frontiers | Frontiers in Education

TYPE Original Research
PUBLISHED 16 August 2022
DOI 10.3389/feduc.2022.957739

OPEN ACCESS

EDITED BY
Jari Laru,
University of Oulu, Finland

REVIEWED BY
Stamatios Papadakis,
University of Crete, Greece
Stefinee Pinnegar,
Brigham Young University,
United States

*CORRESPONDENCE
Solomon Sunday Oyelere
solomon.oyelere@ltu.se

SPECIALTY SECTION
This article was submitted to Digital Learning Innovations, a section of the journal Frontiers in Education

RECEIVED 31 May 2022
ACCEPTED 06 July 2022
PUBLISHED 16 August 2022

CITATION
Oyelere SS, Agbo FJ and Sanusi IT (2022) Developing a pedagogical evaluation framework for computational thinking supporting technologies and tools.
Front. Educ. 7:957739.
doi: 10.3389/feduc.2022.957739

Developing a pedagogical evaluation framework for computational thinking supporting technologies and tools

Solomon Sunday Oyelere[1]*, Friday Joseph Agbo[2,3] and Ismaila Temitayo Sanusi[2]

[1]Department of Computer Science, Electrical and Space Engineering, Luleå University of Technology, Luleå, Sweden, [2]Faculty of Science and Forestry, University of Eastern Finland, Joensuu, Finland, [3]Computing and Data Science, Willamette University, Salem, OR, United States

Frameworks for the evaluation of technological instructional tools provide educators with criteria to assess the pedagogical suitability and effectiveness of those tools to address learners' needs, support teachers' understanding of learning progress, and recognize the levels of achievement and the learning outcomes of the students. This study applied secondary document analysis and case study to identify five pedagogical indicators for teaching and learning computational thinking, including technology, pedagogical approaches, assessment techniques, data aspect, and teacher professional development. Based on the pedagogical indicators, this study proposed a computational thinking pedagogical assessment framework (CT-PAF) aimed at supporting educators with a strategy to assess the different technological learning tools in terms of pedagogical impact and outcome. Furthermore, three case-study instructional tools for teaching CT in K-12 were analyzed for the initial assessment of CT-PAF. Scratch, Google Teachable Machine, and the iThinkSmart minigames were marched to the underpinning characteristics and attributes of CT-PAF to evaluate the framework across the instructional tools. The initial assessment of CT-PAF indicates that the framework is suitable for the intended purpose of evaluating technological instructional tools for pedagogical impact and outcome. A need for expanded assessment is, therefore, necessary to further ascertain the relevance of the framework in other cases.

KEYWORDS

computational thinking, machine learning, evaluation framework, instructional tools, Scratch, Google Teachable Machine, iThinkSmart minigames

Introduction

The recent stride for data-driven society and automation has had an enormous impact on modern society. The impact can be seen in all segments of our society, such as work, education, economy, commerce, and leisure, and currently, we may not have enough data to assess the extent of the societal effects. Besides, the societal

impact of the teaching and learning of computational thinking (CT) in the K-12 level of computing education is gradually gaining much-needed attention (Wong and Cheung, 2020). Knowledge of CT is crucial for children to gain an adequate understanding of how the world around them works and prepare for future roles in society, especially, since they are considered active users of the data-driven technologies (Toivonen et al., 2020). Most countries around the globe have included CT as part of the introduction to computer science education at the K-12 level (Hsu et al., 2019). According to Wing (2006), CT is a necessary skill for everyone. In addition to improved programming and data literacy skills, CT is a suitable medium to develop 21st-century skills (Lye and Koh, 2014; Agbo et al., 2019).

Although certain empirical evidence about the large-scale impact of CT pedagogical supporting technologies and tools is still absent in current literature, the interest and awareness to support K-12 pedagogy have gained a visible increase in the last decade (Burgett et al., 2015; Yadav et al., 2016; Kong et al., 2020; Huang and Looi, 2021). In a CT pedagogical framework proposed by Kotsopoulos et al. (2017), four key experiences were identified, which include unplugged, tinkering, making, and remixing. These pedagogical experiences reflect on teachers' choices of pedagogical strategies and decisions on tools that will enhance learning experiences. Besides, student-centered pedagogical strategies such as problem-solving, open-ended tasks, project-based learning, group work and collaboration, inquiry-based learning, challenge-based learning, blended learning, flexible learning, and peer-to-peer learning were commonly adopted by teachers (Bower et al., 2017; Tucker-Raymond et al., 2021). Teachers have maintained positive experiences and deeper learning experiences are usually a result of student-led learning experiences, which are based on the active and engaging learning processes, pedagogical strategies, and a conducive learning environment (Adler et al., 2004).

Computational thinking pedagogical strategies are not new. They are usually adapted pedagogies to suit the teaching and learning of CT. Supporting technologies and tools play a crucial role in the application of CT pedagogical strategy and improved learning experiences. However, there is a need to set a standard and assessment strategy in place to address future research to understand the effectiveness of CT pedagogical supporting technology and tools (Kitalo et al., 2019). This paper is an initial step to addressing the lack of a pedagogical evaluation framework for understanding the impact of supporting technology and tools used in CT education.

A variety of evaluation frameworks have been studied, however, there is a lack of certainty about their operability and suitability to different instructional tools and evaluation objectives and the extent to which they are appropriate for teaching and learning CT topics across contexts. Fronza et al. (2017) used an agile software engineering-based framework principles and practices to map CT activities, however, contextual factors may hamper the application of the instructional tools in a multidisciplinary context. Besides, Kotsopoulos et al. (2017) identified the training of teachers as a challenge in the application of supporting tools in the teaching of CT. In addition, Gouws et al. (2013) recommended the need to have a CT test to evaluate students' CT levels by the teachers. However, it is important not just to measure CT levels but also to provide a yardstick to support the teachers to make the right decisions regarding the instructional tools which are intended to facilitate CT instructions. Accordingly, this study develops a framework of pedagogical indicators (PIs) for CT to support the standardization of the pedagogical supporting technologies and tools of CT, particularly in computing education. The evaluation framework comprises three layers (goals, indicators, outputs) and five interwoven indicators that determine the application of the tools for relevant pedagogical purposes. The Computational Thinking Pedagogical Assessment Framework or CT-PAF (as it will be referred to from here on) endeavors to proffer a structure to analyze and understand the formal intangible design of pedagogical supporting tools relative to their perspicuous and embedded purposes in supporting the process of CT education. CT-PAF provides the initial stride to create a robust evaluation benchmark for teachers' decision-making about the instructional tools that are designed to enhance students' CT knowledge in K-12 contexts. The CT-PAF framework should not be assumed to be a conclusive and detached measurement instrument, nonetheless, as a proposition on how to construct the assessment of CT teaching aids in terms of their pedagogical impact and improving learning outcomes. Besides, this paper is meant to start a discussion on the creation of data-driven pedagogical technologies and tools that are meant to support the teaching and learning process in K-12 settings. Therefore, this work focuses on providing an answer to the following research question, what are the PIs to consider for developing a CT-PAF in the K-12 computing education context?

Literature review

Computational thinking is an age-long concept for problem-solving approach (Denning and Tedre, 2019). Although the definition of the term CT remains vague (Agbo et al., 2019), the importance of the definition as compared to the pedagogical contribution is being argued (Selby and Woollard, 2013). However, Wings defined the term CT as "an approach to solving problems, designing systems and understanding human behavior that draws on concepts fundamental to computing" (Wing, 2008, p. 3717). Other scholars attempt to contribute to the definition of CT since a consensus is not reached. For example, Denning (2007), Guzdial (2008), and Barr and Stephenson (2011) examined the term CT and gave various definitions based on their expert perspectives.

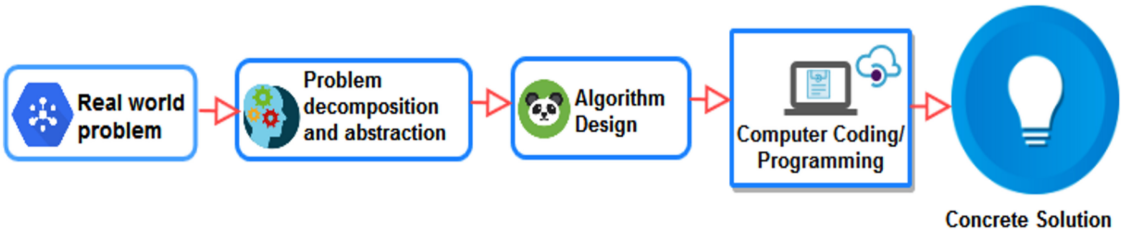

FIGURE 1
Features of computational thinking (adapted from Agbo et al., 2019).

Features of computational thinking

Zhong and Liao (2015) described the characteristic features of CT. Their study suggests that broadly, CT is a conceptualization and not a computer programming. Thus, the steps for introducing CT include problem decomposition, pattern recognition, problem abstraction, and algorithm design (Csizmadia et al., 2015; Agbo et al., 2019). As depicted in **Figure 1**, a real-world complex problem can be turned into one that is easily understood by undergoing these fundamental steps of CT. First, problem decomposition allows for breaking down big and complex problems into smaller and manageable units that can be easily understood and solved (Valenzuela, 2018).

In addition, CT requires the skill of pattern recognition; the ability to map similarities, differences, and patterns that exist in the small sizes of decomposed problems is essential to solving a complex one. For example, students who are able to recognize patterns could make predictions (Valenzuela, 2018). Also, the same problem-solving technique can be applied to solve problems with the same pattern. The third step in the CT approach is problem abstraction, which helps to filter out non-essential parts of a problem, leaving the important aspects that aid understanding and are easy to solve. The last stage of CT is the algorithm design that gives step-wise approaches which emanate from the first three stages of CT to point toward implementing the solution.

Computational thinking in science, technology, engineering, art, and mathematics education

Overall, the concept of CT has been connected to the Science, Technology, Engineering, Art, and Mathematics (STEAM) education (Yadav et al., 2016). As discussed by Wing, CT is a kind of scientific thinking that applies; technological concepts toward solving complex problems; engineering thinking to design a complex system that functions in a constrained real-world scenario; mathematical approach to solving a real-world problem (Wing, 2008). Nowadays, the need to embed CT into STEAM education is being researched (Jona et al., 2014; Weintrop et al., 2014; Yadav et al., 2016). For example, Swaid (2015) presented in a study, a framework for bringing CT to STEM disciplines through the Historically Black Colleges and Universities Undergraduate Program (HBCU-UP) project. This author organized six workshops with hands-on labs in order to introduce CT elements to the STEAM educators.

