Clustering Young Children's Coding Project Scores with Machine Learning

Apittha Unahalekhaka
Child Study and Human Development
Tufts University
Medford, USA
apittha.u@tufts.edu

Marina Umaschi Bers
Child Study and Human Development
Tufts University
Medford, USA
marina.bers@tufts.edu

2022 IEEE Global Engineering Education Conference (EDUCON) | 978-1-6654-4434-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/EDUCON52537.2022.9766579

***Abstract*— Engaging children in making creative coding projects is a popular approach for young children to develop computational skills. Children learn to transform their ideas and knowledge into personally meaningful projects. However, the connection between children's coding competencies and their created coding projects remains unclear. This study explored how ScratchJr coding projects created by first and second graders (*n* = 75) differed and whether the project focuses varied by the children's coding competencies. ScratchJr is a free downloadable tablet game for children ages 5-7 to create animated stories and games. Through unsupervised machine learning algorithms, this study grouped coding project concepts by Principal Component Analysis then classified students into four groups by K-means. The results showed that students across the four groups focused on creating different aspects of the ScratchJr projects. Moreover, the median coding competency score of each student group also differed significantly. This study implies that young children with varying levels of coding competency tend to focus on creating different aspects of ScratchJr projects. Findings from this study can help to inform curriculum designers and teachers on essential approaches for guiding children in creating their own coding projects.**

Keywords*— *Early Elementary, Computer Science Education, Project Assessments, Tablet Game, ScratchJr

I. Introduction

Computer science (CS) lessons in the early years have proved effective in developing children's coding mastery and cognitive skills. However, the teaching pedagogy and learning tools must be developmentally appropriate for younger students [1]–[3]. Examples of these tools include board games, robots, and touchscreen applications that use block-based programming instead of text-based [3]–[5]. Developmentally appropriate tools make it possible for children as young as four to create animated story and game projects through coding [6]–[8].

Based on a constructionist approach, valuable learning can happen when children create personally meaningful projects to share with others [7], [9], [10]. Through this process, children learn to integrate disciplinary knowledge and computational skills to solidify their abstract ideas into real-world applications [10], [11]. For example, when creating open-ended projects, they get to develop coding and debugging strategies, project design elements, and storytelling skills.

Learning tools that allow young children to express their creativity through coding are as crucial as an effective learning pedagogy. One of those is ScratchJr, a freely downloadable tablet application that uses block-based programming language (Refer Figure 1), and it has been downloaded over 22 million times [12]. ScratchJr enables children ages five to eight to create open-ended animated projects by connecting various graphical coding blocks [4]. It was designed with a constructionist perspective and promoted more tinkering in young children than task-based coding software [13].

Fig.1 ScratchJr coding project with a swimming whale.

Once children create coding projects, teachers may want to use a reliable project grading rubric to understand how much children learn. Using the newly developed and tested ScratchJr Project Rubric [14], we split the students according to different aspects of their projects and compared their coding competency scores. To achieve this research goal, we utilized machine learning methods to group coding project concepts and cluster students according to the concepts' groupings.

The methodology that this study used was machine learning (ML), which has been increasingly popular in the educational research field, particularly in learning analytics [15], [16]. However, ML is still not as prevalent in the field of early childhood education (ECE). This study aims to

Authorized licensed use limited to: Boston College. Downloaded on October 05,2023 at 15:47:36 UTC from IEEE Xplore. Restrictions apply.

effectively apply ML to ECE research. For example, an ML approach such as Principal Component Analysis (PCA) can help to handle multicollinearity [17] in the ScratchJr Project Rubric concepts by extracting the highly correlated concepts down to a fewer meaningful concept grouping. Therefore, our research questions are as follows:

- RQ1**:** What are the groupings of the coding project concepts?
- RQ2**:** Can the students be classified into clusters according to the coding project concept groups? If so, how are they different?
- RQ3: Are there differences in the coding competency across classified students' clusters?

