22*(4), 564–599.

Blikstein, P. (2011). Using learning analytics to assess students' behavior in open-ended programming tasks. In *Proceedings of the 1st international conference on learning analytics and knowledge* (pp. 110–116).

Chen, J., & Wilensky, U. (2020). NetLogo Mobile: An Agent-Based Modeling Platform and Community for Learners, Teachers, and Researchers.

Chen, J., & Wilensky, U. (2022). *Turtle Universe.* Evanston, IL: Center for Connected Learning and Computer-Based Modeling, Northwestern University.

Gilbert, N., & Terna, P. (2000). How to build and use agent-based models in social science. *Mind & Society*, *1*(1), 57–72.

Horn, M. S., Brady, C., Hjorth, A., Wagh, A., & Wilensky, U. (2014). Frog pond: a codefirst learning environment on evolution and natural selection. In *Proceedings of the 2014 conference on interaction design and children* (pp. 357–360).

Martin, K., Bain, C., Swanson, H., Horn, M., & Wilensky, U. (2020). Building blocks: kids designing scientific, domain-specific, block-based, agent-based microworlds. In *International conference of the learning sciences (2020).*

Musaeus, L. H., & Musaeus, P. (2019). Computational thinking in the Danish high school: Learning coding, modeling, and content knowledge with Netlogo. In *Proceedings of the 50th acm technical symposium on computer science education* (pp. 913–919).

Shoham, Y. (1993). Agent-oriented programming. *Artificial intelligence*, *60*(1), 51–92.

Tissenbaum, M., & Kumar, V. (2019). See the Collaboration Through the Code: Using Data Mining and CORDTRA Graphs to Analyze Blocks-Based Programming.

Vendome, C., Rao, D. M., & Giabbanelli, P. J. (2020). How do modelers code artificial societies? investigating practices and quality of netlogo codes from large repositories. In *2020 spring simulation conference (springsim)* (pp. 1–12).

Wagh, A., Cook-Whitt, K., & Wilensky, U. (2017). Bridging inquiry-based science and constructionism: Exploring the alignment between students tinkering with code of computational models and goals of inquiry. *Journal of Research in Science Teaching*, *54*(5), 615–641.

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of science education and technology*, *25*(1), 127–147.

Wilensky, U. (1999). *NetLogo. Evanston, IL: Center for connected learning and computer-based modeling, Northwestern University.*

Table 1: Definitions of 4 conceptual stages.

Stage	Category	Definition
Stage 0	Non-agent	The project does not explicitly use agents.
Stage 1	Single-agent	The project only deals with one agent at a time.
Stage 2	Simple multi-agent	The project only deals with all agents or individual agents at a time.
Stage 3	Branching multi-agent	The project deals with multiple agents with branching or filtering. However, each agent does not talk to another.
Stage 4	Communicative multi-agent	The project deals with multiple agents, and individual agents talk to one another.

Wilensky, U., Brady, C. E., & Horn, M. S. (2014). Fostering computational literacy in science classrooms. *Communications of the ACM*, *57*(8), 24–28.

Wilensky, U., & Resnick, M. (1999). Thinking in levels: A dynamic systems approach to making sense of the world. *Journal of Science Education and technology*, *8*(1), 3–19.

Wilkerson-Jerde, M. H., & Wilensky, U. J. (2015). Patterns, probabilities, and people: Making sense of quantitative change in complex systems. *Journal of the Learning Sciences*, *24*(2), 204–251.