Similarly, Sengupta et al. (2018) conducted a phenomenological study where they reviewed a set of studies conducted in partnership with K-12 STEAM teachers and students. For example, the authors explored the opportunity to integrate computational modeling and programming in K-12 science classrooms. Their study suggests that "agent-based programming and modeling can help students overcome conceptual challenges in understanding linear continuity" (Sengupta et al., 2018, p. 16). Additionally, there are growing studies on the use of autonomous and programmable robotics (Atmatzidou and Demetriadis, 2016; Eguchi, 2016; Chalmers, 2018) and games (Wu and Richards, 2011; Kazimoglu et al., 2012; Leonard et al., 2016) designed to teach STEAM education to make students acquire the 21st-century skills required for nowadays employment.

Evaluation frameworks in computational thinking

The study on evaluation framework has been conducted in different spectrums of disciplines, even recently. For example, in smart cities and transportation (Yan et al., 2020), in the Internet of Things (IoT) to evaluate IoT platform development approaches (Fahmideh and Zowghi, 2020), Web technology to evaluate the accessibility of web tools (Alsaeedi, 2020), learning analytic tools (Scheffel et al., 2015; Vigentini et al., 2020), in mobile learning (Ozdamli, 2012). However, in the case of CT, limited studies have been recorded (Wong and Cheung, 2020). Our search revealed that, for about a decade, scholars have proposed frameworks to evaluate CT from broader perspectives (Moreno-León et al., 2015; Román-González et al., 2019).

Earlier studies have made attempts to develop frameworks for CT. These frameworks are however developed to focus on

specific aspects such as CT assessment technique, technology, or pedagogy approaches. Some of these studies include Román-González et al. (2017) which categorized CT assessment in K-12 as summative, formative-iterative, skill-transfer, perceptions-attitudes scales, and vocabulary assessment. Relatedly, Basso et al. (2018) discusses the non-technical skills (relational skills and cognitive life skills) that should be included in a comprehensive CT assessment framework. In the CT curriculum, Perković et al. (2010) proposed an interdisciplinary approach that allows the teaching of CT in different courses taught at the university level. This course-based CT framework was tested in three different aspects of courses, which include Scientific Inquiry Domain (SID) that introduces geospatial information processing; an Arts and Literature (AL) course about game design; and an animation course that focuses on 3-D modeling for gaming. Similarly, Gouws et al. (2013), developed a CT framework to be used for planning and evaluation of CT materials. The authors harnessed the characteristic features of CT such as transformation, abstraction, pattern recognition, and algorithm, to underpin their framework. According to the authors, a case study with Light-Bot shows that the CT framework is able to evaluate a CT resource by highlighting its strength and weakness.

Although some of these studies tried to evaluate CT based on the fundamental attributes regarding its objective of teaching problem-solving skills to students (Papadakis, 2021, 2022), there is a need to investigate how CT tools are performing in terms of providing the objectives for which it is developed. That is, paying close attention to the pedagogical aspect of the CT tools by mirroring it through a set of quality indicators to adjudge its efficacy. This aspect of the evaluation process is still missing from the literature, and we intend to fill this gap by proposing a pedagogical evaluation framework (CT-PAF) of quality indicators for CT tools. To the best of our knowledge, no existing frameworks have aggregated the factors or categorically highlighted the five indicators we identified in our work.

Methodology

Data sources

To address the research question in this study, we collected articles from Scopus and ACM databases to understand the kinds of approaches for CT pedagogy and assessment techniques that already exist. Because Scopus is one of the giant databases containing a huge number of scientific articles (Pranckutė, 2021) from the field of science including computer science (Agbo et al., 2021c), we consider it one of the useful sources to collect the data. In addition, other authors have argued that Scopus contains more distinct records (Singh et al., 2021) that can allow researchers to have a quick overview of scientific papers published in a specific field. In addition, most of the computing education interventions are published in ACM journals and conferences, and it publishes one of the widely read monthly communications where innovations are showcased (Blackburn et al., 2019).

TABLE 1 Keyword search and article selection strategy.

Source	Search string	Outcome
Scopus	(TITLE-ABS-KEY ("computational thinking") AND TITLE-ABS-KEY ("assessment framework" OR "evaluation framework"))	16
ACM	[All: "computational thinking"] AND [[All: "assessment framework"] OR [All: "evaluation framework"]]	32

Data search, inclusion, and exclusion procedure

For the data search, two main categories of keywords were used. The first is "computational thinking" and the second is "assessment framework OR evaluation framework." These keywords were combined in the search menus of the databases as presented in **Table 1**. In the Scopus database, the search keywords were framed to query the fields containing article title, abstract, and keywords, whereas, in the ACM database, the query consists of all the fields.

The procedure used in the data collection and screening includes the four-phased activities of the PRISMA (preferred reporting items for systematic reviews and meta-analysis) by Moher et al. (2009) presented in **Figure 2**. The inclusion criteria include articles that were published in peer-reviewed outlets including journals and conferences, and articles written in the English language. Furthermore, we manually skimmed through the title and abstract of each article to judge its relevance and excluded articles we deem irrelevant to this study in terms of its focus. For example, some of the articles that focused on the evaluation of computer science education in general but not CT, in particular, were removed. In addition, articles that carried out CT assessment in other forms aside from using a concrete tool were removed. Besides, we also removed duplicate articles found in the outcome from the two databases.

After the inclusion and exclusion criteria were applied, the data reviewed to gain insight into how CT assessments were conducted were 6. These articles guided the formulation of the CT-PAF PIs, which consist of the technology, pedagogy, assessment technique, data, and professional development. The authors read through each of the identified articles to understand its focus in terms of how evaluation of CT was conducted and to identify relevant indicators. Further information on how the analysis of these articles were conducted are presented in section "Findings: Toward an evaluation

TABLE 2 Summary of articles according to indicators in CT-PAF.

Article	Summary	Technology: instructional tools	Pedagogical approaches	Assessment techniques	Data aspects	Teacher professional development
Fronza et al., 2017	The paper presented a CT teaching and assessment framework focused on K-12 context.	Scratch, mind map, storyboard and feasibility table	Agile software engineering method	Project analysis, artifact-based interviews and summative examination	Not mentioned	Not mentioned
Kwon et al., 2021	Develop and evaluate an evaluation framework that reveals leaners' problem-solving competency based on Bebras computing challenge.	Scratch projects and Bebras Computing Challenge – tool to evaluate problem-solving	Problem-solving technique and Bebras computing challenge	Qualitative content analysis method was used to interpret the quality of the solutions. Four levels of CT were identified.	Not mentioned	Not mentioned
Gouws et al., 2013	Developed a CT framework for planning, preparing and evaluating CT materials.	Light-Bot, an educational game	Game-based learning approach	Computational thinking score	Not mentioned	Not mentioned
Fagerlund et al., 2020	An empirical study to assess 4th grade students' CT.	Scratch	Project-based, teacher-led demonstration, tutorial method,	Rubrics revision, analysis of programming contents, and interpreting conceptual encounters with CT	Not mentioned	Not mentioned
Knie et al., 2022	The article presented the experiences of integrating CT into a blended learning in-service training program for STEM teachers at secondary level schools.	Media Portal for STEM teachers, with additional teaching material such as instructions for the experiments of *Experimento* as well as other worksheets, graphics, and interactive media for STEM lessons, The Maze, coding game using a block-based programming environment, and microcontroller board Arduino	inquiry-based learning in the classroom, online self-learning modules, blended learning format,	Not mentioned explicitly, but the teachers experience, and perception of the training were evaluated through questionnaire-based quantitative survey and repeated measures.	Measured pH data from the Arduino microcontroller board	Professional development program *Experimento* targeting secondary level teachers
Kadijevich, 2019	Theoretical article that summarizes the concepts of CT and examine the use of data modeling approach for cultivating CT practice.	Interactive displays such as charts (dashboards), specifically, Zoho analytics	Modeling approach	Applying the use-modify-create path, and analysis of students' project portfolios	Presented data on interactive charts and the summary of the results. Besides, data modeling cycle was mapped to CT concepts and practices	Professional development was recommended to support teachers' important data modeling activities based on their skills and awareness of potential challenges in this modeling. In addition, professional development may support teachers in applying specific learning paths and the outcome assessment.

10.3389/feduc.2022.957739

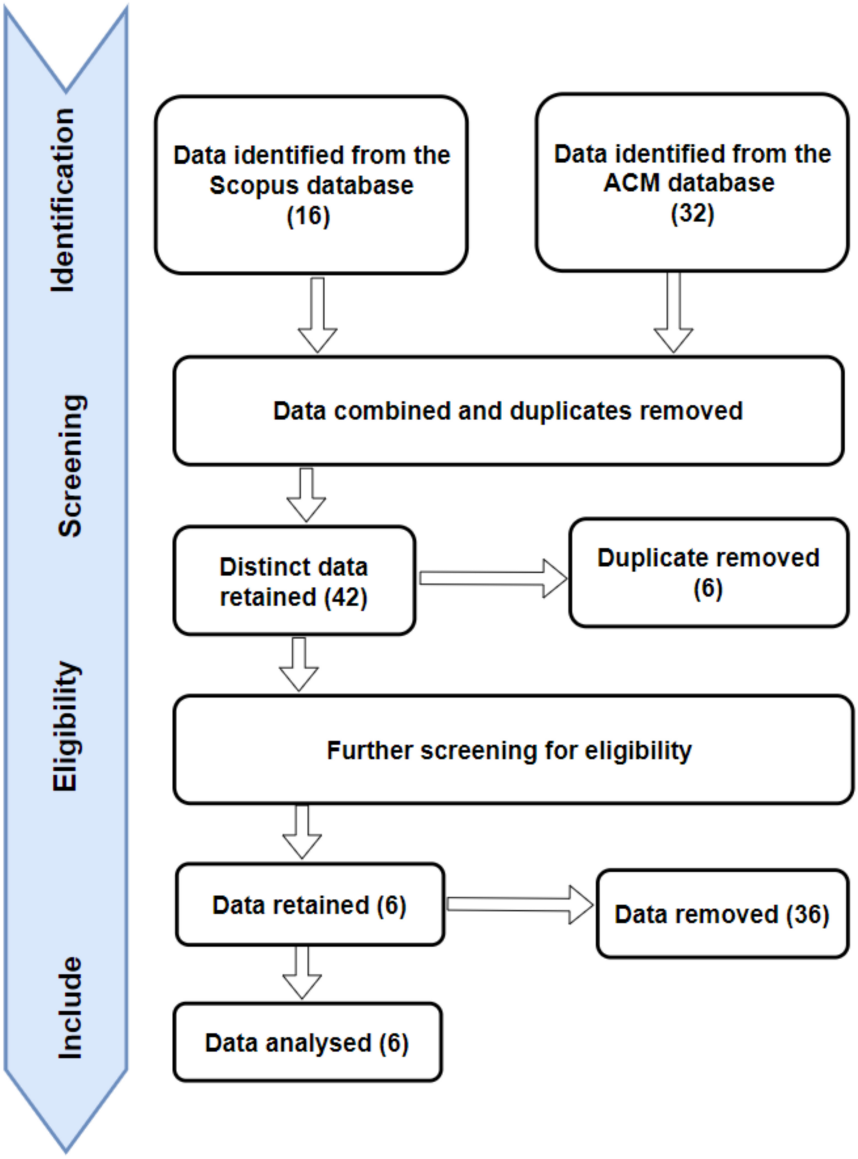

FIGURE 2
PRISMA diagram including the data collection and screening procedure (Moher et al., 2009).

framework for computational thinking supporting technology and tools", Table 2.

