

Original software publication

# VisualCodeMOOC: A course platform for algorithms and data structures integrating a conversational agent for enhanced learning through dynamic visualizations

Mingyuan Li<sup>a,1</sup>, Duan Wang<sup>a,1</sup>, Erick Purwanto<sup>a,1,\*</sup>, Thomas Selig<sup>a</sup>, Qing Zhang<sup>a</sup>, Hai-Ning Liang<sup>b</sup>

<sup>a</sup> Xi'an Jiaotong-Liverpool University, Suzhou, China

<sup>b</sup> Hong Kong University of Science and Technology (Guangzhou), Guangzhou, China

## ARTICLE INFO

### Keywords:

Programming education  
Algorithm visualization  
MOOC  
Conversational agent

## ABSTRACT

The abstract nature of algorithms and data structures poses challenges for students, and the integration of visualization into comprehensive learning systems remains underexplored. This article presents VisualCodeMOOC, incorporating VisualCodeChat, a conversational agent that enhances algorithm and data structure learning through dynamic visualizations and personalized feedback. The platform effectively addresses these challenges, improving student engagement and comprehension. With instructions structuring, novel response-based algorithm visualization, exercise design, VisualCodeMOOC provides a cohesive and supportive learning environment that promotes active learning. Evaluation results demonstrate its usability, responsiveness, and educational value, confirming its potential as a promising tool for advancing computer science education.

## Code metadata

Current code version	v1.0.0
Permanent link to code/repository used for this code version	<a href="https://github.com/ElsevierSoftwareX/SOFTX-D-24-00589">https://github.com/ElsevierSoftwareX/SOFTX-D-24-00589</a>
Code versioning system used	Git
Software code languages, tools, and services used	Typescript, OpenAI API
Compilation requirements, operating environments & dependencies	React, Node.js, Next.js, npm
If available Link to developer documentation/manual	<a href="https://github.com/XJTLU-AIED/VisualCodeMOOC/blob/main/README.md">https://github.com/XJTLU-AIED/VisualCodeMOOC/blob/main/README.md</a>

## 1. Motivation and significance

Learning algorithms and data structures pose significant challenges for students, particularly due to their abstract nature [1,2]. These topics are foundational in computer science, yet students often struggle to grasp them, leading to frustration and disengagement [3]. Algorithm visualization has been identified as a powerful tool for aiding comprehension by transforming abstract concepts into tangible visual representations [4]. However, while visualization has been employed in educational settings [5], its integration into a comprehensive learning system that addresses the diverse needs of students

remains under-explored [6]. This gap highlights the need for innovative approaches that combine visualization with other educational technologies to improve learning outcomes.

Massive Open Online Courses (MOOCs) have democratized access to education, making high-quality learning resources available to a global audience [7]. Despite their widespread adoption, MOOCs often suffer from low engagement and high dropout rates [8,9]. Intelligent Tutoring Systems (ITS) have shown promise in enhancing MOOCs by providing hints [10] and dynamic assessment [11], yet their application remains limited. The integration of ITS within MOOCs, particularly for complex topics requiring deep understanding, is still an evolving area of research

\* Corresponding author.

E-mail addresses: [Mingyuan.Li21@student.xjtlu.edu.cn](mailto:Mingyuan.Li21@student.xjtlu.edu.cn) (Mingyuan Li), [Duan.Wang21@student.xjtlu.edu.cn](mailto:Duan.Wang21@student.xjtlu.edu.cn) (Duan Wang), [Erick.Purwanto@xjtlu.edu.cn](mailto:Erick.Purwanto@xjtlu.edu.cn) (Erick Purwanto), [Thomas.Selig@xjtlu.edu.cn](mailto:Thomas.Selig@xjtlu.edu.cn) (Thomas Selig), [Qing.Zhang02@xjtlu.edu.cn](mailto:Qing.Zhang02@xjtlu.edu.cn) (Qing Zhang), [hainingliang@hkust-gz.edu.cn](mailto:hainingliang@hkust-gz.edu.cn) (Hai-Ning Liang).

<sup>1</sup> These authors contributed equally to this work.

that requires more robust solutions [12].

Intent-based chatbots can identify and respond to user intents using predefined intents [13]. In contrast, conversational agents like ChatGPT outperform intent-based chatbots, making them powerful tools for personalized learning and engagement in education [14]. Thus, conversational agents have emerged as a promising tool in education [15]. Although these agents have been employed to answer queries related to algorithms and data structures [16,17], their application in helping students fully understand these topics remains limited. Existing systems do not fully exploit the potential of combining conversational agents with visualization, an approach that could greatly enhance student comprehension.

This article presents VisualCodeChat, a conversational agent tool that integrates visualization with interactive exercises to support learning algorithms and data structures. The article demonstrates how to incorporate this conversational agent as an ITS within a MOOC platform, offering a more personalized learning experience. Our resulting platform, called VisualCodeMOOC, leverages the chatbot to guide students through course topics using visual representations and engaging exercises.

This article is structured as follows. Section 2 describes the structure and functionalities of VisualCodeMOOC, including detail implementation method of VisualCodeChat. An example of their use is presented in Section 3. The design of the experiments and evaluation results is shown in Section 4. Section 5 discusses the impact of VisualCodeMOOC and VisualCodeChat. Section 6 summarizes our findings and proposes opportunities for future work.

## 2. Software description

### 2.1. Software architecture

#### 2.1.1. The MOOC

Some prevalent MOOC platforms include Coursera, edX, and Udemy, attracting learners from all over the world [18]. MkDocs [19] serves as a lightweight framework for online courses, facilitating a streamlined process for MOOC creation. In this study, we developed a new MOOC platform named VisualCodeMOOC (Fig. 1) using MkDocs, and enhanced with the Material theme [20]. Course contents are inserted through revising files. Numerous projects have successfully leveraged MkDocs to create MOOCs. Notable examples including a highly acclaimed self-study guide for computer science [21], as well as open e-learning courses and platforms across various fields [22,23].

#### 2.1.2. Integrating chatbot into MOOC

Previous studies have made attempts to integrate conversational agents into MOOCs [24,25].

A previous study [26] explored changing the mode of information delivery from text to voice to engage students more effectively.

In this paper, a chatbot capable of dynamically visualizing algorithms (VisualCodeChat) is integrated into VisualCodeMOOC, which supports a new form of adaptive teaching activity. VisualCodeChat is integrated into the MOOC pages as Inline Frame elements. In addition to VisualCodeChat, each topic in VisualCodeMOOC includes other learning activities, such as videos and after-class exercises.

The software architecture is illustrated in Fig. 2. The topics and chatbot interactions can be revised by the instructor (1). Lessons in VisualCodeMOOC begin with an introductory video (2). Students learn the details of the algorithm by engaging with VisualCodeChat (3). After-class exercises are included for each topic (4), and a cloud database is connected to synchronize data (5).

#### 2.1.3. Implementation of VisualCodeChat

VisualCodeChat is based on “NextChat”, an open-source project which is developed using Next.js and Node.js [27]. It connects to an OpenAI API of GPT-3.5 [28] to process all the messages received (6).