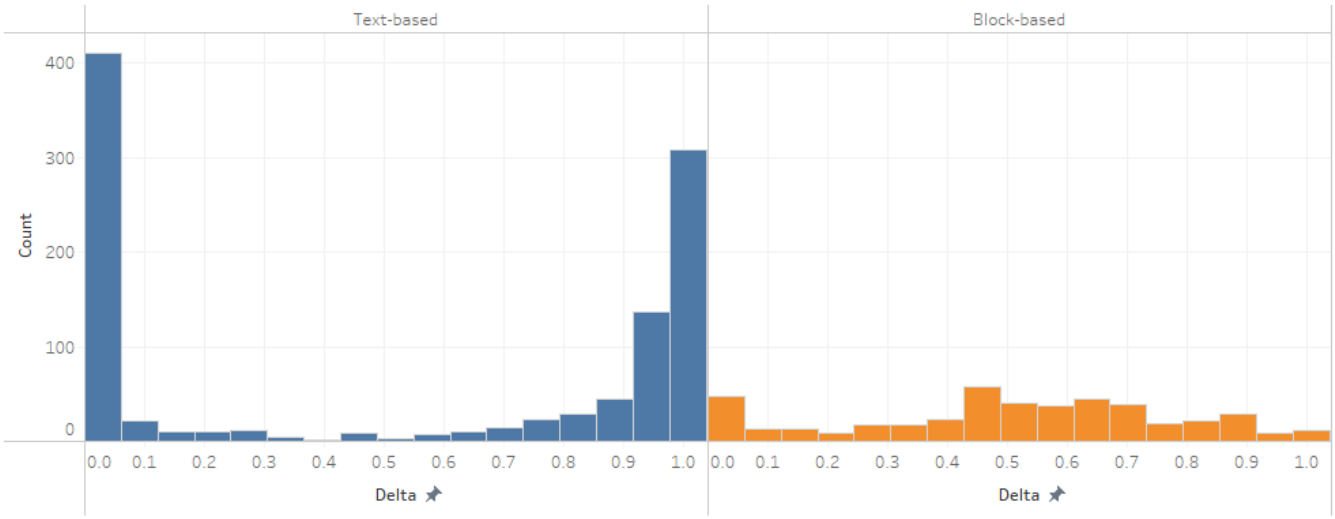

Figure 1: Distribution of difference among learners' projects.

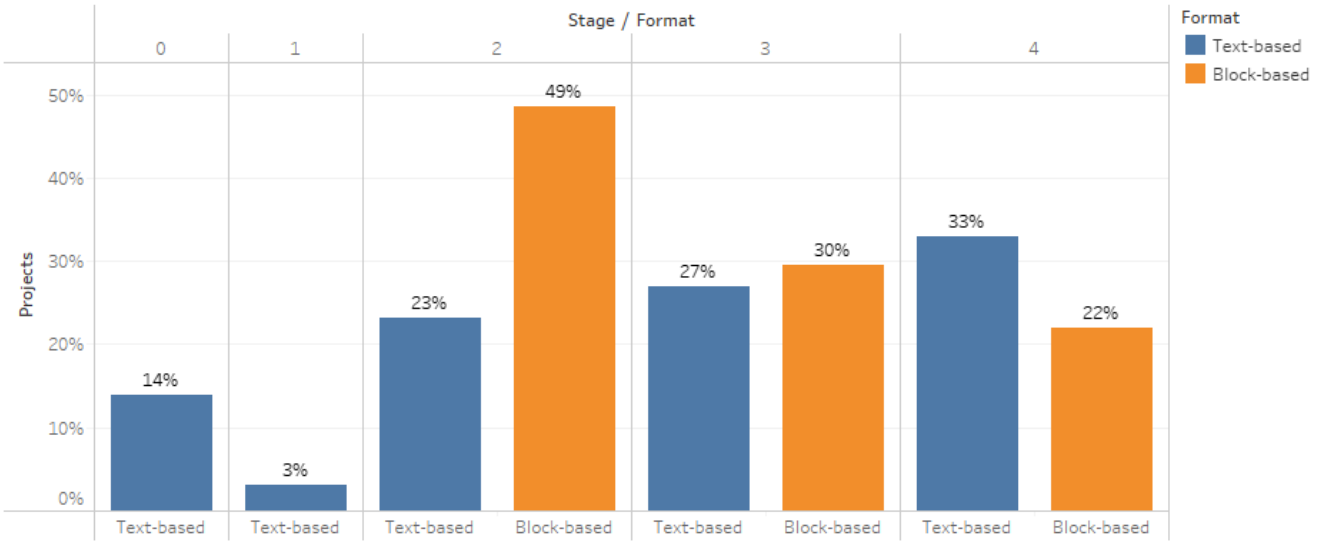

Figure 2: Stage Distribution of Learners' Shared Projects.