II. Background

A. Projects and Computational Competencies

Computational skills are important for children of all ages in the 21st century and can be acquired through coding [18]. Studies with students from late elementary and above have investigated the relationship between coding projects and Computational Thinking (CT). For example, Brennan and Resnick gave concrete examples of the CT concepts displayed through Scratch projects (a coding software for children ages 8-16), consisting of "Sequences", "Repeat loops", "Events", "Parallelism", "Conditionals", "Operators", and "Data" [19]. Another study found that different categories of Scratch projects involved different CT mastery scores [20]; specifically, games scored the highest in all CT concepts, followed by stories and animations. The study suggested that children may want to start creating the types of projects with progressively higher CT dimensions.

Nonetheless, a few studies argued that although children create coding projects, these projects might not truly reflect the computational concepts children know. This is particularly true when the complexity of projects can be affected by children's individual aesthetic preferences and external factors such as the curriculum and hints from teachers or friends [21], [22]. Furthermore, artifact (project) analysis focusing on project component frequencies or lengths (e.g., number of blocks and characters, and length of sequences) showed a weak correlation between the project components and related coding concept written questions [23]. Therefore, more investigation should be performed to understand the relationship between coding projects and coding competencies.

B. ScratchJr Project Rubric

The ScratchJr Project Rubric is an artifact-based assessment that aims to capture young children's ability to create imaginative and purposeful ScratchJr coding projects [14]. The ScratchJr Project Rubric has been tested for reliability, and its 13 categories measure both the Coding Concepts and the Project Design aspects of the project (Refer Table I). Moreover, the score of each concept ranges from 1-4 with an increasing concept complexity or component frequencies (Refer Table II). An example of an "Event" scoring scheme is shown in Table II, while a sequence with a score of 4 for "Events" is shown in Figure 2.

TABLE I. Concepts in the ScratchJr Project Rubric

Coding	***Project Design***
Sequencing	Character Customization
Repeats	Background Customization
Events	Animated Look
Parallelism	Sound
Coordination	Number of Characters
Number Parameter	Number of Settings
	Speech Bubble

TABLE II. Example of events scoring scheme

Scores	***Coding Block Requirements***
1	Attempted but there was no correct usage of "to next page", "start on tap, bump, or messages"
2	Used "start on tap, bump", or "to next page" correctly
3	Used "green flag" to "send message" (1 color) and "received message" correctly
4	Used "start on tap or bump" with "received message" and "send message" correctly or send message(s) from a different message color

According to previous studies, concepts in the ScratchJr Project Rubric (Refer to Table I) reflect CT concepts that young children can master [19], [20] such as "Repeats" (flow control), "Events" (logical thinking), "Coordination" (synchronization), "Number Parameters" (data representation), "Sound" and "Speech Bubble" (user interactivity), "Number of Characters and Background" (problem decomposition).

Fig.2 Example sequence with a score of 4 for Events. Tap on a character to move forward, then send a purple and a red message to start executing corresponding sequences that begin with receiving a purple or a red message.

III. Method

A. Dataset

This study is a part of the Coding as Another Language (CAL) project [24], which implements ScratchJr coding curriculum integrated with literacy to school districts in the United States. Participants were 103 first and second graders from two public schools in Minnesota and Arkansas, USA. Incorporated in the 12 weeks CAL ScratchJr lessons, students created numerous ScratchJr projects, but this study only analyzed the final projects from each student (103 projects). The same set of instructions were given across grades to create open-ended stories on ScratchJr using similar themes. For example, first graders re-created "Where the Wild Things Are," written by Maurice Sendak, while second graders re-created "Stellaluna," written by Janell Cannon.

Authorized licensed use limited to: Boston College. Downloaded on October 05,2023 at 15:47:36 UTC from IEEE Xplore. Restrictions apply.

During data screening in this study, five of the 13 rubric concepts, including "Repeat", "Coordination", "Look", "Sound", and "Speech Bubble", had missing values in more than 66% of the sample. In other words, more than half of the students did not include these five concepts in their projects. This might be because students were creating open-ended projects and were free to choose not to include any concepts. Consequently, we decided to *exclude* these five concepts to focus on the other eight commonly used concepts. To ensure completeness in the dataset, we also dropped cases of missing value in any of the eight project concepts. Consequently, there were 75 projects for this study, in which 38 student creators were girls, and 37 student creators were boys.