**Chatbot instructions structuring.** The key to creating an effective chatbot exercise is to provide clear instructions for the conversation flow before the interaction begins. These instructions are well-structured into steps in JSON format, which is intended to guide and constrain the API responses. The instructions include requesting the API to return data that will be used as input for the visualization module mentioned later. The teaching process of VisualCodeChat is divided into four sections: an English explanation of the algorithm, a detailed annotation with visualization, a code learning portion, and practice testing. During the detailed annotation phase, the chatbot presents the algorithm step by step, each followed by a Q&A session in order to ensure students’ understanding. By structuring instructions in this manner, we limit the scope of the conversation and further mitigate potential hallucinations, thereby creating a more effective and reliable educational setting.

**From text to visualization animation.** Building on the open-source project, a visualization module based on d3.js [29] is integrated (7) to transform abstract algorithmic concepts into visual representations.

The process of generating a complete chatbot response is illustrated in Fig. 3. The structured instruction, along with the entire chat history, is provided as input to the API (a). The API’s response contains both the next tutoring message and the input for the visualization component.

For each API response, visualization data are extracted using regular expression (b), a well-established and common practice for extracting data [30,31]. The extracted data serves as the input parameter of the visualization components to generate animations aligning with the ongoing conversation (c). For certain exercises, the data required for visualization animation is generated in the JSON format, and wrapped in special characters for easier extraction and future removals. The JSON data is separated from conversational content before displayed in the chatbot interface, which ensures a smoother user experience. Finally, the tutoring message, combined with the animation generated by the visualization components, constitutes a complete chatbot response presented to the user (d). Meanwhile, the new message is incorporated into the chat history to generate the next chatbot response (e). Alternative visualization libraries can be used, provided the extracted data is correctly formatted to ensure compatibility with the visualization components.

**Exercise design.** The questions of the practice test are in either gap-fill or multiple-choice format. They are not pre-defined but are dynamically generated by the chatbot to ensure the questions are personalized based on the previous conversation. The four question categories are: conceptual understanding, algorithmic comprehension, coding proficiency, and analysis of time complexity. While the pre-designed teaching plan includes a limited number of questions, students can request additional relevant questions from the chatbot.

**Software customization.** Instructors can tailor the content by modifying existing visualizations, adding new ones, creating additional courses, and so on. For further details on customizing the contents, please refer to the GitHub README.

### 2.2. Software functionalities

VisualCodeMOOC offers 24 pre-defined commonly taught algorithms and data structures topics. These include 12 algorithms for novice-level programmers, 8 for intermediate-level, and 4 for advanced-level. Each algorithm has been rigorously pretested in pilot experiments to ensure compatibility with the OpenAI API.

Each pre-defined exercise proceeds according to the following routine (the numbers in the brackets refer to Fig. 4). First, the user

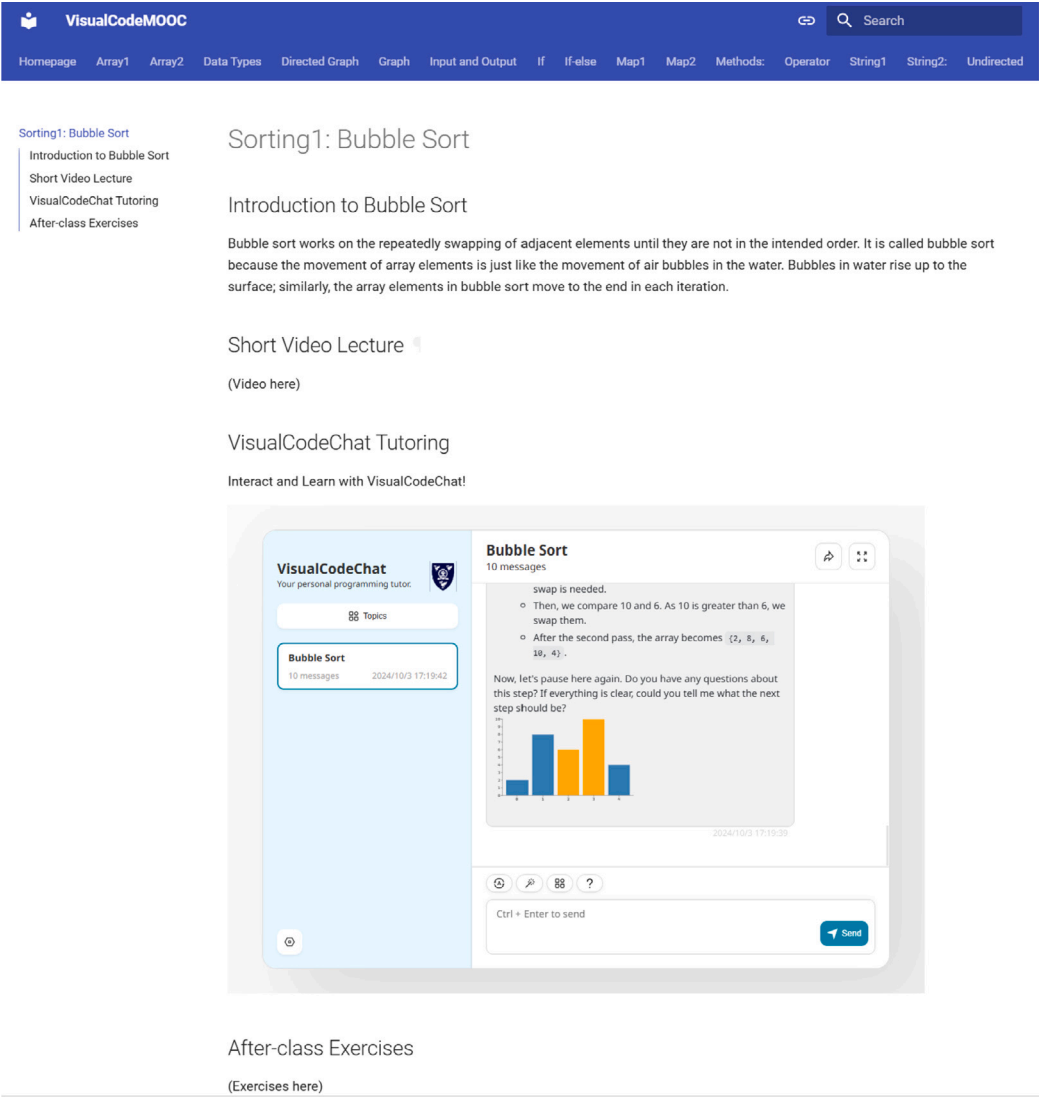


Fig. 1. VisualCodeMOOC Page.

