

VL/HCC 2025

Designing Conversational AI to Support Think-Aloud Practice in Technical Interview Preparation for CS Students

Taufiq Daryanto, Sophia Stil, Xiaohan Ding, Daniel Manesh, Sang Won Lee, Tim Lee, Stephanie Lunn, Sarah Rodriguez, Chris Brown, Eugenia Rho



VIRGINIA
TECH.

CODE
PATH
*ORG

FIU

Introduction

Technical Interviews

A specific job interview that requires candidates to write code that solves programming challenges while simultaneously think-aloud/ think out loud
(Aziz et al., 2012)

Introduction

Think Aloud in Technical Interviews



Think-aloud involves verbalizing interviewee's problem-solving approach to the interviewer while working through a coding task (McDowell, 2013).

Introduction

Think Aloud in Technical Interviews



Why it matters: Interviewers evaluate not only technical ability but also how candidates approach problems and communicate their thought process (Ford et al., 2017).

Introduction

Think Aloud in Technical Interviews



Why it matters: Interviewers evaluate not only technical ability but also how candidates approach problems and communicate their thought process (Ford et al., 2017).



Challenge: The think-aloud process often becomes a major source of stress for candidates (Behroozi et al., 2020).

Introduction

Common Preparation Tools Do Not Support Think-Aloud Practice



Common coding platforms such as LeetCode and HackerRank focus solely on problem-solving, without facilitating think-aloud practice.

Introduction

Leveraging Conversational AI for Interview Practice

- Prior work has shown that LLM-based conversational AI can facilitate interview practice

Introduction

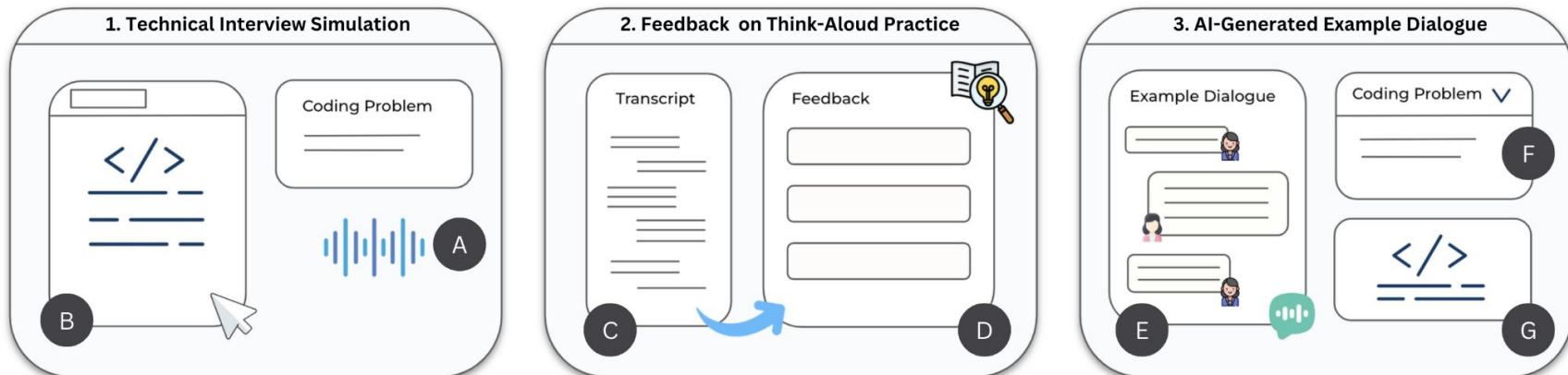
Leveraging Conversational AI for Interview Practice

- Prior work has shown that LLM-based conversational AI can facilitate interview practice
- However existing research on conversational AI has largely focused on general interview practice and communication training, leaving the specific domain technical interview practice underexplored

**RQ: How do users perceive the role of conversational AI
for think-aloud practice in tech interview preparation?**

Facilitating Think-Aloud Practice in Tech Interviews

We developed a technology probe in the form of an LLM-based technical interview practice tool to better elicit users' insights on how AI can support think-aloud practice.