B. Measures

1) ScratchJr Project Rubric concept scores were the essence of this study. This study focused on eight concepts: "Sequencing", "Events", "Parallelism", "Number Parameter", "Character and Background Customization", and "Number of Characters and Settings". These concepts are valuable for children to make coding projects. *Sequencing* is the number of blocks used to form a functional command. *Events* and *Parallelism* are advanced concepts similar to if-then statements and executing multiple commands simultaneously. *Number Parameter* reflects the ability to use numbers to specify how many times an action should happen. *Customization* and *Number of Characters and Backgrounds* are part of the project design aspect.

The first author of the paper evaluated all projects, while another rater evaluated 30%. The inter-rater reliability score was sufficient with an average Krippendorf's alpha [25] across concepts of 0.94, ranging from 0.83-1.00.

2) Coding Stages Assessment (CSA) is a task-based assessment that measures children's ScratchJr programming language competency [26]. CSA demonstrates content validity in measuring CT and an excellent level of internal consistency, Guttman's $\lambda_6 = 0.94$. The CSA calculates a total weighted score, where questions in higher stages would receive heavier weighting. In this study, only the post-curriculum weighted CSA scores were used for the analysis.

C. Data Analysis Techniques

1) RQ1: Principal Components Analysis (PCA). PCA is an unsupervised machine learning method commonly used to summarize variables into fewer principal components (PC) that can explain variability in the dataset [17]. This study used PCA for dimensionality reduction of the eight rubric concepts as they were moderately to highly correlated. Before fitting PCA, the students' concept scores were standardized from setting their means to zeros.

2) RQ2: K-means is reported to be one of the most suitable partitioning algorithms for educational datasets [27]. K-means is another unsupervised machine learning approach, which partitions similar data points into a specified number of clusters. The algorithm works by minimizing within-cluster variation to avoid over-lapping across clusters [17]. This study used K-means to cluster students based on the concepts grouping from PCA. In other words, students that had similar components on the ScratchJr Project Rubric will be grouped.

3) RQ3: Kruskal–Wallis and Mann-Whitney Post-hoc Test. This study used non-parametric comparison tests, Kruskal-Wallis and Mann-Whitney Post-hoc Test to examine whether the CSA scores significantly differed across student clusters. These tests do not assume population normality and the smallest suggested sample size for the analysis is five [28], [29].

IV. Results

The following subsections responded to the three research questions on concepts grouping, project scores clustering, and coding competencies according to the clusters.

A. RQ1: What are the groupings of the eight coding concepts?

The eight ScratchJr Project Rubric concepts were proper for factor analysis shown from the Bartlett Sphericity of $p < 0.001$ and KMO values of 0.55. The first four principal components (PC) were used as they had eigenvalues of above 1, which means that each component can explain the variance of more than one concept variable [30]. Furthermore, these four components could explain approximately 70% of the overall project rubric score variance, which has shown to be sufficient in an educational study [31]. Table III shows the loadings, which contain the direction and magnitude of the correlation between each concept to a particular PC. In other words, the loadings can help us to label each PC based on the related rubric concept features [32]. This study used the loading cut-off as above |0.5|, which is evident as robust [32]. It is important to note that the negative loadings indicate an inverse relationship between the PC values and the associated rubric concept. Therefore, in this case, the higher the PC, the lower the rubric concepts.

TABLE III. Loadings of each principal component

	PC1	***PC2***	***PC3***	***PC4***
Sequencing	-0.14	**-0.57** [a]	-0.32	-0.01
Events	**0.61** [a]	-0.17	-0.11	0.20
Parallelism	**0.63** [a]	-0.02	-0.06	0.16
Number Parameter	0.31	-0.11	0.04	**-0.56**
Character Customization	-0.18	-0.46	0.30	-0.41
Background Customization	0.23	-0.34	**0.61** [a]	-0.05
Number of Characters	0.15	0.17	-0.47	**-0.64** [a]
Number of Settings	-0.07	**-0.53** [a]	-0.43	0.22

[a] Loading is > |0.5|

PC1 can be summarized as the "**Advanced Coding**" concept, which has the highest positive loadings with "Events" and "Parallelism". Prior studies reported that coding concepts such as "Events" and "Parallelism" were advanced concepts that children infrequently used in their projects [14], [33]. "Events" is not straightforward for young children, as they must be used with different types of coding block such as motion blocks [33]. For example, *if* a cat character bumps (an event block) into a ball, a ball *then* starts

Authorized licensed use limited to: Boston College. Downloaded on October 05,2023 at 15:47:36 UTC from IEEE Xplore. Restrictions apply.

moving forward (a motion block). Furthermore, "Parallelism" is also challenging for young children as they must understand how multiple syntaxes execute together. For example, if a sequence has a move forward block (to the right) plays in parallel with a sequence that has a move up block, the character will move diagonally to the up-right corner of the screen.