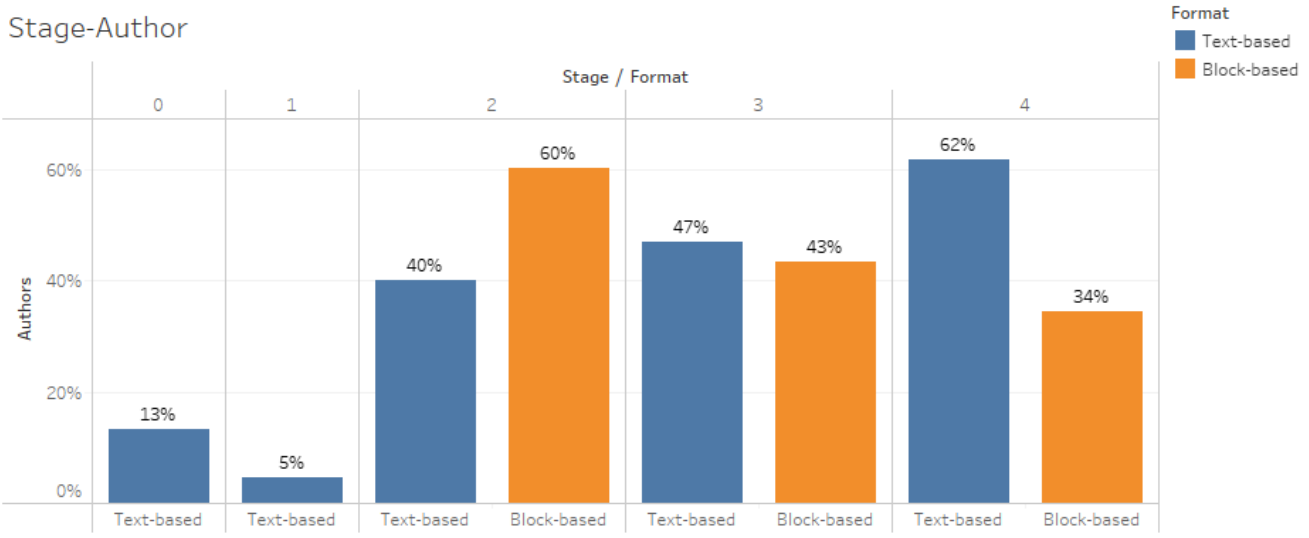

Figure 3: Stage Distribution of Learners with Shared Projects.

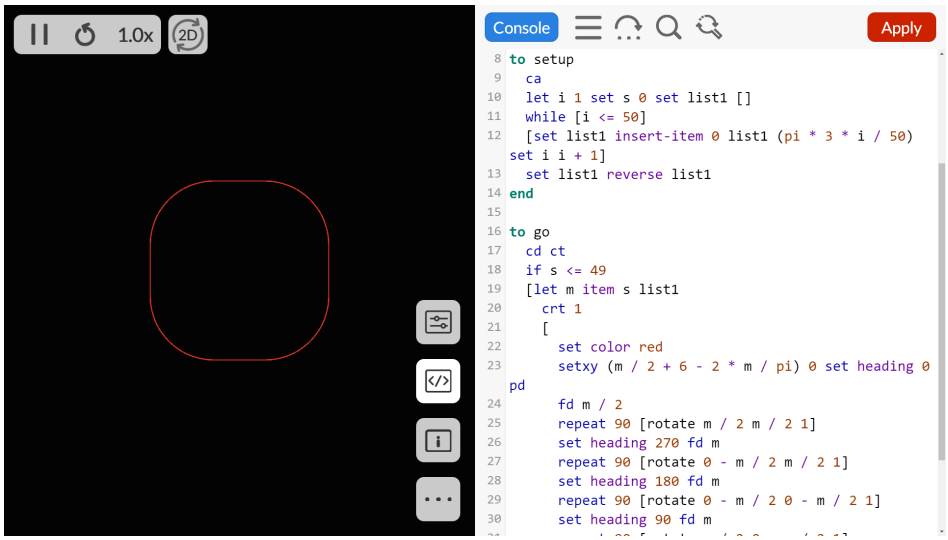

Figure 4: An example Stage 0 project by Weijia Li.

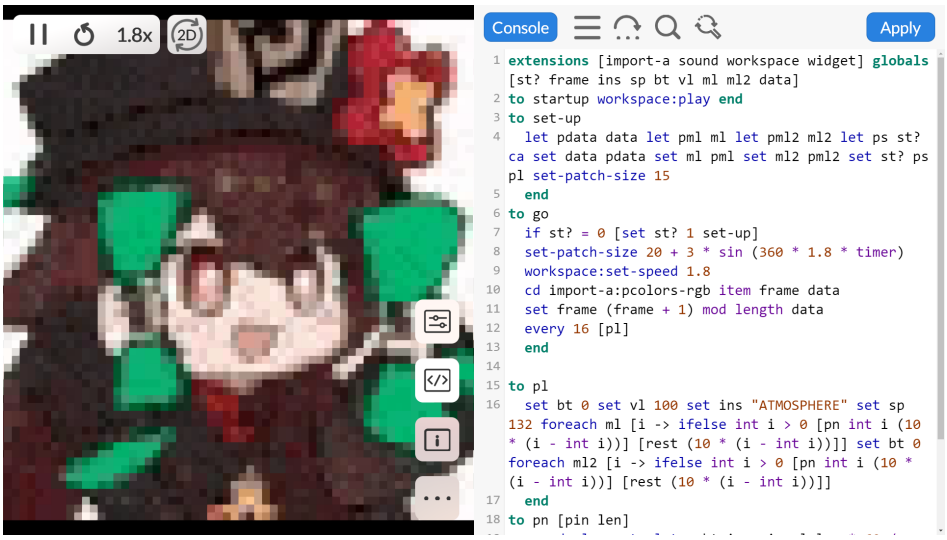

Figure 5: An example Stage 1 project by Brainchon.