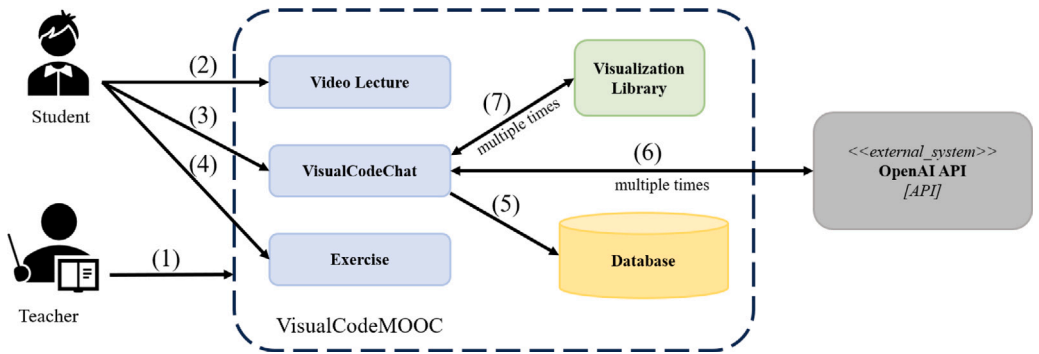


Fig. 2. Software Architecture.

chooses a topic and watches an introductory video (1–4), then ready to interact with VisualCodeChat. The chatbot greets the user (5–8) and begins teaching the algorithm according to the preset four teaching sections mentioned in Section 2.1.3. The teaching procedure starts by explaining the algorithm in words (9–12), followed by step-by-step illustrations with corresponding visual animations (13–18) and a code

interpretation section (19–22). After introducing new concepts or steps, the user is always prompted to confirm understanding. Finally, practice questions are provided, with answers given after students attempt them (23–26). The chat data is synchronized to the cloud database after the conversation for backup (27, 28). There are extra exercises after the chatbot tutoring (29, 30).

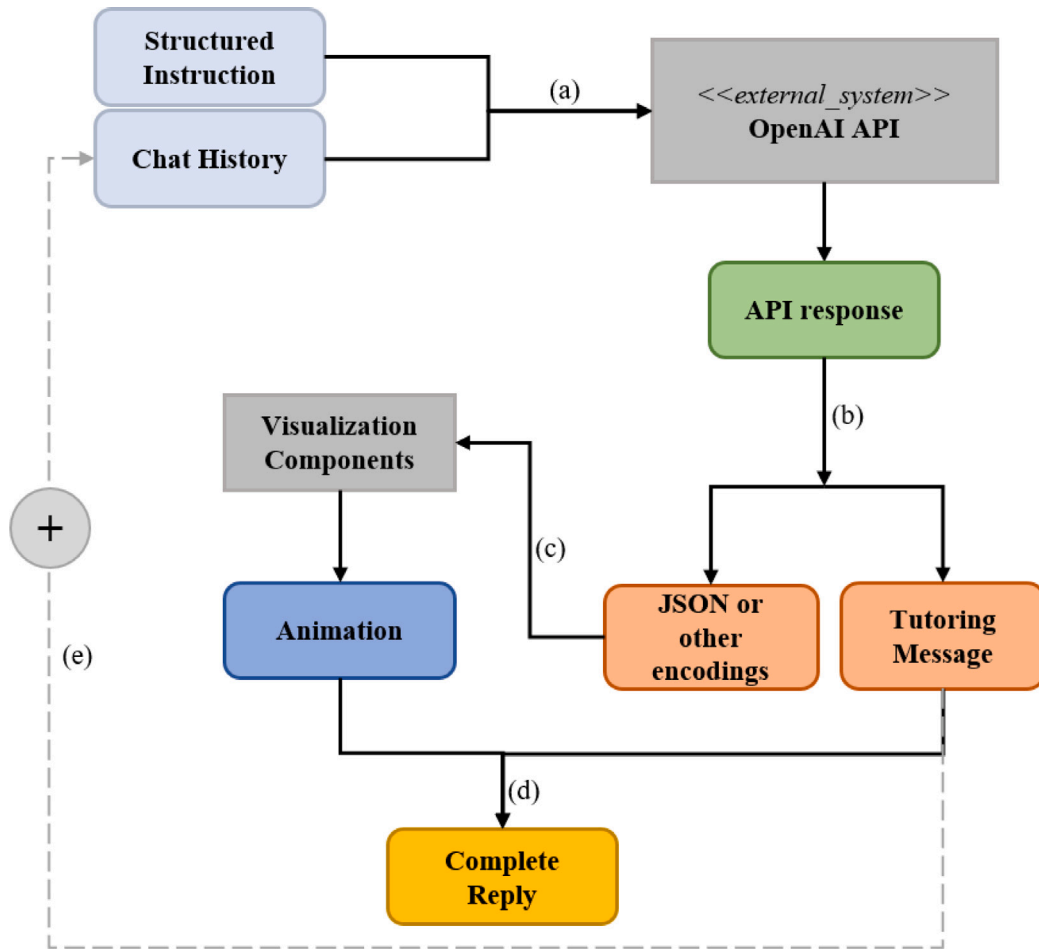


Fig. 3. From Text to Animation.

### 3. Illustrative examples

The following example illustrates an interaction of a student learning the bubble sort algorithm using VisualCodeChat.

The student begins by selecting the topic “bubble sort” and watches an introductory video on the algorithm. Subsequently, the student enters the corresponding chatbot session to learn the detailed implementation method. The following four stages show how the chatbot works.

#### 3.1. VisualCodeChat welcome

The teaching process begins with a welcome message, such as, “Hi there, it looks like you have chosen to learn bubble sort today. Are you ready to dive in?” The student can simply reply “yes” to proceed the conversation.

#### 3.2. Adaptive teaching

VisualCodeChat teaches the student about bubble sort using barchat animation according to the preset pipeline. Besides the four teaching sections mentioned in Section 2.1.3, the student is free to ask any relevant questions along the chatting process.

The student’s interaction with VisualCodeChat is facilitated through simple responses and questions. The chatbot initiates and leads the process, and the student may simply respond to the questions posed by the chatbot or pose related questions during the conversation.

The questions from VisualCodeChat are straightforward, such as “Are there any questions?”. The reply is usually as simple as “No

question, I understand” to proceed to the next step, or type in the problems encountered to obtain further explanations. Because some algorithms may include repeating steps, the chatbot sometimes asks the student to answer questions related to the topic, as demonstrated in the response shown in Fig. 5. In this example, the reply first describes the current step of the algorithm (“compare 10 and 6, and swap...”), then encourages the student to reflect on the next step (“Could you tell me what the next step should be?”). The bar chart at the bottom of the chat bubble is an animation that visually illustrates the bar swap for the current step. The animation repeatedly shows the exchange between the bar representing 10 and the bar representing 6, reinforcing the concept for the students. The reply message dynamically adapts to accommodate student’s learning pace and individual inquiries. In Fig. 6, the content in the first chat bubble is similar to the one in Fig. 5, which incorporates analysis of the current step, animation and interactive prompt. In the second chat bubble, the student asks a question about the order of comparisons, then in the third chat bubble, the chatbot answers the question by discussing the example the students provided.

#### 3.3. Interactive learning

Mistakes from students are addressed in different ways depending on the stage. VisualCodeChat provides an immediate correction if the student makes a mistake during the step-by-step illustration phase. If the mistake occurs during the multiple-choice question (MCQ) stage, the student is prompted to try up to a total of three attempts. The chatbot provides comments on the student’s answer to the quizzes, and offers elaborate analysis.