PC2 represents *lower* "**Sequence Length and Setting Variety**," which has the highest negative loadings with "Sequencing" and "Number of Settings". A lower "Sequencing" score means there are fewer coding blocks, while a lower "Number of Settings" implies fewer project settings. It is important to note that longer sequences might not always be more advanced as they might be inefficient and compose only of basic blocks.

PC3 is the "**Background Customization**" as it has "Background Customization" as the only concept with the high loading. "Background Customization" criterion in the ScratchJr Project Rubric assesses how elaborately children decorated their projects' backgrounds by counting the number of customizations (e.g., drawing, coloring, stamping, taking photos, and inserting shapes).

PC4 can be summarized as *lower* "**Character Quantity and Coding Efficiency**", which has the highest negative loadings with the "Number of Characters" and the "Number Parameter". A lower "Number of Characters" does not only indicate fewer characters but may also mean that there are fewer functional coding sequences. Furthermore, the "Number Parameter" concept is numbers that children can type on the screen to specify how many times a block should be executed. Therefore, infrequent usage of "Number Parameter" suggests sequences that have repeating blocks side by side, resulting in lower efficiency.

In summary, the eight ScratchJr Project Rubric concepts can be grouped into four categories, which we then identified as: 1) Advanced Coding, 2) Sequence Length and Setting Variety, 3) Background Customization, and 4) Character Quantity and Coding Efficiency. Moreover, these four new groupings were essential to classify students to answer the next research question.

B. RQ2: Can the students be classified into clusters according to the coding project concept groupings?

From the K-Means elbow method and silhouette coefficient, four student clusters were selected. As shown in Figure 3, Cluster 1 (red dots) is the largest group with around 61% of the students, while Clusters 2 and 3 each has approximately 11%.

Cluster 1 (n = 46, 61%), on average, had longer coding sequences with higher efficiency and more characters across story settings than most other clusters (Refer Figure 4). However, their projects included more basic coding concepts and lower "Background Customization".

Cluster 2 (n = 8, 11%) students, on average, showed more Advanced Coding concepts, longer Sequences, and higher Setting Variety (Refer Figure 4). However, the long sequences might not always be efficient. They also used fewer characters across story pages that were not as customized as clusters 3 and 4.

Cluster 3 (n = 13, 17%) students, on average, focused on project design; they decorated their story background the most across a great mixture of story settings (Refer Figure 4). Although their coding Sequences were relatively more extended than the other clusters, they had lower Advanced Coding and Coding Efficiency. Additionally, there might not be many coding sequences in their projects as they had fewer characters.

Cluster 4 (n = 8, 11%) students, on average, had shorter coding Sequences with higher Efficiency and Advanced Coding concept than Cluster 1 and 3 (Refer Figure 4). Although their projects had fewer settings, they contained more characters, which also implies more coding Sequences. Lastly, they also decorated their project background further than clusters 1 and 2.

In summary, students were classified into four groups based on their projects' characteristics. Some student groups focused more on adding complex coding commands to their sequences, while the other groups focused more on customizing their projects' aesthetic. In the next research question, we explored how the student clusters differed in their coding competency scores.

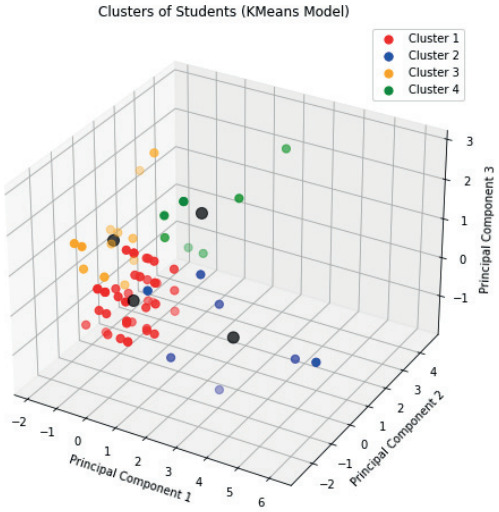

Fig. 3 Clusters of students' scores by principal components

Authorized licensed use limited to: Boston College. Downloaded on October 05,2023 at 15:47:36 UTC from IEEE Xplore. Restrictions apply.