Findings: Toward an evaluation framework for computational thinking supporting technology and tools

This study examined the relevant indices to create an assessment framework useful for the evaluation of technological instructional tools in the context of CT in computing education. The evaluation frameworks of technological instructional tools offer educators criteria to assess the tool's suitability to support the intended learning outcome, facilitate learning activities, recognize learners' needs, and levels of achievement, facilitate reflection, and overall improve learning outcomes. Several frameworks are designed and developed for evaluating different technological learning platforms (Scheffel et al., 2015; Vigentini et al., 2020), tools (Park et al., 2010; Vigentini et al., 2020), and specifically pedagogies for CT (Kotsopoulos et al., 2017). Though studies have proposed the framework of CT, to the best of our knowledge, the development of a pedagogical framework to assess CT tools has not been established yet which is the aim of this study. Based on the literature review process outlined in

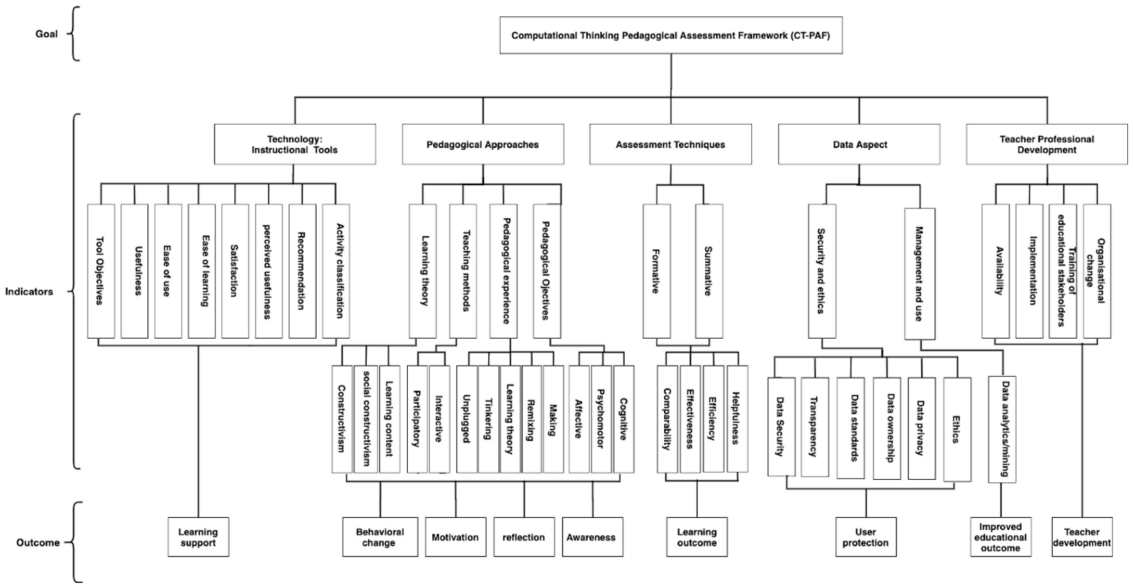

FIGURE 3
Computational thinking pedagogical assessment framework (CT-PAF).

section "Data search, inclusion, and exclusion procedure", the articles identified (**Table 2**) in which this study is focused were examined and relevant indicators were identified and framed into the CT assessment framework. Our framework coined CT-PAF presented in **Figure 3** comprises three layers which include goals, indicators, outcomes; and five indicators the layers (technology, pedagogical approaches, assessment techniques, data aspect, and teacher professional development, TPD).

Goals

Evaluation frameworks in educational scenarios enable a scientific and systematic approach to the assessment of important components of pedagogical processes. An effective and efficient instructional design process usually begins with defining the intended goals in a bid to maximize the student's learning experiences Czerkawski (2014). The overarching goal of the assessment of CT is needed to understand the expectation of the educational stakeholders and create the much-needed synergy among the different PIs in the learning ecosystem. Usually, CT frameworks are rooted in constructivism theory, which holds the view that learning entails reconstruction rather than direct transmission of knowledge Papert (1980) and Papert and Harel (1991); and the social constructivism viewpoint consider social group constructing knowledge for each other Vygotsky (1978). Honebein (1996) opined that instructional designers of constructivist learning environments should consider seven pedagogical goals which inform essential characteristics for the goals in CT-PAF.

Indicators

The following section will highlight the indicators of the assessment framework.

Technology: Instructional tools

Innovations around novel technology development have been the cornerstone of any changes in the education sector. The instructional tools in the CT-PAF depict the technologies, instructional materials, and tools to facilitate the teaching and learning of CT. Based on our literature review, we found the following indicators are necessary to consider when assessing the pedagogical relevance of the technologies, instructional materials, and supporting tools: the intended purpose of the tool (referred to as the tool's objectives), ease of use, ease of learning, satisfaction, perceived usefulness, recommendation, and activity classification. For example, educational tools must have a purpose that is expected to be derived from them and should support learning objectives which are usually derived from Bloom's revised taxonomy of educational objectives (Krathwohl, 2002). The ease of use (EOU) of a system has been regarded as "the degree to which a person believes that using a particular system would be free of effort" (Davis, 1989). This study describes the ease of learning as the rate at which learning through CT tools is achieved without much effort. Lu (2014) study signifies that the perceived usefulness signifies the individual's readiness to use information systems as a supporting tool. This, however, suggests that the usefulness of the learning tools can lead to satisfaction and critical reflection while

TABLE 3 Evaluation of Scratch based on CT-PAF.

CT-PAF indicators			Evaluation and remarks
Technology: Instructional Tools	Tool's objectives		Scratch is an open-source software developed by MIT Media Lab to support young people to learn to "think creatively, reason systematically, and work collaboratively" (Resnick et al., 2009). As studies that explore Scratch for CT education increases, scholars and educators are to ensure that the core objectives of Scratch are upheld. That is to say that Scratch should provide teachers with the training on blending CT and CS into their classroom; bring CT into K-12 domain to gain computing/programming skills through visualized (drag and drop) approach; build students to gain algorithmic and problem-solving skills. The CT-PAF perspective regarding instructional tools such as Scratch must meet the core purpose for which it was originally created.
	Ease of use		Scratch is easy for children to use (Souza and Bittencourt, 2018), since it is a block-based playful environment developed to make programming fun. Learners are able to effortlessly drag and drop blocks of codes without necessarily writing lines of codes. Some empirical studies may have validated ease of use of Scratch. For example, Bean et al. (2015). Based on the CT-PAF indicator, CT tools should generally allow learner to build problem-solving skills without much effort to use.
	Ease of learning		Evaluation of ease of learning and learning satisfaction of a programming tool or problem-solving intervention among students is critical to the overall outcome of such tool: For instance, in the case of Scratch, Lai and Lai (2012) examined students' understanding for using Scratch and revealed a positive outcome. This is one of the ways of assessing an educational tool.
	Satisfaction		Users satisfaction in an educational tool such as Scratch is important. Otherwise, the objective of motivating students to embrace such a tool cannot be totally achieved. To create enthusiasm for continues learning, Costa et al. (2016) conducted a language learning study with foreign children using Scratch and their findings were positive.
	Perceived usefulness		An experimentation of visual programming with Scratch in school indicated perceived usefulness of the tool to be high (Sáez-López et al., 2016).
	Recommendation		According to Estevez et al. (2019), the result from a study conducted on teaching high school student AI using Scratch suggests that visual programming tools such as Scratch holds high potential of impacting understanding to students.
	Activity classification		In Scratch project, it is easy to classify activities according to group of participants such as gender, age, grade level, prior knowledge level etc. Empirical studies usually develop data collection instrument through this easy approach.
Pedagogical Approaches	Learning theory		Research shows that constructionism, which is a design-oriented theory is suitable for visual programming tool such as Scratch (Peppler and Kafai, 2007). This theory places learners in designer role as expected if a tool must have relevance in teaching CT and ML.
	Teaching methods		Project based learning methodology has been adopted in study that utilized Scratch as demonstrated recently by Husna et al. (2019) and Plaza et al. (2019). Both studies indicate positive outcome from the theoretical and pedagogical perspectives.
	Pedagogical experience		Pedagogical experiences are usually measured by utilizing instruments such as "Did you have chances to select the learning methods that were suitable to you?". For instance, Tsukamoto et al. (2016) adapted this approach to measure pedagogical experience of participants.
	Pedagogical objectives		Overall objective of Scratch is specifically to help young students develop 21st century learning skills such as computational thinking, problem-solving skills, cognitive thinking, through creativity, collaborations, and other relevant pedagogical approaches that is suitable for the K-12.
Assessment Techniques	Formative and Summative	• Comparability • Effectiveness • Efficiency • Helpfulness	Research shows that assessment technique of study that utilized Scratch yields positive outcome of participants' performance. For example, Park and Shin (2019) reported an evaluation study where they compared Scratch and App Inventor. Although both are computational thinking tools, however, their result shows Scratch to score high on the average regarding efficiency, effectiveness, etc. compared to App inventor.
Data Aspect	Security and ethics	• Data Security • Transparency • Data standards • Data ownership • Data privacy Ethics	Security concerns of Scratch seems not to be a major discussion among scholars. With the potential for cloud computing and online accessibility, Scratch users should have a concrete framework for data security, standards, and framework. For instance, user's authentication, management of user's credentials and learning performance data, ownership of millions of data from projects shared online, and many other data sources should form critical concern for the developers of Scratch. Comparing with similar tool such as ML, Scratch need to improve its process regarding data security and ethics. Also, as recommended by Almutairy et al. (2019), Scratch need to develop a monitoring control and virtualization security policy of its user.
	Management and use	Data analytics/mining	Research in data analytics and mining for visual programming is still marginal. However, data analytics to identify level of creativity of learners using visual programming such as Scratch is recently emerging (Filvà et al., 2019; Kovalkov et al., 2020). Effort should continue in this area to make the available big data from experimentation of visual programming more meaningful.