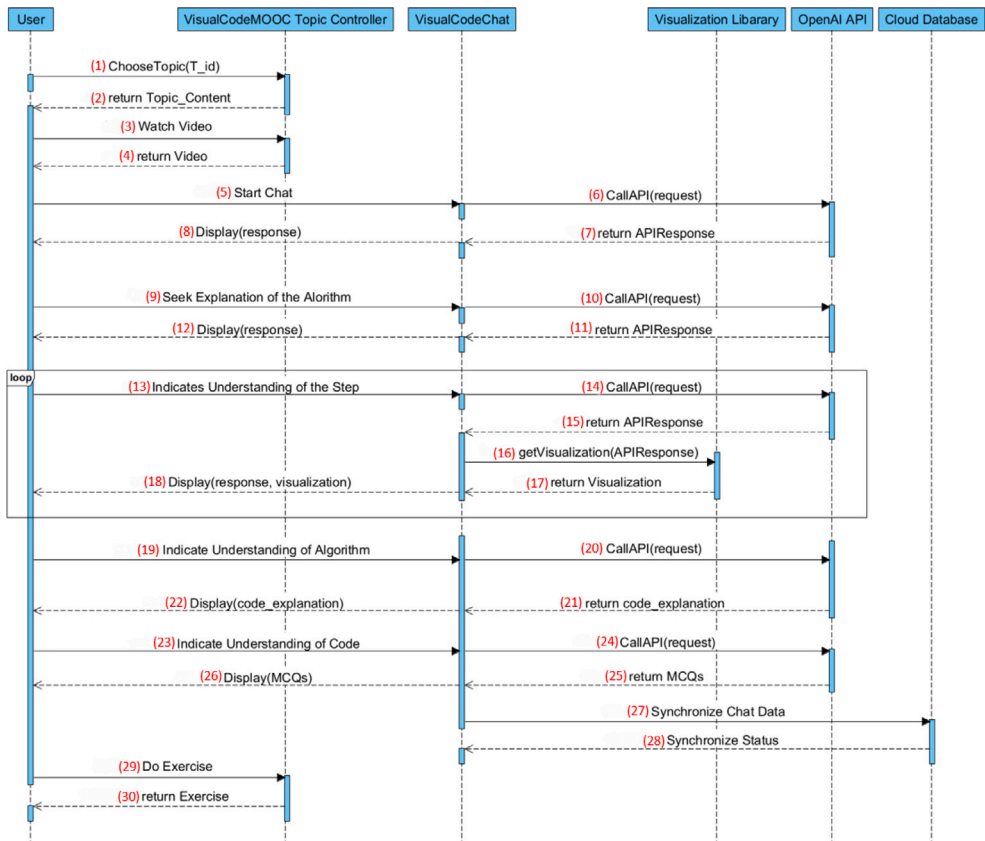


Fig. 4. Sequence Diagram of VisualCodeMOOC.

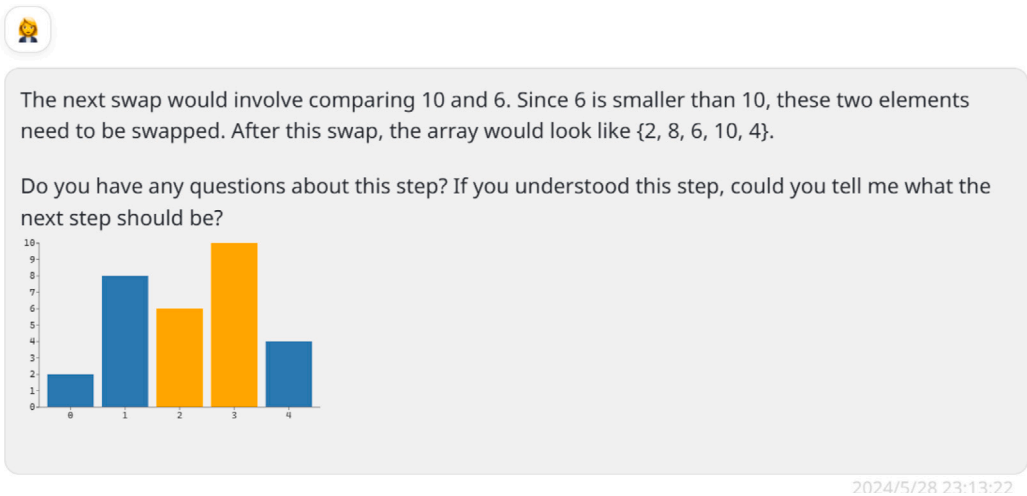


Fig. 5. VisualCodeChat Encourages Students' Independent Thinking.

The student can request hints for the questions. For instance, the student asks, “Can you give me a hint to solve this question?”. The chatbot responds with a hint such as, “Consider how the elements should be arranged in ascending order”.

3.4. Completion of the learning

Upon completion of the learning procedure, the message, “Congratulations! You have completed the learning. Feel free to ask me any related questions”. will be displayed to the user. If they wish, users then have the option to either inquire further about the algorithm or ask for more practice questions.

4. Evaluation

4.1. Pilot testing

The pilot testing was conducted with 16 international college students in a programming course at Ulink College Suzhou. The primary objective was to gather preliminary feedback and refine the experimental procedures, and the results confirmed the validity of the experimental design. For instance, when a student asked a question about vague content, such as “What is array indexing?”, our VisualCodeChat contextualized the explanation further: “Array indexing refers to the method of accessing individual elements within an array using their