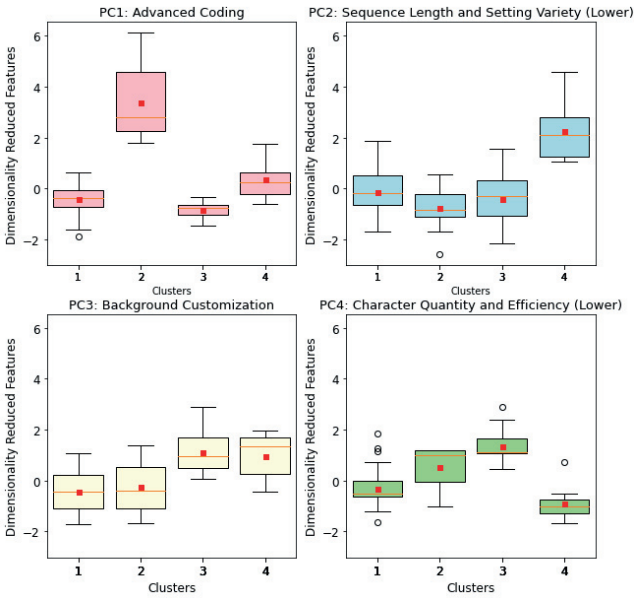

Fig. 4 Dimensionality reduced features of each principal component by clusters. Red square = mean, orange line = median. As PC2 and PC4 had negative loadings, there was an inverse relationship between the concepts and reduced features.

C. *RQ3: Are there differences in the coding competency (CSA score) across classified student clusters?*

Kruskal-Wallis showed that the median differences between the four student clusters significantly differed (F = 18.77, $p < 0.001$). In Table IV, Cluster 2 students had the highest post-CSA score, following by Cluster 4, Cluster 1, and Cluster 3. However, the Mann-Whitney pairwise presented that only the difference between Cluster 2-Cluster 1 ($p < 0.001$), Cluster 2-Cluster 3 ($p < 0.001$), Cluster 3-Cluster 4 ($p < 0.05$) were statistically significant. There was not enough evidence to claim that the median CSA scores of Clusters 2 and 4 and Clusters 1 and 3 were different.

In summary, the four student clusters varied in how they created ScratchJr projects as well as in how they performed on the post-CSA. We found that two student clusters scored relatively higher on the CSA than the two other clusters. These clusters will be discussed in the next section.

TABLE IV. DESCRIPTIVE STATISTICS OF CSA SCORES BY CLUSTERS

Clusters	*N*	*Mean*	*SD*	*SE*	*Median*
1	46	16.32	6.28	0.93	14.25
2	8	27.81	5.85	2.07	26.20
3	13	14.26	4.67	1.29	12.90
4	8	23.14	10.44	3.69	24.35

V. DISCUSSION AND CONCLUSION

Prior studies utilizing artifact analysis showed that children's creations might not accurately reflect the computational concepts they know. Therefore, this study explored the relationship between children's coding projects and their coding competencies, particularly whether young children with varying levels of coding competency create projects with different focus areas.

This study found four new principal component concepts from the eight ScratchJr Project Rubric concepts, which could explain approximately 70% of the total variance. We then named the four components as, "Advanced Coding", "Sequence Length and Setting Variety", "Background Customization", and "Character Quantity and Coding Efficiency". According to these four concept groupings, the K-means clustering algorithm then split the students into four groups with distinct project focus areas.

Cluster 2 and 4 are the two student groups that scored higher on the post-Coding Stages Assessment (CSA). They usually created projects that encompassed more Advanced Concepts or Sequences with higher Efficiency. Students from Cluster 2 used more Advanced Coding concepts (Events and Parallelism) while focusing less on customizing their story backgrounds. These characteristics are represented in the project example shown in Figure 5. In contrast, students from Cluster 4 customized their background more and used lower Advanced Concepts. Furthermore, they also created shorter Sequences with higher coding Efficiency.