System Design

1. Technical Interview Simulation

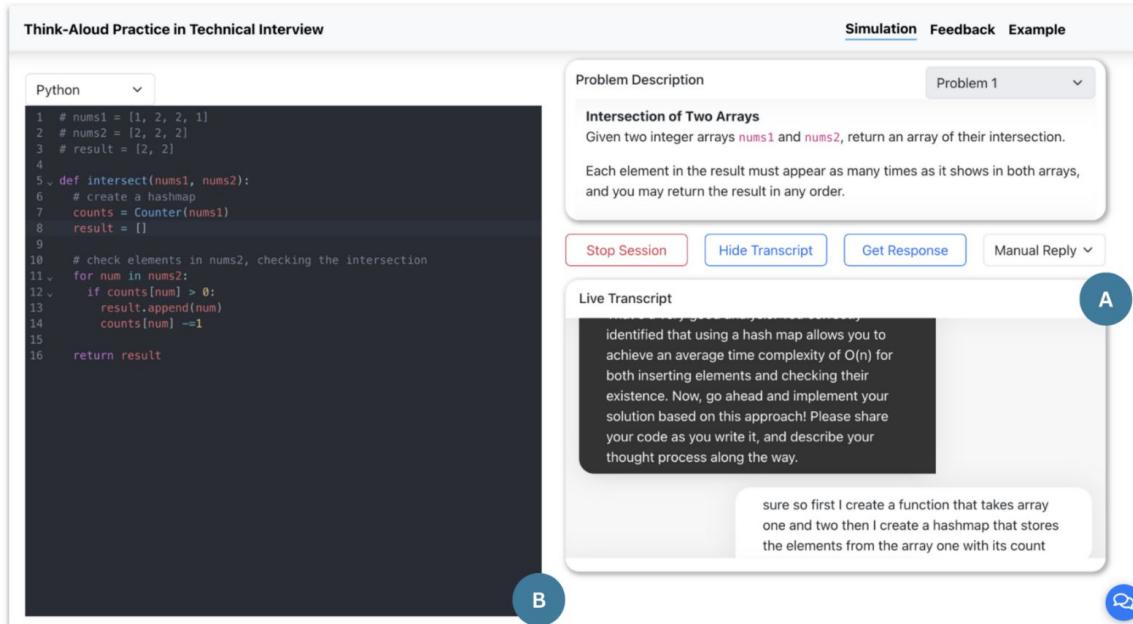


Fig. 1: **Technical Interview Simulation**. AI-facilitated mock interview tool that simulates a technical interview experience through (A) voice-based natural conversation. The interaction flow mimics an actual technical interview, beginning with the AI interviewer asking questions, followed by user responses. During this interaction, users can type code in an (B) integrated editor while verbally *think-aloud* explaining their thoughts and solutions.

System Design

2. Feedback on Think-Aloud Practice

The screenshot displays a user interface for 'Think-Aloud Practice in Technical Interview'. At the top, there are tabs for 'Simulation', 'Feedback' (which is currently selected), and 'Example'. Below the tabs, two sections are shown: 'Transcript' (labeled A) and 'Assessment of the Think-Aloud Process' (labeled B).

Transcript (Section A):

- interviewer:** Hi! Thank you for joining the interview today. I hope you're doing well. Let's dive into the coding problem I'll be presenting to you: Intersection of Two Arrays. Given two integer arrays nums1 and nums2 return an array of their intersection. Each element in the result must appear as many times as it shows in both arrays, and you may return the result in any order. Do you understand the problem? If you have any questions or need clarification, feel free to ask!
- interviewee:** so I have a question what say there's like no common elements in both arrays how should it be sorted should be returned an empty array or
- interviewer:** Great question! If there are no common elements in both arrays, you would return an empty array. You can assume that this case is valid and should be handled in your solution. Do you have any initial ideas on how to approach solving the problem?
- interviewee:** and I have another question like how should duplicates how should duplicates be handled

Assessment of the Think-Aloud Process (Section B):

- Feedback on Understanding:** The interviewee demonstrated a good understanding of the problem by asking relevant clarifying questions about edge cases, such as handling duplicates and the expected output when there are no common elements. They effectively grasped the requirements and did not miss opportunities to seek clarity.
- Feedback on Initial Ideation:** The interviewee brainstormed multiple approaches, discussing the use of a hashmap (Counter) and comparing it to a brute force method. They articulated their thought process clearly, explaining why the hashmap would be more efficient for counting occurrences and finding intersections. However, the explanation could have been more structured.
- Feedback on Idea Justification:** The interviewee justified their choice of using a hashmap by explaining its efficiency in terms of lookup time and handling duplicates. They provided a rationale for their approach, but could have compared it more explicitly to alternative methods to strengthen their justification.
- Feedback on Implementation:** While coding, the interviewee communicated their thought process reasonably well, detailing the steps they were taking. However, there were moments where their explanations became a bit convoluted, making it harder to follow their reasoning. A clearer structure in their communication would enhance understanding.
- Feedback on Review (Dry Run):** The interviewee performed a thorough dry run of their code using a specific test case. They clearly explained the flow of execution, how counts were modified, and how the result was built step-by-step. They identified the expected output correctly and demonstrated a solid understanding of the code's behavior.
- Feedback on Evaluation:** The interviewee did not explicitly discuss potential optimizations or additional edge cases after coding. While they performed a dry run effectively, a more in-

Fig. 2: **AI Feedback on Think-Aloud Practice**. Providing (B) feedback on users' think-aloud performance based on the interview simulation (A) transcript.