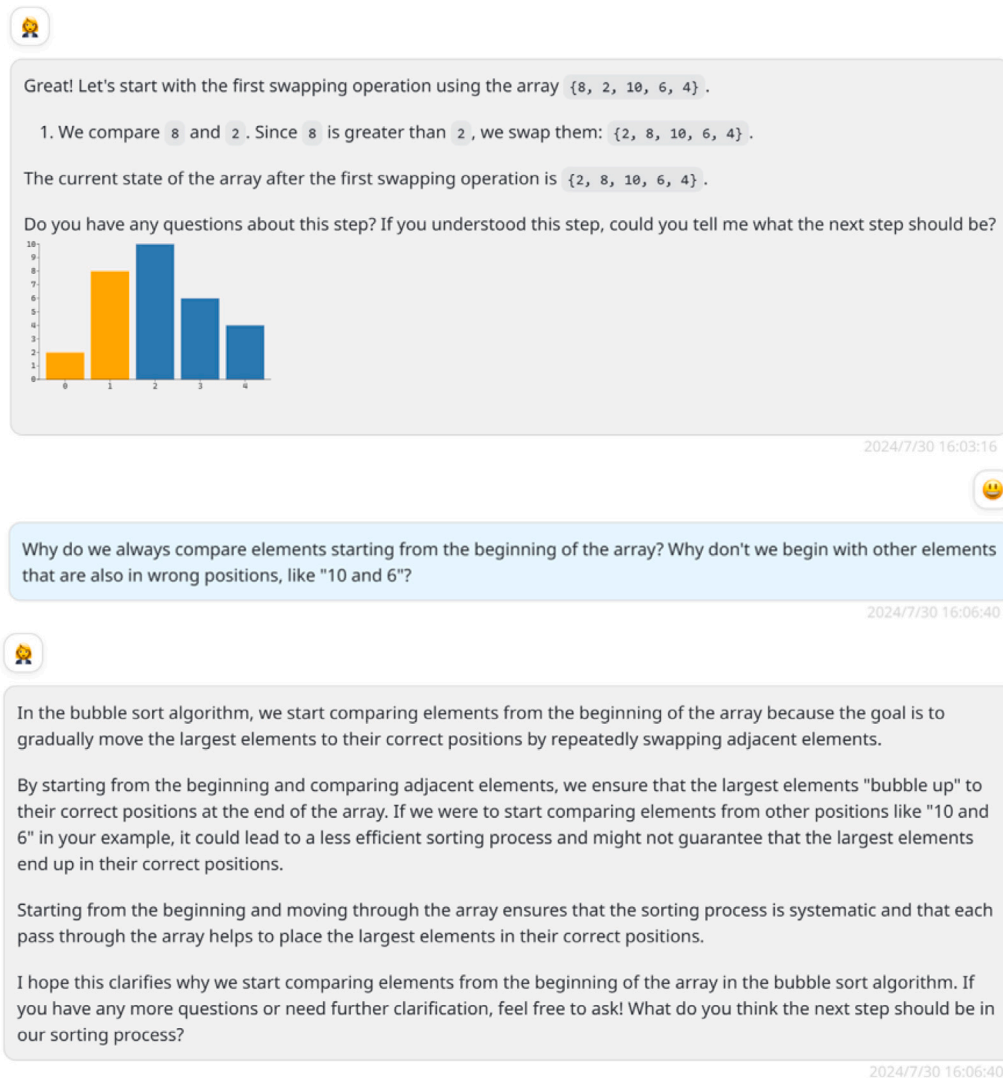


Fig. 6. Adaptive Interaction: VisualCodeChat Explain Based on Student's Question.

index", accompanied by a specific array example. Furthermore, if a student asked questions unrelated to the teaching content, leading to topic drift [32], our VisualCodeChat successfully maintained the conversation within the original instructional scope or declined to answer.

#### 4.2. Main study

The main study took place in the course "Computer Programming for Financial Mathematics" at Xi'an Jiaotong-Liverpool University. This involved 88 students from the BSc Financial Mathematics, who had no prior background in programming, making this their first systematic programming course. The topic of the experimental course was three types of sorting algorithms, which require computational thinking and understanding of algorithmic complexity. This makes them a challenging topic for programming beginners and an ideal subject for assessing the effectiveness of our approach [33]. The following usage data was collected:

- 88 students used our platform, resulting in a total of 171 individual learning engagements.
- The students engaged with VisualCodeMOOC a total of 1336 times.

A mixed research methodology was employed integrating both qualitative and quantitative methods in practice to uncover how VisualCodeMOOC facilitates and enhances the learning experience of programming students. The rationale for choosing this approach is to enhance the robustness of the research, providing a deeper understanding of the phenomena, and offering better explanations of the research findings [34].

##### 4.2.1. Quantitative evaluation

To collect quantitative data, a questionnaire was designed utilizing the Chatbot Usability Scale (BUS) [35]. The widespread application of BUS in numerous studies related to Chatbots has confirmed its rationality and efficiency [36,37]. This questionnaire features a set of BUS attributes designed to assess satisfaction with chatbots, along with items informed by Šimoňák's study [38] on the helpfulness of visualization tools for learning algorithms. The questionnaire comprises 17 items, which are grouped into the following categories: Perceived Accessibility (PA); Perceived Quality (PQ); Conversation and Information (CI); Perceived Privacy and Security (PPS); Response Time (TR); and Algorithm Visualization (AV). Each item is evaluated using a 5-point Likert Scale (LS), where 1 represents strong disagreement and 5 represents strong agreement.

**Table 1**  
Reliability analysis: Item-total statistics.

Item	Scale Mean if Item Deleted	Scale Variance if Item Deleted	Cronbach's Alpha if Item Deleted
Q01	66.679	65.396	0.888
Q02	66.333	68.425	0.889
Q03	66.704	64.211	0.881
Q04	66.617	64.439	0.881
Q05	66.642	63.508	0.880
Q06	66.605	66.517	0.887
Q07	66.556	65.950	0.885
Q08	67.259	62.019	0.885
Q09	67.074	63.319	0.884
Q10	66.716	64.531	0.882
Q11	67.086	61.355	0.879
Q12	67.247	64.688	0.892
Q13	67.185	63.178	0.883
Q14	67.222	62.9	0.888
Q15	66.704	69.186	0.895
Q16	66.568	66.398	0.887
Q17	66.802	62.41	0.88

**Table 2**  
Questionnaire Results: Number (N), Standard Deviation (SD), Average (AVE), Response Distribution.

ID	Cat.	N	SD	AVE	Response
Q01	PA	81	0.85	4.32	
Q02	PA	81	0.52	4.67	
Q03	PQ	81	0.71	4.30	
Q04	PQ	81	0.66	4.38	
Q05	PQ	81	0.76	4.36	
Q06	PQ	81	0.72	4.49	
Q07	PQ	81	0.65	4.44	
Q08	PQ	81	1.06	3.74	
Q09	PQ	81	0.91	3.93	
Q10	CI	81	0.73	4.28	
Q11	CI	81	0.89	4.14	
Q12	CI	81	0.74	4.52	
Q13	CI	81	0.90	4.38	
Q14	PPS	81	0.85	3.78	
Q15	TR	81	0.75	4.30	
Q16	AV	81	0.71	4.40	
Q17	AV	81	0.89	4.20	
Average					4.18
5-point LS Category					

SPSSPRO [39] was utilized for the reliability analysis, employing Cronbach's alpha and Standardized Cronbach's alpha to assess internal consistency [40,41]. The result of Cronbach's alpha was 0.891, indicating a high level of internal consistency close to 0.9 [42]. The Standardized Cronbach's alpha was 0.897, further corroborating the internal consistency [43]. Furthermore, The analysis in Table 1 indicates that item deletion minimally affects mean and variance, with Cronbach's alpha remaining above 0.88, thus reaffirming the results' high internal consistency [44].

Specifically, given that the participants' past experiences with similar chatbots, finding that over 96% had previously used such products, with the majority using them daily. This indicates that the participants possess a foundational ability to evaluate VisualCodeChat. The detailed analysis followed the methodology of Kayal et al. [45] and is presented in Table 2. The average score of 4.18 reflects positive feedback, with PA and most PQ questions scoring over 4.0 and a standard deviation below 1, indicating strong usability and clarity; CI and PPS questions surpassed the midpoint, suggesting favorable comprehension and accuracy, while AV scores exceeded the overall average, highlighting the effectiveness of our algorithm visualization.