Cluster 1, the largest student cluster (61%) was one of the two groups that scored lower on the post-CSA. These two clusters focused on expanding on the story details. Cluster 1 students used simple coding concepts but had relatively longer Sequences with a greater Number of Characters and Setting Variety. An example of a project created by a Cluster 1 student is shown in Figure 6. Similarly, students in Cluster 3 also included a wider variety and customization of the settings.

Fig. 5 A project example from Cluster 2, which shows two sequences playing simultaneously (Parallelism) and sending a message (Events).

Authorized licensed use limited to: Boston College. Downloaded on October 05,2023 at 15:47:36 UTC from IEEE Xplore. Restrictions apply.

Fig. 6 A project example from Cluster 1, which shows a long and basic sequence with motion blocks and sound recording at the end.

This study highlighted two Advanced Coding and CT concepts, "Events" and "Parallelism" [19], [20], which were found in student clusters that scored higher on the post-CSA. However, only a few proportions of students used these advanced concepts. This finding aligns with prior studies that reported how middle school students could use simple CS constructs but infrequently used the complex concepts in their projects [22], [34]. Further study is needed to clarify why children infrequently use the more complex coding blocks, and if the reason is due to the complexity of the concept, then the curriculum may want to provide more examples for learners.

In conclusion, the mixed component values across clusters in this study suggested that both coding and design components were crucial for projects. An extra emphasis on the advanced concepts and efficiency might greatly benefit students' coding competency. To expand students' coding knowledge, teachers may firstly want to encourage their students to tinker with the advanced blocks and secondly, provide more free-play time for students. A study found that children that played with ScratchJr in an informal setting at home used more advanced and broader variety of coding blocks than children at school [35]. Although the reason is inconclusive, the study implied that it might be due to more unstructured learning at home. Finally, it is also crucial for students to get sufficient time to design their projects, as the design aspect is important to complete their personally meaningful creations.

This study explored the general trend between children's coding projects and their coding competencies to guide future studies. A more in-depth investigation should be conducted to conclude whether we can accurately indicate children's coding competencies from their projects.

VI. Limitation and Future Direction

Out of three limitations, the first one was the small sample size. Although there were significant differences across some of the four clusters, the small sample sizes in Cluster 2 and 4 might still make generalizability difficult. Furthermore, the second limitation was that this study did not use all the original ScratchJr Project Rubric features due to missing values. Limited focus on commonly used project concepts might lead to inaccurate conclusions on students' learning. The third limitation was that we assessed the projects without observing students' work process. External factors may impact students' completion, such as influence from teachers and friends. For example, advanced blocks that children used might be suggestions from teachers.

In future studies, we would like to increase the sample size by collecting more projects. Moreover, we are interested in collecting and analyzing students' demographics and teachers' information as the student's grade level, gender, or race, as well as teachers' experiences may impact the student's clusters. Lastly, qualitative class observation is essential to truly understand students' competencies when using the coding blocks.

Acknowledgment

We highly appreciate the support from the project grading team, and the members of the DevTech Research Group. Additionally, we would like to recognize Kyle Monahan and Saran Liukasemsarn for their recommendations and expertise in data science.

References

[1] P.-N. Chou, "Using ScratchJr to Foster Young Children's Computational Thinking Competence: A Case Study in a Third-Grade Computer Class," *J. Educ. Comput. Res.*, p. 073563311987290, Sep. 2019, doi: 10.1177/0735633119872908.

[2] S. Çiftci and A. Bildiren, "The effect of coding courses on the cognitive abilities and problem-solving skills of preschool children," *Comput. Sci. Educ.*, vol. 30, no. 1, pp. 3–21, Jan. 2020, doi: 10.1080/08993408.2019.1696169.

[3] A. Strawhacker, M. Lee, and M. U. Bers, "Teaching tools, teachers' rules: exploring the impact of teaching styles on young children's programming knowledge in ScratchJr," *Int. J. Technol. Des. Educ.*, vol. 28, no. 2, pp. 347–376, Jun. 2018, doi: 10.1007/s10798-017-9400-9.