(Continued)

TABLE 3 Continued

CT-PAF indicators		Evaluation and remarks
Teacher Professional Development	Availability	Scratch is available free of charge for educators, students, and trainers to download and used on the tablets, laptops, and desktop computers. It also has dozens of Sprites freely available in the latest version 3.0. Thanks to MIT media lab. Besides, plenty resources to support users are available online with no cost.
	Implementation	As reported by MIT, scratch has over 50 million projects already created and shared on Scratch website. These projects include animation, games, simulations, music, etc.
	Training of educational stakeholders	As an open source software, the inventor of Scratch has provided the opportunity for educational stakeholders to have access to the resources necessary to allow them teach students the 21st-century computing skills. This approach has been leveraged by researchers who are focused on training pre-service teachers (Papadakis and Kalogiannakis, 2019).
	Organizational change	The evolving use of technology to support learning and teaching keep the demand for training and retraining in teaching profession high. This imply that organizational change in education will remain evident provided the stakeholders in education are constantly engaging in designing program and policies to enhance teachers and learners. To this regard, Scratch has been explored in training teachers in developing more skills.

navigating through the learning environment (Honebein, 1996; Kotsopoulos et al., 2017). The outcome of the technical indicator is a kind of learning support that can be achieved with the learning tools.

Pedagogical approaches

In this study, we categorize the pedagogical approaches which focus on teaching and learning using the supporting tools into four main aspects (learning theory, teaching methods, pedagogical experience, and pedagogical objectives). Each of the categories further has indicators to measure the impact on learning. Learning theory includes constructivism and social constructivism Vygotsky (1978), Papert (1980), and Papert and Harel (1991); teaching methods include interactive and participatory methods (Vartiainen et al., 2018); pedagogical experience includes unplugged, tinkering, making, and remixing (Kotsopoulos et al., 2017); pedagogical objectives include cognitive, psychomotor and affective (Krathwohl, 2002; Lajis et al., 2018). For the learning theory, we choose constructivism as the theory most appropriate to guide the use of the tools (Agbo et al., 2021d). According to Papert's theory of constructionism, learners construct internal representations of their environment to develop knowledge (Papert, 1987). Besides, other Constructivist theorists posit that learning should be student-centered (Matthew et al., 2009) and encourages social interactions among students by taking part in constructing information actively for the fulfillment of the learning (Ozdamli, 2012). As regards the teaching method, interactive and participatory learning methods were selected (Vartiainen et al., 2018). The methods were purposefully chosen due to their peculiarity in that they create environments that provide children with opportunities to explore real-world phenomena in an interest-driven and inquiry-oriented manner (Vartiainen et al., 2018). For the pedagogical experiences, unplugged experiences are often first and foundational in learning CT because they require possibly the least amount of cognitive demand and technical knowledge (Kotsopoulos et al., 2017). One of the ways students can visualize and experience the process needed to complete a task is through unplugged experiences. The unplugged activities allow students to situate CT in a real life context (Curzon et al., 2014). During tinkering, students explore changes to existing objects and then consider the implications of the changes (Kotsopoulos et al., 2017) for example, tinkering is modifying existing computer programming code. Furthermore, to gain experiences, students must solve problems, make plans, select tools, reflect, communicate, and make connections across concepts. In other words, experiences can occur through computer programming. Lastly, remixing experiences as conceived by Kotsopoulos et al. (2017) involve sharing an object and embedding it within another object with the possibility to modify or adapt it in some way and/or to use it for substantially different purposes.

Assessment techniques

Assessment is an important indicator to consider in ascertaining whether a pedagogical process has been effective for the purpose it was designed. Assessment strategies may include self-assessment, diagnostic, or achievement tests, rating scales, and anecdotal techniques. According to Harrison (2010) self-assessment is an important element in the learning process. With self-assessment, students define suitable targets for their learning. While the different techniques exist, this study is interested in certain indicators for formative and summative assessment. Assessment helps in gauging the comparability, effectiveness, efficiency, and helpfulness of the pedagogical process as experienced by the learners. Van der Vleuten et al. (2017) described the assessment as a tool for learning. Assessment is seen to have value in helping to inform students' learning, instead of just judging how well they have learned

TABLE 4 Evaluation of Google Teachable Machine based on CT-PAF.

Indicators			Evaluation
Technology: Instructional Tools	Tool Objectives		It was created to help students, teachers, designers, and others learn about ML by creating and using their own classification models (Carney et al., 2020).
	Ease of use		Individuals use a Teachable Machine in ways that imply the tool enables learning and exploration.
	Ease of learning		From past study, it is evident that GTM can be easily learned, used, and taught, even by those without prior programming or ML experience (Carney et al., 2020).
	Satisfaction		In GTM, the default parameters are for training image classification models. All new image classification projects default to these parameters and most users gets satisfactory results.
	Perceived usefulness		The usefulness of the tool include: • A generalized, flexible-input interface for making ML classification models that can be easily learned and used without prior experience or expertise in ML or coding. • A set of product decisions that enable learning and experimentation for new users of ML. • An example of how content surrounding the interface allows people to learn ML concepts (Carney et al., 2020).
	Recommendation		A Canadian STEM education non-profit cites GTM as a "tool found particularly useful for introducing key concepts of AI" (Actua, 2019). Payne (2019) also recommends GTM as it was stated to introduce student to the concept of classification. Also, that by exploring GTM tool, students learn about supervised machine learning.
	Activity classification		It uses transfer learning, an ML technique, to find patterns and trends within the images or sound samples, and create a simple and easy classification model within seconds. With transfer learning, a user is able to add their own data and retrain a model on top of a previously trained base model that has learned a specific domain from a large dataset.
Pedagogical Approaches	Learning theory		It encourages learners-centered approach, such as constructivism.
	Teaching methods		GTM enables active learning method. Active learning is based on a theory of learning called constructivism, which emphasizes the fact that learners construct or build their understanding that can then apply to new contexts and problems. According to Carney et al. (2020), based on use of the tool amongst teachers and curricula, it was posited that GTM facilitates active learning of AI concepts by requiring students to interact with those concepts by making models themselves.
	Pedagogical experience		Unplugged experiences are often first and foundational in learning GTM as they require possibly the least amount of cognitive demand and technical knowledge (Kotsopoulos et al., 2017).
	Pedagogical objectives		The specific expected student learning outcomes were identified such as the ability to perform new skill after the activity.
Assessment Techniques	Formative and Summative	• Comparability • Effectiveness • Efficiency • Helpfulness	Teachable Machine builds on and extends related work. It provides an approachable yet well-featured interface for children and adults to create their own ML classification models through its website. It enables users to train classifiers for an arbitrary number of classes, provides data collection, classification, model training, and model evaluation in the same interface, and trains on-device Arising from the study of Vartiainen et al. (2020), with the use of GTM, it is recognized that very young children are able to engage in the exploration of machine learning based technologies
Data Aspect	Security and ethics	Data Security	Teachable Machine trains users' models within the browser on their device, with their own data, without that data leaving their computer.
		Transparency	GTM supports transparency as the web-based tools is open and easily accessible to everyone to work
		Data standards	In GTM, there is data standard which provides the guidelines through which users can confidently exchange information with the tool.
		Data ownership	GTM allows for a greater sense of data ownership without needing to worry about storing and saving large files, datasets, or models to the cloud.
		Data privacy	GTM helps users feel safe experimenting. The tool allows the users to exhibit sense of ownership by downloading the trained model locally using TensorFlow.js, which facilitate privacy and trust among users. In addition, GTM is flexible and has less permanent structure to play and experiment with machine learning, without needing to worry about storing and saving large files, datasets, or models to the cloud (Carney et al., 2020).
		Ethics	GTM provides ethical principles such as to help secure user data, allow users to export their data Provide users with clear information about how their information is used.
	Management and use	Data analytics/mining	With GTM, systems can learn to analyze data without being programmed. According to the authors and developers of GTM, the tool was created it to help students, teachers, designers, and others to learn about ML by creating and using their own classification models.

(Continued)

TABLE 4 Continued

Indicators		Evaluation
Teacher Professional Development	Availability	Training on how to teach AI in classroom to K-12 was held by Google.
	Implementation	The use of GTM has been explored by instructors, teachers and researchers to teach the concept of ML and computational thinking (Toivonen et al., 2020; Vartiainen et al., 2020; Zhang et al., 2020).
	Training of educational stakeholders	The training of educational stakeholders on the use of tools is important in supporting K-12 student in learning ML
	Organizational change	Building the capacity of educational stakeholder will affect the whole organization as all the stakeholders has added at least a skill to their skillsets.

in a given period of time (Houston and Thompson, 2017). Formative assessment was attached to improvement of learning progress, whereas summative assessment was attached to making judgments about achievement at the end of a course.

Data aspect

The pedagogy of CT cannot exist without data Shabihi and Kim (2021) and Eloy et al. (2022), and led students to cultivate CT practices through data manipulations Kadijevich (2019). Being a data-driven topic, an enormous amount of learning data and learner traces are released in learning activities. The set of instructional goals, especially the ones under study may not be measurable without paying attention to learning data. The pedagogical supporting tools provide an avenue for data collection and classification (Carney et al., 2020). The data aspect of the supporting tools in CT-PAF concerns transparency, data standards, data ownership, data privacy, and ethics (Barr and Stephenson, 2011; Dagiene and Stupuriene, 2016). These are considered in the framework to contribute to the effectiveness of the use of the tool in ensuring the overarching goals are achieved. Ensuring data privacy using the supporting technologies and tools will help users feel safe experimenting with their own data Tabesh (2017). Besides, data standards are best practices that determine how different types of data should be formatted and what metadata and documentation need to be included while data ownership gives a clear responsibility about both the possession and distribution of the information Alsancak (2020). Ownership gives the user the power as well as control, whereas information includes not just the ability to access, create, modify, package, derive benefit from, sell, or remove data, but also the right to assign these access privileges to others Ata and Yıldırım (2020).

Teacher professional development

The role of the instructor is to guide students in the assimilation and construction of information (Wheeler et al., 2008). With this in mind, TPD is necessary to guarantee successful teaching and the development of competencies in the use of the instructional tools to achieve the set pedagogical objectives. The indicators put forward in this study to assess TPD as it concerns the pedagogical impact of teachers' effort to support teaching CT are the availability, the implementation of training, training of educational stakeholders, and the organizational change that will lead to the overall development of the teacher.