#### 4.2.2. Qualitative evaluation

In addition to the quantitative analysis, qualitative data was collected from open-ended questions to gain deeper insights into our

study. Thematic analysis [46] was employed as the methodological framework due to its effectiveness in summarizing key features and generating unexpected insights [47]. Adhering to the six phases [48] to create Table 3, which presents the initial codes derived from the topics and corresponding participant responses, in accordance with Shoufan's methodology [49]. Based on concrete feedback and the alignment between initial codes and topics, our work meets the four evaluation criteria.

Additionally, chat logs were analyzed using criteria centered on Response Quality Judgments [50] to assess response quality. Notably, multiple records indicated that participants asked specific questions and received accurate answers. For example, when a participant requested, "give me the definition of Insertion Sort", VisualCodeChat successfully responded, "Insertion sort is a simple, comparison-based sorting algorithm...". Most records indicated that the visual content generated was accurate and provided clear assistance, which was positively affirmed by students with responses such as "yes, ready" and "let's go" for subsequent coding tasks.

## 5. Impact

Existing research on ITS as chatbots has mostly focused on enhancing personalized student learning experiences, such as when learning SQL (e.g., ChatbotSQL [51]), learning to code (e.g., Dodona [52]) or for

**Table 3**  
Thematic analysis result: The initial codes and examples.

Topic	Initial Codes	Example
PA	Ease of Use	The operation is simple, and the interaction is easy to understand.
	Availability	The page is clean and convenient, with quick access.
	Inclusivity	Compared to regular AI, it provides more targeted responses, allowing for a deeper understanding of my programming learning needs.
PQ	Response Speed	The response is extremely quick.
	Response Accuracy	Provides complete algorithm information, with accurate and thorough explanations.
	User Satisfaction	I find Code Examples extremely beneficial for my studies. The chatbot can generate and display code examples to demonstrate the practical application of Java concepts. These examples are invaluable resources for learners to study and learn from.
CI	Clarity and Conciseness	With detailed code explanations and in-depth interpretations of concepts, I can better understand newly defined code and learn to use it effectively.
	Information Relevance	The teaching-oriented topics are also very convenient, well-guided, and paced. Sometimes, if I stray off-topic, it reminds me to continue studying the relevant topic.
	Personalization	This chat is very intelligent, capable of understanding some of the more difficult questions I pose and providing suitable answers.
AV	Transparency	Detailed visual representations of algorithms provide an intuitive understanding of each step in the code, greatly aiding my comprehension and learning process.
	Interpretability	Visualization accurately depicts the algorithm, helping me grasp how these algorithms work more intuitively and allowing me to internalize abstract concepts in a visual and intuitive manner.
	Educational Value	When I understand the steps of the algorithm based on the text it provides, using the visualization helps me remember better.

medical training [53]. Chatbots have also been used to create MOOC assessments, enhancing self-evaluation and personalizing the learning process [54]. Building on these previous works, our VisualChatMOOC uses a chatbot for assessments and offers detailed instructional steps for learning algorithms and data structures. Further research is needed to optimize the integration of ITS within MOOCs, exploring more sophisticated adaptive learning mechanisms and leveraging real-time chat data to personalize learning paths.

VisualCodeChat, the chatbot component, provides visualizations in the interactive discussions to support the learning of algorithms and data structures. While chatbots have been used to generate images for visualizing data [55], our approach dynamically generates animations based on randomly generated examples provided by the chatbot within the conversation. This enables students to request visualized examples, addressing the gap in algorithms and data structures education by turning abstract concepts into tangible visual representations.

Future research could focus on refining the visualization capabilities of chatbots, making them more intuitive and responsive to student queries [56]. Additionally, exploring the long-term impact of chatbot-assisted learning on student performance in algorithms and data structures could offer deeper insights into the efficacy and knowledge retention of this tool [57]. This research is conducted at a local college and a university via VisualCodeMOOC platform, which will be shared globally with more institutions and students for future implementation. Further development and use of visualization-capable chatbots for learning other subjects [58] will be reflected on VisualCodeMOOC as well. Furthermore, exploring how our chatbot and visualizations can be applied in other educational settings, such as non-MOOC-based LMS platforms, is valuable. A potential approach could involve developing more flexible learning environments, like interactive “playgrounds”, to offer personalized learning experiences.

## 6. Conclusions

VisualCodeMOOC, integrating VisualCodeChat as an ITS within a MO-OC platform, has demonstrated significant potential in enhancing the learning of algorithms and data structures. By offering dynamic visualizations and personalized feedback, the platform addresses the inherent challenges of learning these abstract topics, leading to improved student engagement and comprehension. Our evaluation results highlight the system’s effectiveness, with positive feedback on usability, responsiveness, and educational value. This confirms the system’s success in achieving its objectives, making it a promising tool for improving computer science and software engineering education.

Looking forward, future research should focus on enhancing the personalization and responsiveness of the platform, potentially through advanced adaptive learning mechanisms and more responsive visualizations. Expanding the application of the platform to other complex subjects and assessing its long-term impact on student performance are also crucial steps. Additionally, integrating the VisualCodeChat with broader educational ecosystems could provide a more interactive learning experience. With these enhancements, the platform could serve as a model for a transformative approach to adaptive online teaching and learning.

## CRedit authorship contribution statement

**Mingyuan Li:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Duan Wang:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Erick Purwanto:** Writing

– review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Conceptualization. **Thomas Selig:** Writing – review & editing, Supervision, Methodology, Investigation. **Qing Zhang:** Writing – review & editing, Supervision, Resources, Project administration. **Hai-Ning Liang:** Writing – review & editing, Supervision, Project administration.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgments

This work is supported by research funds XJTLU RDF-22-01-062, TDF2324-R27-230, RC4AIED202406, AoFE Implementing Computational Thinking Education in K-12 and Higher Education Special Interest Group, and Advance HE Global Impact Grants 2023-24. We thank the STEAM Centre at Ulink College of Suzhou Industrial Park for their support.