[4] L. P. Flannery, B. Silverman, E. R. Kazakoff, M. U. Bers, P. Bontá, and M. Resnick, "Designing ScratchJr: support for early childhood learning through computer programming," in *Proceedings of the 12th International Conference on Interaction Design and Children - IDC '13*, New York, New York, 2013, pp. 1–10. doi: 10.1145/2485760.2485785.

[5] A. Sullivan and A. Strawhacker, "Screen-Free STEAM: Low-Cost and Hands-on Approaches to Teaching Coding and Engineering to Young Children," in *Embedding STEAM in Early Childhood Education and Care*, C. Cohrssen and S. Garvis, Eds. Cham: Springer International Publishing, 2021. doi: 10.1007/978-3-030-65624-9.

[6] M. U. Bers, C. González-González, and M. B. Armas–Torres, "Coding as a playground: Promoting positive learning experiences in childhood classrooms," *Comput. Educ.*, vol. 138, pp. 130–145, Sep. 2019, doi: 10.1016/j.compedu.2019.04.013.

[7] S. Papavlasopoulou, M. N. Giannakos, and L. Jaccheri, "Exploring children's learning experience in constructionism-based coding activities through design-based research," *Comput. Hum. Behav.*, vol. 99, pp. 415–427, Oct. 2019, doi: 10.1016/j.chb.2019.01.008.

[8] D. J. Portelance, A. L. Strawhacker, and M. U. Bers, "Constructing the ScratchJr programming language in the early childhood classroom," *Int. J. Technol. Des. Educ.*, vol. 26, no. 4, pp. 489–504, Nov. 2016, doi: 10.1007/s10798-015-9325-0.

[9] M. U. Bers, L. Flannery, E. R. Kazakoff, and A. Sullivan, "Computational thinking and tinkering: Exploration of an early childhood robotics curriculum," *Comput. Educ.*, vol. 72, pp. 145–157, Mar. 2014, doi: 10.1016/j.compedu.2013.10.020.

[10] S. Papert, *Mindstorms: children, computers, and powerful ideas*. New York: Basic Books, 1980.

[11] J. C. Adams and A. R. Webster, "What do students learn about programming from game, music video, and storytelling projects?," in *Proceedings of the 43rd ACM technical symposium on Computer*

Authorized licensed use limited to: Boston College. Downloaded on October 05,2023 at 15:47:36 UTC from IEEE Xplore. Restrictions apply.

Science Education - SIGCSE '12, Raleigh, North Carolina, USA, 2012, p. 643. doi: 10.1145/2157136.2157319.

[12] "ScratchJr – DevTech Research Group," 2020. https://sites.tufts.edu/devtech/research/scratchjr/ (accessed Jan. 03, 2021).

[13] S. P. Rose, M. P. J. Habgood, and T. Jay, "An Exploration of the Role of Visual Programming Tools in the Development of Young Children's Computational Thinking," *Electron. J. E-Learn.*, vol. 15, no. 4, p. 13, 2017.

[14] A. Unahalekhaka and M. U. Bers, "Evaluating young children's creative coding: rubric development and testing for ScratchJr projects," *Educ. Inf. Technol.*, p. 21, 2022, doi: https://doi.org/10.1007/s10639-021-10873-w.

[15] R. Agus and S. Mohamad Samuri, "Learning Analytics Contribution in Education and Child Development: A Review on Learning Analytics," *Asian J. Assess. Teach. Learn.*, vol. 8, pp. 36–47, Dec. 2018, doi: 10.37134/ajatel.vol8.4.2018.

[16] C. Alonso-Fernández, A. Calvo-Morata, M. Freire, I. Martínez-Ortiz, and B. Fernández-Manjón, "Applications of data science to game learning analytics data: A systematic literature review," *Comput. Educ.*, vol. 141, p. 103612, Nov. 2019, doi: 10.1016/j.compedu.2019.103612.

[17] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning*, vol. 103. New York, NY: Springer New York, 2013. doi: 10.1007/978-1-4614-7138-7.

[18] E. Relkin, L. E. de Ruiter, and M. U. Bers, "Learning to code and the acquisition of computational thinking by young children," *Comput. Educ.*, vol. 169, p. 104222, Aug. 2021, doi: 10.1016/j.compedu.2021.104222.

[19] K. Brennan and M. Resnick, "New frameworks for studying and assessing the development of computational thinking," *Proc. 2012 Annu. Meet. Am. Educ. Res. Assoc.*, vol. 1, p. 25, 2012.