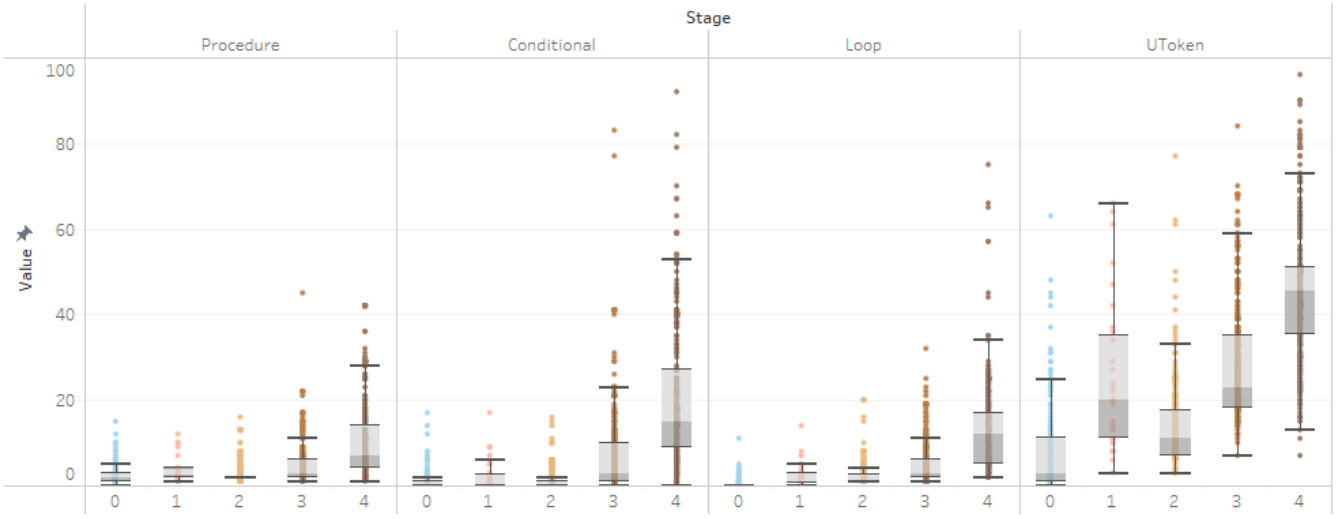

Figure 6: Complexity Metrics of Text-based Projects.

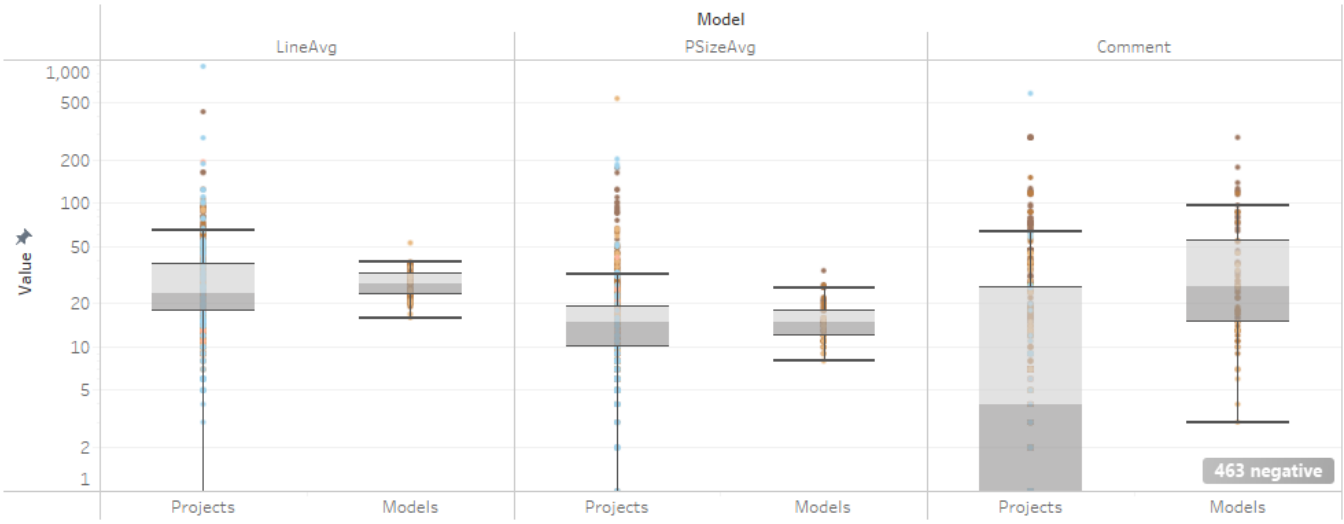

Figure 7: Readability Metrics of Text-based Projects