Outcomes

The purpose of CT-PAF as an assessment framework is to obtain important evidence about both teacher and student performance and progress for using the supporting technologies in the process of teaching and learning CT. Besides, the outcome in the CT-PAF determines how the different indicators lead to the overall goal of the framework and permit exploration of the prospects to apply computing tools as a medium for teaching CT, Grover and Pea (2013). Particularly, students' interest to create an appropriate assessment of their learning process is monitored through the outcome of the CT-PAF framework.

Cases – Scratch, iThinkSmart minigames for teaching computational thinking, and Google Teachable Machine for teaching machine learning

Case study 1: Scratch for teaching computational thinking

Computational thinking is a skill that "involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science" (Wing, 2006). According to Bocconi et al. (2016) it is considered in many countries a key set of problem-solving skills that must be acquired and developed by today's generation of learners. Repenning et al. (2016) states that CT tools aim to minimize coding overhead by supporting users through three fundamental stages of the CT development cycle: problem formulation, solution expression, and solution (execution/evaluation). Among the several CT tools, Scratch

TABLE 5 Evaluation of iThinkSmart based on CT-PAF.

Indicators			Evaluation
Technology: Instructional Tools	Tool Objectives		The minigames integrated into the iThinkSmart application include Tower of Hanoi, River Crossing, and Mount Patti Treasure Hunt. These minigames are aimed to showcase how to visualize concepts of computational thinking such as problem decomposition, abstraction, and algorithmic design.
	Ease of use		An evaluation of the iThinkSmart with students shows that the tool is easy to use. However, some of the minigames were found to be challenging for students to complete within the regulated conditions – for example, the duration of a gameplay session constrained the students.
	Ease of learning		Evidence from the field experiment shows that some of the iThinkSmart minigames that are Multi-choice Questions (MCQ) are quite easy to learn, whereas others are difficult to unravel. The development of the iThinkSmart recognizes the impact of overloading too many learning outcomes, which can demotivate learners (Huang, 2011).
	Satisfaction		As an evolving tool whose features and requirements are iteratively modified based on the users' feedback, there is moderate satisfaction expressed by users of the iThinkSmart application.
	Perceived usefulness		The current version of the iThinkSmart was evaluated with students to understand how the intervention improve their perceived cognition, computational thinking competency, interest, and attitude toward future use of a similar tool.
	Recommendation		Students who played the iThinkSmart to gain computational thinking education agreed that the tool supported their learning and would recommend it to other students. Notwithstanding, ongoing testing and evaluation of the tool will provide more empirical evidence of users' experiences.
	Activity classification		iThinkSmart classifies players' computational thinking competency into three (high, satisfactory, and low) by using the objective distance model (Chaichumpa et al., 2021). Through this model integrated into the tool, players' learning progress can be enhanced to provide a personalized learning experience.
Pedagogical Approaches	Learning theory		Being a study conducted to focus on users, the design and development of the iThinkSmart application utilized both constructivism and experiential learning theories.
	Teaching methods		The iThinkSmart employed pedagogies such as game-based learning, problem-based learning, and storytelling to demonstrate the visualization of computational thinking concepts (Agbo, 2022).
	Pedagogical experience		Fundamentally, students co-designed minigames with researchers through a participatory process, through which they gained computational thinking skills. Besides, the immersive experience by players can provide an opportunity for integrating other pedagogies such as collaborative learning.
	Pedagogical Objectives		The specific expectation of students' learning objective is an improved computational thinking competency and problem-solving skills after playing the minigames in the iThinkSmart application.
Assessment Techniques	Formative and Summative	• Comparability • Effectiveness • Efficiency • Helpfulness	The objective distance model integrated into the iThinkSmart is a unique way of assessing learners during gameplay. According to Serrano-Laguna et al. (2018), assessment of players during gameplay using a methodology similar to the objective distance model that tracks players learning progress is a suitable way to assess game effectiveness.
Data Aspect	Security and ethics	Data Security	The data generated during the gameplay are stored in a secured cloud server that is password protected. iThinkSmart application mainly collects users' data as logs during the gameplay and generates a unique ID for each player.
		Transparency	iThinkSmart supports transparency by allowing players to see how s/he are progressing by displaying errors, rewards, hints, and other attributes that personalize the player's experience.
		Data standards	All data in iThinkSmart are standardized. For example, the satisfactory and total scores of each learning object represented in the minigames are defined by experts.
		Data ownership	Although, iThinkSmart is data-driven where the system collects data to assess players' competency, however, the data are only used for the purpose of assessment which guides the behaviors of the system by responding to the player based on learning progress and performance.
		Data privacy	Data privacy of players of the iThinkSmart minigames is assured by immersing the player in a virtual environment where they are shielded from the real world.
		Ethics	Ethical principles are upheld to ensure that players voluntarily opted to play the iThinkSmart minigames in order to gain computational thinking education.
	Management and use	Data analytics/mining	There are no data analytics or data mining techniques implemented in the iThinkSmart application yet.
Teacher Professional Development	Availability		iThinkSmart is an open-source application that is freely available for users to access. The tool can be installed on any android smartphone and played with a low-cost head-mounted display (HMD). Teachers can use this tool to introduce students to problem-solving and computational thinking education in their classrooms.

(Continued)

TABLE 5 Continued

Indicators		Evaluation
	Implementation	Requirements for implementing teaching using the iThinkSmart application are mainly a smartphone and HMD. Because these technologies are affordable nowadays, it is possible for teachers to integrate the use of iThinkSmart minigames to supplement their teaching pedagogy.
	Training of educational stakeholders	Some nuances related to the use of virtual reality technology in the classroom can be envisaged that may cause, for example, anxiety among stakeholders. Therefore, training is required to create the venue for wider acceptance and use of the iThinkSmart application.
	Organizational change	The introduction of a new intervention can create organizational change. The iThinkSmart application provides an opportunity for an improvement in teaching methods by supplementing traditional teaching through the use of minigames. Reaction to this new tool could create organizational change, which can be positive or otherwise.

which Moreno-León et al. (2015) regarded as the most used programming language in primary and secondary education worldwide. Lye and Koh (2014) and Çatlak et al. (2015) described Scratch as a usable tool in teaching programming or ensuring that students acquire CT skills. It was further described as a free web-based programming tool that allows the creation of media projects, such as games, interactive stories and animations, connected to young peoples' personal interests and experiences (Fagerlund et al., 2020).

Scratch is used in all levels of formal educational environments in K-12 schools (Meerbaum-Salant et al., 2013; Moreno-León et al., 2015) and even universities (Malan and Leitner, 2007) worldwide. Studies have explored the use of scratch in teaching CT in K-12. The study by Moreno-León et al. (2015) held workshops with students in the range from 10 to 14 years in 8 schools, involving over 100 learners. The result shows that at the end of the workshop, students increased their CT scores and, consequently, improved their coding skills. Oluk and Korkmaz (2016) study compared 5th graders' scores obtained from Scratch projects developed in the framework of Information Technologies and Software classes. A high-level significant relationship was observed between students' programming skills with Scratch and their CT skills. Scratch, besides supporting teachers in the evaluation tasks, is to act as a stimulus to encourage students to keep on improving their programming skills (Moreno-León et al., 2015). This study evaluates Scratch as a case study by measuring the tool in line with CT-PAF indicators (see **Table 3**).

Case study 2: Google Teachable Machine for teaching machine learning

According to Denning and Tedre (2019), machine learning (ML) can be considered a vital part of future computational skills. As a result, Mariescu-Istodor and Jormanainen (2019) opined that it is justifiable to include ML education as part of CT teaching agenda at the K-12 context. The concepts of ML and CT are emerging and essential in teaching with technology (Zhang et al., 2020). Zhang et al. (2020) designed a workshop with a teachable machine to teach CT using experiential learning models.

Teachable Machine is a web-basedtool for creating custom ML classification models without specialized technical expertise (Carney et al., 2020). There are several related tools such as machine learning for Kids and scratch nodes ML, and both tools works in similar fashion as Google Teachable Machine. For example, machine learning for kids is an educational tool that guides children through ML training (Lane, 2020); whereas scratch nodes ML enables children to create gesture classification models that integrate with Scratch (Agassi et al., 2019). There are also interactive machine learning (IML) tools such as Wekinator (Fiebrink, 2011) for creative practice. IML tools mostly target novice users without specific technical or domain expertise (Carney et al., 2020) such as crayons used for image classification (Fails and Olsen, 2003).

Carney et al. (2020) stated that teachable machine (TM) was created to help students, teachers, designers, and others learn about ML by creating and using their own classification models. Educators have found TM useful to introduce concepts of AI (UBC Geering Up, 2019). Payne (2019) has also used TM to teach AI Ethics Education, explaining concepts of bias, supervised learning in her MIT Curriculum. In another study, Carney et al. (2020) explored TM and found that it facilitates active learning of AI concepts. While administrators, educators, and students stated that there is a dearth of tools and activities to support active learning in AI. TM has also been explored in higher education, For instance, Shi (2019) used it to teach her machine learning for the Web students the basics of ML classification, and her students used it to make their own projects. Carney et al. (2020) used TM to explore how grad students from diverse disciplines can apply ML to their own domain. According to Carney et al. (2020), outcome from their study suggest that the tool can be useful not only to learn ML concepts but also as a resource for students' creative projects, even with no prior ML experience. **Table 4** presents the evaluation of GTM in line with CT-PAF.

Case study 3: iThinkSmart minigames

Acquiring CT skills through game-based learning is found to be a relevant approach for augmenting teaching and learning for

both K-12 and college students (Agbo et al., 2021a,c; Hooshyar et al., 2021). In the last one-decade, educational minigames are increasingly utilized in the classroom to motivate students and enhance their learning experience (Huang, 2011; Van Borkulo et al., 2011; Agbo et al., 2021a). The iThinkSmart is a virtual reality application consisting of minigames (Tower of Hanoi, River Crossing, and Mount Patti Treasure Hunt) developed to facilitate CT education and problem-solving skills (Agbo et al., 2021b; Agbo, 2022). These minigames provide knowledge of CT by allowing players to visualize concepts such as problem decomposition, abstraction, and algorithmic design through interaction with the game elements. For example, the concept of divide and conquer and problem decomposition is visualized when playing the River Crossing puzzle, and players gain problem-solving skills by playing the game to unravel the puzzle. The evaluation of the iThinkSmart minigames through lens of CT-PAF is demonstrated in **Table 5**.