[20] J. Moreno-Leon, G. Robles, and M. Roman-Gonzalez, "Towards Data-Driven Learning Paths to Develop Computational Thinking with Scratch," *IEEE Trans. Emerg. Top. Comput.*, vol. 8, no. 1, pp. 193–205, Jan. 2020, doi: 10.1109/TETC.2017.2734818.

[21] J. Denner, L. Werner, and E. Ortiz, "Computer games created by middle school girls: Can they be used to measure understanding of computer science concepts?," *Comput. Educ.*, vol. 58, no. 1, pp. 240–249, Jan. 2012, doi: 10.1016/j.compedu.2011.08.006.

[22] S. Grover, S. Basu, and P. Schank, "What We Can Learn About Student Learning From Open-Ended Programming Projects in Middle School Computer Science," in *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, Baltimore Maryland USA, Feb. 2018, pp. 999–1004. doi: 10.1145/3159450.3159522.

[23] J. Salac and D. Franklin, "If They Build It, Will They Understand It? Exploring the Relationship between Student Code and Performance," in *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education*, Trondheim Norway, Jun. 2020, pp. 473–479. doi: 10.1145/3341525.3387379.

[24] "Coding as Another Language." https://sites.tufts.edu/codingasanotherlanguage/curricula/scratchjr/ (accessed Apr. 21, 2021).

[25] K. Krippendorff, "Computing Krippendorff's Alpha-Reliability," p. 12, 2011.

[26] L. E. de Ruiter and M. U. Bers, "The Coding Stages Assessment: development and validation of an instrument for assessing young children's proficiency in the ScratchJr programming language," *Comput. Sci. Educ.*, pp. 1–30, Jul. 2021, doi: 10.1080/08993408.2021.1956216.

[27] Á. M. Navarro and P. Moreno-Ger, "Comparison of Clustering Algorithms for Learning Analytics with Educational Datasets," *Int. J. Interact. Multimed. Artif. Intell.*, vol. 5, no. 2, p. 9, 2018, doi: 10.9781/ijimai.2018.02.003.

[28] "Data considerations for Kruskal-Wallis Test." https://support.minitab.com/en-us/minitab-express/1/help-and-how-to/modeling-statistics/anova/how-to/kruskal-wallis-test/before-you-start/data-considerations/ (accessed Jul. 28, 2021).

[29] T. V. Hecke, "Power study of anova versus Kruskal-Wallis test," *J. Stat. Manag. Syst.*, vol. 15, no. 2–3, pp. 241–247, May 2012, doi: 10.1080/09720510.2012.10701623.

[30] "Principal Components (PCA) and Exploratory Factor Analysis (EFA) with SPSS." https://stats.idre.ucla.edu/spss/seminars/efa-spss/ (accessed Jul. 31, 2021).

[31] M. Yoon, J. Lee, and I.-H. Jo, "Video learning analytics: Investigating behavioral patterns and learner clusters in video-based online learning," *Internet High. Educ.*, vol. 50, p. 100806, Jun. 2021, doi: 10.1016/j.iheduc.2021.100806.

[32] E. Aboagye, "Principal Component Analysis of Students' Academic Performance in Mathematics and Statistics," *Am. Based Res. J.*, vol. 5, no. 10, p. 10, 2016.

[33] A. Strawhacker and M. U. Bers, "What they learn when they learn coding: investigating cognitive domains and computer programming knowledge in young children," *Educ. Technol. Res. Dev.*, vol. 67, no. 3, pp. 541–575, Jun. 2019, doi: 10.1007/s11423-018-9622-x.

[34] S. Basu, "Using Rubrics Integrating Design and Coding to Assess Middle School Students' Open-ended Block-based Programming Projects," in *Proceedings of the 50th ACM Technical Symposium on Computer Science Education.*, 2019, p. 7.

[35] A. Unahalekhaka and M. U. Bers, "Taking coding home: analysis of ScratchJr usage in home and school settings," *Educ. Technol. Res. Dev.*, May 2021, doi: 10.1007/s11423-021-10011-w.

Authorized licensed use limited to: Boston College. Downloaded on October 05,2023 at 15:47:36 UTC from IEEE Xplore. Restrictions apply.