## References

- [1] Gomes A, Mendes A. Learning to program - difficulties and solutions. In: Proceedings of the international conference on engineering education - ICEE 2007. 2007. p. 283–7. <https://api.semanticscholar.org/CorpusID:55563372>.
- [2] Mtaho A, Mselle L. Difficulties in learning the data structures and algorithms course: Literature review. *J Informatics* 2024;4(1). <http://dx.doi.org/10.59645/tji.v4i1.136>.
- [3] Lahtinen E, Ala-Mutka K, Järvinen HM. A study of the difficulties of novice programmers. *SIGCSE Bull* 2005;37(3):14–8. <http://dx.doi.org/10.1145/1151954.1067453>.
- [4] Cetin I, Andrews-Larson C. Learning sorting algorithms through visualization construction. *Comput Sci Educ* 2016;26(1):27–43. <http://dx.doi.org/10.1080/08993408.2016.1160664>.
- [5] Adorni G, Piatti S, Karpenko V. Virtual CAT: A multi-interface educational platform for algorithmic thinking assessment. *SoftwareX*. 2024;27:101737. <http://dx.doi.org/10.1016/j.softx.2024.101737>.
- [6] Hu L. Utilization of differentiated instruction in K-12 classrooms: a systematic literature review (2000–2022). *Asia Pac Educ Rev* 2024;25:507–25. <http://dx.doi.org/10.1007/s12564-024-09931-y>.
- [7] Baturay MH. An overview of the world of MOOCs. *Procedia - Soc Behav Sci* 2015;174:427–33. <http://dx.doi.org/10.1016/j.sbspro.2015.01.685>.
- [8] Ferguson R, Clow D. Examining engagement: Analysing learner subpopulations in massive open online courses (MOOCs). In: Proceedings of the fifth international conference on learning analytics and knowledge. New York, NY: Association for Computing Machinery; 2015. p. 51–8. <http://dx.doi.org/10.1145/2723576.2723606>.
- [9] Zhang Q, Bonafini FC, Lockee BB, Jablow KW, Hu X. Exploring demographics and students' motivation as predictors of completion of a massive open online course. *Int Rev Res Open Distrib Learn* 2019;20(2). <http://dx.doi.org/10.19173/irrodl.v20i2.3730>.
- [10] Baneres D, Saiz J. Intelligent tutoring system for learning digital systems on MOOC environments. In: 2016 10th international conference on complex, intelligent, and software intensive systems. CISIS, 2016. p. 95–102. <http://dx.doi.org/10.1109/CISIS.2016.47>.
- [11] Yilmaz R, Yurdugül H, Karaoglan Yilmaz FG, Şahin M, Sulak S, Aydin F, Tepeç M, Müftüoğlu CT, Oral Ö. Smart MOOC integrated with intelligent tutoring: A system architecture and framework model proposal. *Comput Education: Artif Intell* 2022;3:100092. <http://dx.doi.org/10.1016/j.caeai.2022.100092>.
- [12] Alevan V, et al. The beginning of a beautiful friendship? Intelligent tutoring systems and MOOCs. In: Conati C, Heffernan N, Mitrovic A, Verdejo M, editors. Artificial intelligence in education. AIED 2015. Lecture notes in computer science, vol. 9112, Cham: Springer; 2015. [http://dx.doi.org/10.1007/978-3-319-19773-9\\_53](http://dx.doi.org/10.1007/978-3-319-19773-9_53).
- [13] Alsafari B, Atwell E, Walker A, Callaghan M. Towards effective teaching assistants: From intent-based chatbots to LLM-powered teaching assistants. *Nat Lang Process J* 2024;8:100101. <http://dx.doi.org/10.1016/j.nlp.2024.100101>.
- [14] Sánchez Cuadrado J, Pérez-Soler S, Guerra E, De Lara J. Automating the development of task-oriented LLM-based chatbots. In: ACM conversational user interfaces 2024. ACM. 2024. p. 1–10. <http://dx.doi.org/10.1145/3640794.3665538>.
- [15] Okonkwo CW, Ade-Ibijola A. Chatbots applications in education: A systematic review. *Comput Education: Artif Intell* 2021;2:100033. <http://dx.doi.org/10.1016/j.caeai.2021.100033>.
- [16] Okonkwo CW, Ade-Ibijola A. Python-bot: A chatbot for teaching python programming. *Eng Lett* 2020;29(1):25–34.
- [17] Shah A, Jain B, Agrawal B, Jain S, Shim S. Problem solving chatbot for data structures. In: 2018 IEEE 8th annual computing and communication workshop and conference. Las Vegas, NV, USA; 2018. p. 184–9. <http://dx.doi.org/10.1109/CCWC.2018.8301734>.
- [18] Perifanou M, Economides AA. The landscape of MOOC platforms worldwide. *Int Rev Res Open Dist Learn* 2022;23(3):104–33. <http://dx.doi.org/10.19173/irrodl.v23i3.6294>.
- [19] Christie T. Mkdocs team. MkDocs. Proj Doc mark-down. 2014. Accessed26July2024.
- [20] Donath M. Material for mkdocs. 2023. Accessed26July2024.
- [21] PKUFlyingPig. CS-SELF-LEARNING. <https://csdiy.wiki/>, URL Accessed26July2024.
- [22] Wang J, Zhang C, Ding Y, Yang Y. Awesome-META+: Meta-learning research and learning platform. 2023. <http://dx.doi.org/10.48550/arXiv.2304.12921>, arXiv preprint.
- [23] Potůčková M, Albrechtová J, Anders K, Červená L, Dvořák J, Gryguc K, Höfle B, Hunt L, Lhotáková Z, Marcinkowska-Ochtyra A, et al. E-TRAINEE: Open e-learning course on time series analysis in remote sensing. *Int Arch Photogramm Remote Sens Spat Inf Sci* 2023;48:989–96. <http://dx.doi.org/10.5194/isprs-archives-XLVIII-1-W2-2023-989-2023>.
- [24] Amarnani S, Bhagat N, Ekade H, Gupta A, Sahu S. A complete chatbot based architecture for answering user's course-related queries in MOOC platforms. In: 2021 international conference on computing, communication and green engineering. CCGE, IEEE; p. 2021. <http://dx.doi.org/10.1109/CCGE50943.2021.9776437>.
- [25] Mzwri K, Turcsányi-Szabó M. Chatbot development using APIs and integration into the MOOC. *Cent Eur J New Technol Res Educ Pr* 2023;5(1):18–30. <http://dx.doi.org/10.36427/CEJNTREP.5.1.5041>.
- [26] Pereira J, Fernández-Raga M, Osuna-Acedo S, Roura-Redondo M, Almazán-López O, Buldón-Orlala A. Promoting learners' voice productions using chatbots as a tool for improving the learning process in a MOOC. *technol knowl learn*. vol. 24, (4):Springer; 2019. p. 545–65. <http://dx.doi.org/10.1007/s10758-019-09414-9>.
- [27] NextChat. ChatGPT-Next-Web. 2023, GitHub. Accessed26July2024.
- [28] Brown T, Mann B, Ryder N, Subbiah M, Kaplan JD, Dhariwal P, et al. Language models are few-shot learners. *Adv Neural Inf Process Syst* 2020;33:1877–901. <http://dx.doi.org/10.48550/arXiv.2005.14165>.
- [29] D3. d3. 2010, GitHub. Accessed26July2024.
- [30] Chen Q, Banerjee A, Demiralp Ç, Durrett G, Dillig I. Data extraction via semantic regular expression synthesis. *Proc ACM Program Lang* 2023;7:287:1848–77. <http://dx.doi.org/10.1145/3622863>, (OOPSLA2).
- [31] Li Y, Krishnamurthy R, Raghavan S, Vaithyanathan S, Jagadish HV. Regular expression learning for information extraction. In: Lapata M, Ng HT, editors. Proceedings of the 2008 conference on empirical methods in natural language processing. association for computational linguistics. 2008. p. 21–30. <https://aclanthology.org/D08-1003>.
- [32] Verkijk S, Roothaert R, Pernis R, Schlobach S. Do you catch my drift? On the usage of embedding methods to measure concept shift in knowledge graphs. In: K-CAP 2023 - proceedings of the 12th knowledge capture conference, 2023. Pensacola, FL, United States; 2023. p. 70–4. <http://dx.doi.org/10.1145/3587259.3627555>.
- [33] Kordaki M, Miatidis M, Kapsampelis G. A computer environment for beginners' learning of sorting algorithms: design and pilot evaluation. *Comput Educ* 2008;51(2):708–23. <http://dx.doi.org/10.1016/j.compedu.2007.07.006>.
- [34] Al-Hafdi FS, AlNajdi SM. The effectiveness of using chatbot-based environment on learning process, students' performances and perceptions: A mixed exploratory study. *Educ Inf Technol* 2024. <http://dx.doi.org/10.1007/s10639-024-12671-6>.
- [35] Borsci S, Malizia A, Schmettow M, et al. The chatbot usability scale: the design and pilot of a usability scale for interaction with AI-based conversational agents. *Pers Ubiquit Comput* 2022;26:95–119. <http://dx.doi.org/10.1007/s00779-021-01582-9>.
- [36] Borsci S, Schmettow M, Malizia A, et al. A confirmatory factorial analysis of the chatbot usability scale: A multilanguage validation. *Pers Ubiquit Comput* 2023;27:317–30. <http://dx.doi.org/10.1007/s00779-022-01690-0>.
- [37] Amaro I, Barra P, Della Greca A, Francese R, Tucci C. Believe in artificial intelligence? A user study on the ChatGPT's fake information impact. *IEEE Trans Comput Soc Syst* 2023. <http://dx.doi.org/10.1109/TCSS.2023.3291539>.
- [38] Šimonač S. Using algorithm visualizations in computer science education. *Open Comput Sci* 2014;4(3):183–90. <http://dx.doi.org/10.2478/s13537-014-0215-4>.
- [39] Scientific platform serving for statistics professional. SPSSPRO (version 1.0.11) [software]. 2021. <https://www.spsspro.com>.
- [40] Flaherty S-J, McCarthy M, Collins A, McAuliffe F. Can existing mobile apps support healthier food purchasing behaviour? Content analysis of nutrition content. *Behav Chang Theory User Qual Integr Public Heal Nutr* 2018;21(2):288–98. <http://dx.doi.org/10.1017/S1368980017002889>.