Conclusion

This study utilized secondary document analysis and case studies to identify five PIs for teaching and learning CT. The indicators include technology, pedagogical approaches, assessment techniques, data aspect, and TPD. Based on the PIs, a CT-PAF was proposed which aimed at supporting educators with strategies to assess the different technological learning tools in terms of pedagogical impact and outcome. Initial assessment of the framework was carried out with three case-study which include Scratch, Google Teachable Machine, and the iThinkSmart minigames. The evaluation of Scratch as a case study of visual programming tool for acquiring CT skills revealed that aside from the data aspects, it considerably fits into CT-PAF. While the pedagogical design and application in training seem to be well explored, the aspect of data security, standards, privacy, and ethics lacks sufficient evidence to show strong compliance with CT-PAF. However, it is worthy to note that the accessibility and usability features of Scratch are a positive development. Evaluating GTM based on CT-PAF shows that the technological learning tool fulfills the objective for which it was designed. This is ascertained in line with the indicators employed as seen in **Table 4** but for the TPD which is an area that needs more attention as it relates to the passage of instruction through GTM. While more empirical evidence is needed to understand the potential of GTM and its pedagogical impact, some findings exist from recent studies. Toivonen et al. (2020) study found out that Google Teachable Machine is a feasible tool for K–12 education. Relatedly, von Wangenheim et al. (2020) developed an introductory course to teach the basics of ML and GTM was useful in teaching ML concepts. The research of Vartiainen et al. (2020) also shows that young children can engage in the exploration of machine learning-based technologies. The findings from the literature suggest that GTM supports teaching and learning. Regarding the evaluation of iThinkSmart minigames based on CT-PAF, the integration of learners' assessment technique within the gameplay, which shows players' CT competency seems a strong point among all the indicators. In addition, the evaluation shows that the pedagogical objectives of the minigames – hosted in the iThinkSmart application – were clearly defined to address specific learning goals of CT concepts. Contrarily, CT-PAF indicators such as data aspects that require integration of data analytics or data mining have not been explored in the iThinkSmart application. In addition, the tool still lacks an interface for teachers to explore their CT modules as minigames. Therefore, further improvement of the tool is required as exposed by its evaluation based on CT-PAF indicators in **Table 5**.

In conclusion, CT-PAF is the first step to implementing a robust assessment yardstick to help teachers in decision-making regarding instructional tools that are intended to improve the learning outcome within K-12 teaching contexts. The initial assessment of CT-PAF indicates that the framework is suitable for the intended purpose of evaluating technological instructional tools for pedagogical impact and outcome. Future research is necessary to assess and ascertain the relevance of the framework using other tools such as Wekinator, Lego, Blockly, and Lightbot, among others. In addition, an empirical study is needed to further validate the framework.

Data availability statement

The original contributions presented in this study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

All authors listed have made a substantial, direct, and intellectual contribution to the work, and approved it for publication.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

References

Actua (2019). *Bringing AI into the Classroom*. Paris: Actua.

Adler, R. W., Whiting, R. H., and Wynn-Williams, K. (2004). Student-led and teacher-led case presentations: Empirical evidence about learning styles in an accounting course. *Account. Educ.* 13, 213–229. doi: 10.1080/0963928041000167662O

Agassi, A., Erel, H., Wald, I. Y., and Zuckerman, O. (2019). "Scratch nodes ML: A playful system for children to create gesture recognition classifiers," in *Proceedings of the Conference on Human Factors in Computing Systems*, New York, NY. doi: 10.1145/3290607.3312894

Agbo, F. J. (2022). *Co-designing a Smart Learning Environment to Facilitate Computational Thinking Education in the Nigerian Context*. Ph.D. thesis. Kuopio: University of Eastern Finland.

Agbo, F. J., Oyelere, S. S., Suhonen, J., and Adewumi, S. (2019). "A Systematic Review of Computational Thinking Approach for Programming Education in Higher Education Institutions," in *Proceedings of the 19th Koli Calling International Conference on Computing Education Research*, New York, NY. doi: 10.1145/3364510.3364521

Agbo, F. J., Oyelere, S. S., Suhonen, J., and Tukiainen, M. (2021c). Scientific production and thematic breakthroughs in smart learning environments: a bibliometric analysis. *Smart Learn. Environ.* 8, 1–25. doi: 10.1186/s40561-020-00145-4

Agbo, F. J., Yigzaw, S. T., Sanusi, I. T., Oyelere, S. S., and Mare, A. H. (2021d). "Examining theoretical and pedagogical foundations of computational thinking in the context of higher education," in *2021 IEEE Frontiers in Education Conference (FIE)*, Manhattan, NY. doi: 10.1109/FIE49875.2021.9637405

Agbo, F. J., Oyelere, S. S., Suhonen, J., and Laine, T. H. (2021a). Co-design of mini games for learning computational thinking in an online environment. *Educ. Inf. Technol.* 26, 5815–5849. doi: 10.1007/s10639-021-10515-1

Agbo, F. J., Oyelere, S. S., Suhonen, J., and Tukiainen, M. (2021b). "iThinkSmart: Immersive Virtual Reality Mini Games to Facilitate Students' Computational Thinking Skills," in *Proceedings of the Koli Calling'21: 21st Koli Calling International Conference on Computing Education Research*, New York, NY.

Almutairy, N. M., Al-Shqeerat, K. H., and Al Hamad, H. A. (2019). A taxonomy of virtualization security issues in cloud computing environments. *Indian J. Sci. Technol.* 12:3. doi: 10.17485/ijst/2019/v12i3/139557

Alsaeedi, A. (2020). Comparing web accessibility evaluation tools and evaluating the accessibility of webpages: proposed frameworks. *Information* 11:40. doi: 10.3390/info11010040

Alsancak, D. (2020). Investigating computational thinking skills based on different variables and determining the predictor variables. *Participat. Educ. Res.* 7, 102–114. doi: 10.17275/per.20.22.7.2

Ata, R., and Yıldırım, K. (2020). Analysis of the relation between computational thinking and new media literacy skills of first-year engineering students. *J. Educ. Multimed. Hypermed.* 29, 5–20.

Atmatzidou, S., and Demetriadis, S. (2016). Advancing students' computational thinking skills through educational robotics: a study on age and gender relevant differences. *Robot. Autonom. Syst.* 75, 661–670. doi: 10.1016/j.robot.2015.10.008

Barr, V., and Stephenson, C. (2011). Bringing computational thinking to K-12: What is Involved and what is the role of the computer science education community? *ACM Inroads* 2, 48–54. doi: 10.1145/1929887.1929905

Basso, D., Fronza, I., Colombi, A., and Pahl, C. (2018). "Improving assessment of computational thinking through a comprehensive framework," in *Proceedings of the 18th Koli Calling International Conference on Computing Education Research*, New York, NY. doi: 10.1145/3279720.3279735

Bean, N., Weese, J., Feldhausen, R., and Bell, R. S. (2015). "Starting from scratch: Developing a pre-service teacher training program in computational thinking," in *Proceedings of the 2015 IEEE Frontiers in Education Conference (FIE)*, Manhattan, NY. doi: 10.1109/FIE.2015.7344237

Blackburn, S. M., McKinley, K. S., and Xie, L. (2019). Author Growth Outstrips Publication Growth in Computer Science and Publication Quality Correlates with Collaboration. *arXiv* [Preprint].

Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., and Engelhardt, K. (2016). *Developing computational thinking in compulsory education – Implications for policy and practice; EUR 28295 EN*. Luxembourg: Publications Office of the European Union.

Bower, M., Wood, L. N., Lai, J. W., Howe, C., Lister, R., Mason, R., et al. (2017). Improving the computational thinking pedagogical capabilities of school teachers. *Austral. J. Teach. Educ.* 42, 53–72. doi: 10.14221/ajte.2017v42n3.4

Burgett, T., Folk, R., Fulton, J., Peel, A., Pontelli, E., and Szczepanski, V. (2015). "DISSECT: Analysis of pedagogical techniques to integrate computational thinking into K-12 curricula," in *Proceedings of the 2015 IEEE Frontiers in Education Conference (FIE)*, Manhattan, NY. doi: 10.1109/FIE.2015.7344241

Carney, M., Webster, B., Alvarado, I., Phillips, K., Howell, N., Griffith, J., et al. (2020). "Teachable Machine: Approachable Web-Based Tool for Exploring Machine Learning Classification," in *Proceedings of the 2020 CHI conference on human factors in computing systems*, Honolulu, HI. doi: 10.1145/3334480.3382839

Çatlak, Ş, Tekdal, M., and Baz, F. Ç (2015). The status of teaching programming with scratch: a document review work. *J. Instr. Technol. Teach. Educ.* 4, 13–25.

Chaichumpa, S., Wicha, S., and Temdee, P. (2021). Personalized learning in a virtual learning environment using a modification of objective distance. *Wireless Pers. Commun.* 118, 1–18. doi: 10.1007/s11277-021-08126-7

Chalmers, C. (2018). Robotics and computational thinking in primary school. *Int. J. Child Comput. Interact.* 17, 93–100. doi: 10.1016/j.ijcci.2018.06.005

Costa, S., Gomes, A., and Pessoa, T. (2016). Using Scratch to Teach and Learn English as a Foreign Language in Elementary School. *Int. J. Educ. Learn. Syst.* 1, 207–213.

Csizmadia, A., Curzon, P., Dorling, M., Humphreys, S., Ng, T., Selby, C., et al. (2015). *Computational Thinking–A Guide for Teachers*. Available online at: https://eprints.soton.ac.uk/424545/

Curzon, P., McOwan, P. W., Plant, N., and Meagher, L. R. (2014). "Introducing teachers to computational thinking using unplugged storytelling," in *Proceedings of the 9th workshop in primary and secondary computing education*, New York, NY. doi: 10.1145/2670757.2670767

Czerkawski, B. (2014). "Educational Objectives for Promoting Computational Thinking in E-Learning," in *Proceedings of the E-Learn: World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education*, Morgantown, WV.

Dagiene, V., and Stupuriene, G. (2016). Informatics concepts and computational thinking in K-12 education: a Lithuanian perspective. *J. Inf. Process.* 24, 732–739. doi: 10.2197/ipsjjip.24.732

Davis, F. D. (1989). Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Q.* 13, 319–340. doi: 10.2307/249008

Denning, P. J. (2007). Computing is a natural science. *Commun. ACM* 50, 13–18. doi: 10.1145/1272516.1272529

Denning, P. J., and Tedre, M. (2019). *Computational thinking*. Cambridge, MA: Mit Press. doi: 10.7551/mitpress/11740.001.0001

Eguchi, A. (2016). RoboCupJunior for promoting STEM education, 21st century skills, and technological advancement through robotics competition. *Robot. Autonom. Syst.* 75, 692–699. doi: 10.1016/j.robot.2015.05.013

Eloy, A., Achutti, C. F., Fernandez, C., and De Deus Lopes, R. (2022). A Data-Driven Approach to Assess Computational Thinking Concepts Based on Learners' Artifacts. *Inf. Educ.* 21, 33–54. doi: 10.15388/infedu.2022.02

Estevez, J., Garate, G., and Graña, M. (2019). Gentle introduction to artificial intelligence for high-school students using scratch. *IEEE Access* 7, 179027–179036. doi: 10.1109/ACCESS.2019.2956136

Fagerlund, J., Häkkinen, P., Vesisenaho, M., and Viiri, J. (2020). Assessing 4th grade students' computational thinking through scratch programming projects. *Inf. Educ.* 19, 611–640. doi: 10.15388/infedu.2020.27

Fahmideh, M., and Zowghi, D. (2020). An exploration of IoT platform development. *Inf. Syst.* 87:101409. doi: 10.1016/j.is.2019.06.005

Fails, J., and Olsen, D. (2003). "A design tool for camera-based interaction," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Honolulu, HI. doi: 10.1145/642611.642690

Fiebrink, R. A. (2011). *Real-time Human Interaction with Supervised Learning Algorithms for Music Composition and Performance*. Princeton: Princeton University. doi: 10.1145/1753846.1753889

Filvà, D. A., Forment, M. A., García-Peñalvo, F. J., Escudero, D. F., and Casañ, M. J. (2019). Clickstream for learning analytics to assess students' behavior with Scratch. *Future Generat. Comput. Syst.* 93, 673–686. doi: 10.1016/j.future.2018.10.057

Fronza, I., Ioini, N. E., and Corral, L. (2017). Teaching computational thinking using agile software engineering methods: a framework for middle schools. *ACM Trans. Comput. Educ.* 17, 1–28. doi: 10.1145/3055258

Gouws, L. A., Bradshaw, K., and Wentworth, P. (2013). "Computational thinking in educational activities: an evaluation of the educational game lightbot," in *Proceedings of the 18th ACM conference on Innovation and technology in computer science education*, New York, NY. doi: 10.1145/2462476.2466518

Grover, S., and Pea, R. (2013). Computational thinking in k–12: A review of the state of the field. *Educ. Res.* 42, 38–43. doi: 10.3102/0013189X12463051

Guzdial, M. (2008). Education Paving the way for computational thinking. *Commun. ACM* 51, 25–27. doi: 10.1145/1378704.1378713

Harrison, C. (2010). *Peer-and self-Assessment. Social and Emotional Aspects of Learning*. Amsterdam: Elsevier. doi: 10.1016/B978-0-08-044894-7.00313-4

Honebein, P. (1996). "Seven goals for the design of constructivist learning environments," in *Constructivist learning environments*, ed. B. Wilson (Englewood Cliffs, NJ: EducationalTechnology Publications), 17–24.

Hooshyar, D., Pedaste, M., Yang, Y., Malva, L., Hwang, G. J., Wang, M., et al. (2021). From gaming to computational thinking: an adaptive educational computer game-based learning approach. *J. Educ. Comput. Res.* 59, 383–409. doi: 10.1177/0735633120965919

Houston, D., and Thompson, J. N. (2017). Blending formative and summative assessment in a capstone subject:'It's not your tools, it's how you use them'. *J. Univ. Teach. Learn. Pract.* 14:2. doi: 10.53761/1.14.3.2

Hsu, Y.-C., Irie, N. R., and Ching, Y.-H. (2019). Computational thinking educational policy initiatives (CTEPI) across the globe. *TechTrends* 63, 260–270. doi: 10.1007/s11528-019-00384-4

Huang, W., and Looi, C. K. (2021). A critical review of literature on "unplugged" pedagogies in K-12 computer science and computational thinking education. *Comput. Sci. Educ.* 31, 83–111. doi: 10.1080/08993408.2020.1789411

Huang, W. H. (2011). Evaluating learners' motivational and cognitive processing in an online game-based learning environment. *Comput. Hum. Behav.* 27, 694–704. doi: 10.1016/j.chb.2010.07.021

Husna, A., Cahyono, E., and Fianti, F. (2019). The effect of project based learning model aided scratch media toward learning outcomes and creativity. *J. Innov. Sci. Educ.* 8, 1–7.

Jona, K., Wilensky, U., Trouille, L., Horn, M. S., Orton, K., Weintrop, D., et al. (2014). Embedding computational thinking in science, technology, engineering, and math (CT-STEM). *Paper Presented at the future directions in computer science education summit meeting*, Orlando, FL.

Kadijevich, D. M. (2019). "Cultivating Computational Thinking Through Data Practice," in *Empowering Learners for Life in the Digital Age. OCCE 2018. IFIP Advances in Information and Communication Technology*, Vol. 524, eds D. Passey, R. Bottino, C. Lewin, and E. Sanchez (Cham: Springer). doi: 10.1007/978-3-030-23513-0_3

Kazimoglu, C., Kiernan, M., Bacon, L., and MacKinnon, L. (2012). Learning programming at the computational thinking level via digital game-play. *Procedia Comput. Sci.* 9, 522–531. doi: 10.1016/j.procs.2012.04.056

Kitalo, K. H. M., Tedre, M., Laru, J., and Valtonen, T. (2019). *Computational Thinking in Finnish Pre-Service Teacher Education*. Los Angeles, CA: CoolThink@JC.

Knie, L., Standl, B., and Schwarzer, S. (2022). First experiences of integrating computational thinking into a blended learning in-service training program for STEM teachers. *Comput. Appl. Eng. Educ.* doi: 10.1002/cae.22529

Kong, S. C., Lai, M., and Sun, D. (2020). Teacher development in computational thinking: design and learning outcomes of programming concepts, practices and pedagogy. *Comput. Educ.* 151:103872. doi: 10.1016/j.compedu.2020.103872

Kotsopoulos, D., Floyd, L., Khan, S., Namukasa, I. K., Somanath, S., Weber, J., et al. (2017). A pedagogical framework for computational thinking. *Digit. Exp. Math. Educ.* 3, 154–171. doi: 10.1007/s40751-017-0031-2

Kovalkov, A., Segal, A., and Gal, K. (2020). "Inferring Creativity in Visual Programming Environments," in *Proceedings of the Seventh ACM Conference on Learning@ Scale*, New York, NY. doi: 10.1145/3386527.3406725

Krathwohl, D. R. (2002). A revision of Bloom's taxonomy: an overview. *Theory Pract.* 41, 212–218. doi: 10.1207/s15430421tip4104_2

Kwon, K., Cheon, J., and Moon, H. (2021). Levels of problem-solving competency identified through Bebras Computing Challenge. *Educ. Inf. Technol.* 26, 5477–5498. doi: 10.1007/s10639-021-10553-9

Lai, C. S., and Lai, M. H. (2012). "Using computer programming to enhance science learning for 5th graders in Taipei," in *Proceedings of the 2012 International Symposium on Computer, Consumer and Control*, Manhattan, NY. doi: 10.1109/IS3C.2012.45

Lajis, A., Nasir, H. M., and Aziz, N. A. (2018). "Proposed assessment framework based on bloom taxonomy cognitive competency: Introduction to programming," in *Proceedings of the 2018 7th International Conference on Software and Computer Applications*, New York, NY. doi: 10.1145/3185089.3185149

Lane, D. (2020). *Machine Learning for Kids*. San Francisco, CA: No Starch Press.

Leonard, J., Buss, A., Gamboa, R., Mitchell, M., Fashola, O. S., Hubert, T., et al. (2016). Using robotics and game design to enhance children's self-efficacy, STEM attitudes, and computational thinking skills. *J. Sci. Educ. Technol.* 25, 860–876. doi: 10.1007/s10956-016-9628-2

Lu, J. (2014). Are personal innovativeness and social influence critical to continue with mobile commerce?. *Internet Res.* 24, 134–159. doi: 10.1108/IntR-05-2012-0100

Lye, S. Y., and Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Comput. Hum. Behav.* 41, 51–61. doi: 10.1016/j.chb.2014.09.012

Malan, D. J., and Leitner, H. H. (2007). Scratch for budding computer scientists. *ACM Sigcse Bull.* 39, 223–227. doi: 10.1145/1227504.1227388

Mariescu-Istodor, R., and Jormanainen, I. (2019). "Machine Learning for High School Students," in *Proceedings of the 19th Koli Calling International Conference on Computing Education Research (Koli Calling '19)*, New York, NY. doi: 10.1145/3364510.3364520

Matthew, K. I., Felvegi, E., and Callaway, R. A. (2009). Wiki as a collaborative learning tool in a language arts methods class. *J. Res. Technol. Educ.* 42, 51–72. doi: 10.1080/15391523.2009.10782541

Meerbaum-Salant, O., Armoni, M., and Ben-Ari, M. (2013). Learning computer science concepts with scratch. *Comput. Sci. Educ.* 23, 239–264. doi: 10.1080/08993408.2013.832022

Moher, D., Liberati, A., Tetzlaff, J., Altman, D. G., and Prisma Group. (2009). Preferred reporting items for systematic reviews and meta-analyses: the PRISMA statement. *Ann. Internal Med.* 151, 264–269. doi: 10.7326/0003-4819-151-4-200908180-00135

Moreno-León, J., Robles, G., and Román-González, M. (2015). Dr. Scratch: automatic analysis of scratch projects to assess and foster computational thinking. *Rev. Educ. Dist.* 46, 1–23.

Oluk, A., and Korkmaz, Ö (2016). Comparing students' scratch skills with their computational thinking skills in terms of different Variables. *Online Submiss.* 8, 1–7. doi: 10.5815/ijmecs.2016.11.01

Ozdamli, F. (2012). Pedagogical framework of m-learning. *Procedia Soc. Behav. Sci.* 31, 927–931. doi: 10.1016/j.sbspro.2011.12.171

Papadakis, S. (2021). The impact of coding apps on young children Computational Thinking and coding skills. A literature review. *Front. Educ.* 6:657895. doi: 10.3389/feduc.2021.657895

Papadakis, S. (2022). Can Preschoolers Learn Computational Thinking and Coding Skills with ScratchJr? A Systematic Literature Review. *Int. J. Educ. Reform* doi: 10.1177/10567879221076077

Papadakis, S., and Kalogiannakis, M. (2019). Evaluating a course for teaching introductory programming with Scratch to pre-service kindergarten teachers. *Int. J. Technol. Enhanced Learn.* 11, 231–246. doi: 10.1504/IJTEL.2019.100478

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York, NY: Basic Books.

Papert, S. (1987). *A new opportunity for science education. NSF Grant Application*. Available online at: https://nsf.gov/awardsearch/showAward?AWD_ID=8751190 (accessed May 12, 2008).