- [41] George D, Mallery P. IBM SPSS statistics 25 step by step: a simple guide and reference. 15th ed. New York: Routledge; 2018. <http://dx.doi.org/10.4324/9781351033909>.
- [42] Streiner DL. Starting at the beginning: An introduction to coefficient alpha and internal consistency. *J Pers Assess* 2003;80(1):99–103. [http://dx.doi.org/10.1207/S15327752JPA8001\\_18](http://dx.doi.org/10.1207/S15327752JPA8001_18).
- [43] Sun W, Chou CP, Stacy AW, et al. SAS and SPSS macros to calculate standardized Cronbach's alpha using the upper bound of the phi coefficient for dichotomous items. *Behav Res* 2007;39:71–81. <http://dx.doi.org/10.3758/BF03192845>.
- [44] Kopalle PK, Lehmann DR. Alpha inflation? The impact of eliminating scale items on Cronbach's alpha. *Organ Behav Hum Decis Process* 1997;70(3):189–97. <http://dx.doi.org/10.1006/obhd.1997.2702>.
- [45] Kayalı B, Yavuz M, Balat Ş, Çalışan M. Investigation of student experiences with ChatGPT-supported online learning applications in higher education. *Australas J Educ Technol* 2023;39(5):20–39. <http://dx.doi.org/10.14742/ajet.8915>.
- [46] Braun V, Clarke V. Thematic analysis. In: H Cooper, PM Camic, DL Long, AT Panter, D Rindskopf, KJ Sher, editors. *APA handbook of research methods in psychology, 2. research designs: quantitative, qualitative, neuropsychological, and biological*. American Psychological Association; 2012, p. 57–71. <http://dx.doi.org/10.1037/13620-004>.
- [47] Yeung MWL, Yau AHY. A thematic analysis of higher education students' perceptions of online learning in Hong Kong under COVID-19: challenges, strategies and support. *Educ Inf Technol* 2022;27:181–208. <http://dx.doi.org/10.1007/s10639-021-10656-3>.
- [48] Braun V, Clarke V. Using thematic analysis in psychology. *Qual Res Psychol* 2006;3(2):77–101. <http://dx.doi.org/10.1191/1478088706qp063oa>.
- [49] Shoufan A. Exploring students' perceptions of ChatGPT: thematic analysis and follow-up survey. *IEEE Access* 2023;11:38805–18. <http://dx.doi.org/10.1109/ACCESS.2023.3268224>.
- [50] Jiang J, Ahuja N. Response quality in human-chatbot collaborative systems.. In: *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval (SIGIR '20)*. vol. 154, New York, NY, USA: Association for Computing Machinery; 2020, p. 5–1548. <http://dx.doi.org/10.1145/3397271.3401234>.
- [51] Pérez-Mercado R, Balderas A, Muñoz A, Cabrera JF, Palomo-Duarte M, Doderio JM. ChatbotSQL: Conversational agent to support relational database query language learning. *SoftwareX*. 2023;22:101346. <http://dx.doi.org/10.1016/j.softx.2023.101346>.
- [52] Van Petegem C, Dawyndt P, Mesuere B, Dodona: Learn to code with a virtual co-teacher that supports active learning. *SoftwareX*. 2023;24:101578. <http://dx.doi.org/10.1016/j.softx.2023.101578>.
- [53] Freedman R, Edwards L, Sullivan I, Brockmann A, LaBarbera D, Naples V. Teaching cardiovascular physiology with a minigame-based ITS. In: Frasson C, Mylonas P, Troussas C, editors. *Intelligent tutoring systems (ITS) 2023. Lecture notes in computer science*. 13891, Cham: Springer; 2023, p. 137–46. [http://dx.doi.org/10.1007/978-3-031-32883-1\\_12](http://dx.doi.org/10.1007/978-3-031-32883-1_12).
- [54] Rai L, Deng C, Liu F. Developing massive open online course style assessments using generative AI tools. In: *Proceedings of the 2023 IEEE 6th international conference on electronic information and communication technology*. Qingdao, China; 2023, p. 1292–4. <http://dx.doi.org/10.1109/ICEICT57916.2023.10244824>.
- [55] Ye Y, Hao J, Hou Y, Wang Z, Xiao S, Luo Y, Zeng W. Generative AI for visualization: State of the art and future directions. *Vis Informatics*. 2024;8(2):43–66. <http://dx.doi.org/10.1016/j.visinf.2024.04.003>.
- [56] Kim NW, Ko HK, Myers G, Bach B. ChatGPT in data visualization education: A student perspective. 2024, <http://dx.doi.org/10.48550/arXiv.2405.00748>, arXiv preprint.
- [57] Zhang R, Zou D, Cheng G. A review of chatbot-assisted learning: pedagogical approaches, implementations, factors leading to effectiveness, theories, and future directions. *Interact Learn Environ* 2023;1–29. <http://dx.doi.org/10.1080/10494820.2023.2202704>.
- [58] Kavaz E, Puig A, Rodríguez I. Chatbot-based natural language interfaces for data visualisation: A scoping review. *Appl Sci* 2023;13(12):7025. <http://dx.doi.org/10.3390/app13127025>.