Papert, S., and Harel, I. (1991). *Constructionism*. New York, NY: Ablex publishing corporation.

Park, J., Cho, W., and Rho, S. (2010). Evaluating ontology extraction tools using a comprehensive evaluation framework. *Data Knowledge Eng.* 69, 1043–1061. doi: 10.1016/j.datak.2010.07.002

Park, Y., and Shin, Y. (2019). Comparing the effectiveness of scratch and app inventor with regard to learning computational thinking concepts. *Electronics* 8:1269. doi: 10.3390/electronics8111269

Payne, B. H. (2019). *An Ethics of Artificial Intelligence Curriculum for Middle School Students*. Available online at: https://ec.europa.eu/futurium/en/system/files/ged/mit_ai_ethics_education_curriculum.pdf (accessed Oct 10, 2019).

Peppler, K. A., and Kafai, Y. B. (2007). From SuperGoo to Scratch: exploring creative digital media production in informal learning. *Learning Media Technol.* 32, 149–166. doi: 10.1080/17439880701343337

Perković, L., Settle, A., Hwang, S., and Jones, J. (2010). "A framework for computational thinking across the curriculum," in *Proceedings of the fifteenth annual conference on Innovation and technology in computer science education*, Çankaya. doi: 10.1145/1822090.1822126

Plaza, P., Sancristobal, E., Carro, G., Blazquez, M., García-Loro, F., Muñoz, M., et al. (2019). "STEM and educational robotics using scratch," in *Proceedings of the 2019 IEEE Global Engineering Education Conference (EDUCON)*, Manhattan, NY. doi: 10.1109/EDUCON.2019.8725028

Pranckutė, R. (2021). Web of science (Wos) and scopus: the titans of bibliographic information in today's academic world. *Publications* 9, 1–56. doi: 10.3390/publications9010012

Repenning, A., Basawapatna, A., and Escherle, N. (2016). "Computational thinking tools," in *Proceedings of the 2016 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, Manhattan, NY. doi: 10.1109/VLHCC.2016.7739688

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., et al. (2009). Scratch: programming for all. *Commun. ACM* 52, 60–67. doi: 10.1145/1592761.1592779

Román-González, M., Moreno-León, J., and Robles, G. (2017). "Complementary tools for computational thinking assessment," in *Proceedings of International Conference on Computational Thinking Education (CTE 2017)*, Victoria.

Román-González, M., Moreno-León, J., and Robles, G. (2019). "Combining assessment tools for a comprehensive evaluation of computational thinking interventions," in *Computational Thinking Education*, eds S. C. Kong and H. Abelson (Singapore: Springer), 79–98. doi: 10.1007/978-981-13-6528-7_6

Sáez-López, J. M., Román-González, M., and Vázquez-Cano, E. (2016). Visual programming languages integrated across the curriculum in elementary school: a two year case study using "Scratch" in five schools. *Comput. Educ.* 97, 129–141. doi: 10.1016/j.compedu.2016.03.003

Scheffel, M., Drachsler, H., and Specht, M. (2015). "Developing an evaluation framework of quality indicators for learning analytics," in *Proceedings of the Fifth International Conference on Learning Analytics and Knowledge*, Irvine, CA. doi: 10.1145/2723576.2723629

Selby, C., and Woollard, J. (2013). *Computational thinking: the developing definition*. Available online at: http://eprints.soton.ac.uk/id/eprint/356481 (accessed June 14, 2020).

Sengupta, P., Dickes, A., and Farris, A. V. (2018). "Toward a Phenomenology of Computational Thinking in K-12 STEM," in *Computational Thinking in STEM Discipline: Foundations and Research Highlights*, ed. M. S. Khine (Berlin: Springer). doi: 10.1007/978-3-319-93566-9_4

Serrano-Laguna, Á, Manero, B., Freire, M., and Fernández-Manjón, B. (2018). A methodology for assessing the effectiveness of serious games and for inferring player learning outcomes. *Multimed. Tools Appl.* 77, 2849–2871. doi: 10.1007/s11042-017-4467-6

Shabihi, N., and Kim, M. S. (2021). "Data-Driven Understanding of Computational Thinking Assessment: A Systematic Literature Review," in *Proceedings of the European Conference on e-Learning*, Hatfield.

Shi, Y. (2019). *Repository for the "Machine Learning for the Web"*. Available online at: https://github.com/yining1023/machine-learning-for-the-web (accessed July 13, 2022).

Singh, V. K., Singh, P., Karmakar, M., Leta, J., and Mayr, P. (2021). The journal coverage of Web of Science, Scopus and Dimensions: a comparative analysis. *Scientometrics* 126, 5113–5142. doi: 10.1007/s11192-021-03948-5

Souza, S. M., and Bittencourt, R. A. (2018). "Computer Programming Workshops with Playful Environments for Middle School Girls," in *Proceedings of the 2018 IEEE Frontiers in Education Conference (FIE)*, New York, NY. doi: 10.1109/FIE.2018.8659111

Swaid, S. I. (2015). Bringing Computational thinking to STEM education. *Procedia Manufact.* 3, 3657–3662. doi: 10.1016/j.promfg.2015.07.761

Tabesh, Y. (2017). Computational thinking: A 21st century skill. *Olymp. Inf.* 11, 65–70. doi: 10.15388/ioi.2017.special.10

Toivonen, T., Jormanainen, I., Kahila, J., Tedre, M., Valtonen, T., and Vartiainen, H. (2020). "Co-Designing Machine Learning Apps in K–12 With Primary School Children," in *Proceedings of the 2020 IEEE 20th International Conference on Advanced Learning Technologies (ICALT)*, New York, NY. doi: 10.1109/ICALT49669.2020.00099

Tsukamoto, H., Takemura, Y., Oomori, Y., Ikeda, I., Nagumo, H., Monden, A., et al. (2016). "Textual vs. visual programming languages in programming education for primary school children," in *Proceedings of the 2016 IEEE Frontiers in Education Conference (FIE)*, New York, NY. doi: 10.1109/FIE.2016.7757571

Tucker-Raymond, E., Cassidy, M., and Puttick, G. (2021). Science teachers can teach computational thinking through distributed expertise. *Comput. Educ.* 173:104284. doi: 10.1016/j.compedu.2021.104284

UBC Geering Up (2019). *The University of British Columbia*. Vancouver, BC: UBC Geering Up.

Valenzuela, J. (2018). *How to Develop Computational Thinkers*, Available online at: https://www.iste.org/explore/Computational-Thinking/How-to-develop-computational-thinkers (accessed June 14, 2020).

Van Borkulo, S., Van Den Heuvel-Panhuizen, M., Bakker, M., and Loomans, H. (2011). "One mini-game is not like the other: Different opportunities to learn multiplication tables," in *Poster presented at Joint Conference on Serious Games*, Berlin. doi: 10.1007/978-3-642-33814-4_9

Vartiainen, H., Tedre, M., and Valtonen, T. (2020). Learning machine learning with very young children: Who is teaching whom? *Int. J. Child Comput. Interact.* 25:100182. doi: 10.1016/j.ijcci.2020.100182

Vartiainen, H., Nissinen, S., Pöllänen, S., and Vanninen, P. (2018). Teachers' insights into connected learning networks: emerging activities and forms of participation. *AERA Open* 4, 1–17. doi: 10.1177/2332858418799694

Vigentini, L., Liu, D. Y., Arthars, N., and Dollinger, M. (2020). Evaluating the scaling of a LA tool through the lens of the SHEILA framework: a comparison of two cases from tinkerers to institutional adoption. *Internet High. Educ.* 45:100728. doi: 10.1016/j.iheduc.2020.100728

Van der Vleuten, C., Sluijsmans, D., and Joosten-ten Brinke, D. (2017). "Competence assessment as learner support in education," in *Competence-Based Vocational and Professional Education*, ed. M. Mulder (Cham: Springer), 607–630. doi: 10.1007/978-3-319-41713-4_28

von Wangenheim, C. G., Marques, L. S., and Hauck, J. C. (2020). *Machine Learning for All-Introducing Machine Learning in K-12*. Available online at: https://doi.org/10.31235/osf.io/wj5ne

Vygotsky, L. S. (1978). *Mind in society*. Cambridge: Harvard University Press.

Weintrop, D., Beheshti, E., Horn, M. S., Orton, K., Trouille, L., Jona, K., et al. (2014). "Interactive assessment tools for computational thinking in High School STEM classrooms," in *Proceedings of the International Conference on Intelligent Technologies for Interactive Entertainment*, Cham, 22–25. doi: 10.1007/978-3-319-08189-2_3

Wheeler, S., Yeomans, P., and Wheeler, D. (2008). The good, the bad and the wiki: Evaluating students-generated content for collaborative learning. *Br. J. Educ. Technol.* 39, 987–995. doi: 10.1111/j.1467-8535.2007.00799.x

Wing, J. M. (2006). Computational thinking. *Commun. ACM* 49, 33–35. doi: 10.1145/1118178.1118215

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.* 366, 3717–3725. doi: 10.1098/rsta.2008.0118

Wong, G. K. W., and Cheung, H. Y. (2020). Exploring children's perceptions of developing twenty-first century skills through computational thinking and programming. *Interact. Learn. Environ.* 28, 438–450. doi: 10.1080/10494820.2018.1534245

Wu, M. L., and Richards, K. (2011). "Facilitating computational thinking through game design," in *Proceedings of the International Conference on Technologies for E-Learning and Digital Entertainment*, Berlin. doi: 10.1007/978-3-642-23456-9_39

Yadav, A., Hong, H., and Stephenson, C. (2016). Computational thinking for all: pedagogical approaches to embedding 21st century problem-solving in K-12 classrooms. *TechTrends* 60, 565–568. doi: 10.1007/s11528-016-0087-7

Yan, J., Liu, J., and Tseng, F. M. (2020). An evaluation system based on the self-organizing system framework of smart cities: A case study of smart transportation systems in China. *Technol. Forecast. Soc. Change* 153:119371. doi: 10.1016/j.techfore.2018.07.009

Zhang, H., McNeil, S., Gronseth, S., Dogan, B., Handoko, E., and Ugwu, L. (2020). "Building Computational Thinking Through Teachable Machine," in *Proceedings of the Society for Information Technology & Teacher Education International Conference*, Chesapeake, VA.

Zhong, J., and Liao, H. (2015). "The study on features of computational thinking and its common operation mode methods," in *Proceedings of the 2015 International Conference on Intelligent Systems Research and Mechatronics Engineering*, Paris. doi: 10.2991/isrme-15.2015.384