System Design

3. AI-Generated Think-Aloud Example Dialogue

The screenshot shows a user interface for a technical interview practice tool. At the top, it says "Think-Aloud Practice in Technical Interview". Below that, there are tabs for "Simulation", "Feedback", and "Example", with "Example" being the active tab. A blue circle labeled "A" is positioned above a text box containing a user's thought process about solving a problem using a hash map. A blue circle labeled "B" is positioned above a "Problem Description" section for "Intersection of Two Arrays". A blue circle labeled "C" is positioned above a code editor showing Python code for finding the intersection of two arrays using a hash map.

A

I think one way to solve this is to use a hash map to count the occurrences of each number in the first array, and then iterate through the second array to check for common elements. This way, we can efficiently track the counts.

That sounds like a solid approach. Can you explain how you would implement this?

Sure! First, I'll create a hash map to store the counts of each element in `nums1`. Then, I'll create a result array to store the intersection. Finally, I'll iterate through `nums2` and check if each element exists in the hash map. If it does, I'll add it to the result array and decrease its count in the hash map. Let me start coding this.

Now, I'll start coding. First, I'll create the hash map.

Next, I'll create the result array and iterate through `nums2`.

B

Problem Description

Given two integer arrays `nums1` and `nums2`, return an array of their intersection.

Each element in the result must appear as many times as it shows in both arrays, and you may return the result in any order.

C

```
1 hash_map = {}
2 for num in nums1:
3     if num in hash_map:
4         hash_map[num] += 1
5     else:
6         hash_map[num] = 1
7 result = []
8 for num in nums2:
9     if num in hash_map and hash_map[num] > 0:
10        result.append(n)
11
```

Explanation: Here, I'm checking if the number from `nums2` exists in the hash map and if its count is greater than zero. If both conditions are met, I add it to the result and decrement the count. This ensures that we only add the number as many times as it appears in both arrays.

Fig. 3: **AI-Generated Think-Aloud Example Dialogue**. Providing (A) AI-generated dialogues that model thinking aloud during a technical interview for each (B) coding problem. The goal is to help users learn from examples relevant to their current task and better articulate their thought processes. The feature also includes a step-by-step (C) code solution alongside the dialogue, with each utterance displayed sequentially and accompanied by voice output.

User Study

- **Method:** Participants used the tool, followed by semi-structured interviews to gather insights on using AI for think-aloud practice.
- **Participants:** 17 CS students (10 men, 7 women; 13 undergrads, 4 grads) with varied interview experience

Findings

Finding 1: Technical Interview Simulations

AI Facilitate Think-Aloud Practice Through Interactive Conversation

Concerns About Overly Positive AI Interviewers

- A few users noted that the AI interviewer was overly positive
- Suggested adding customizable AI personas to reflect diverse interviewer styles

Finding 2: AI Feedback on Think-Aloud Practice

Providing Feedback Beyond Verbal Content Analysis

- e.g. suggestion to provide feedback on balancing time between thinking, talking, and coding (P17)

Framing Feedback to Improve Trust

- suggestion on framing the feedback from a third-person interviewer perspective rather than directly from AI (P14)

Finding 3: AI-Generated Example of Technical Interview Dialogues

Vicarious Learning Through AI-Generated Examples Allow Self-Evaluation

- *“Conversation examples are more helpful than just someone telling the logic” - P6*

Unrealistic AI-Generated Examples Can Discourage Learners

- *“The answer is too good to be true because I don’t think in any real interview, even if you are very good at coding interviews [...] you are going to have this exact, very organized line of thought.” - P3*
- Suggestion on human-AI collaborative crowdsourcing approach:
“crowdsource [examples] from users simulations with AI” (P14)

Finding 4: Supporting Inclusion and Addressing Intersectional Challenges

Promoting Equal Access Through AI-Assisted Technical Interview Practice

- *"I don't have a lot of female friends who are in computer science ... so having a tool like this ... would allow for more people to have access to [practice]."* - P11

Intersectional Challenges in AI-Assisted Think-Aloud Practice

- *"I'm not a native speaker, ... English is not my first language. So sometimes I'm like, 'Yeah, sorry, blah blah.' [So] in the feedback, it says, 'You are very unsure.'... But I think [there should be a] room for error."*
- P3



Discussion

Discussion

Rethinking the Role of AI in Technical Interview Practice

Prior works in interview practice has primarily focused on developing AI interviewers as stand-alone practice partners. Including our work. While such systems can provide practice opportunities, however:

**Should the primary goal of these systems be to entirely
replace human practice partners?**

Discussion

Rethinking the Role of AI in Technical Interview Practice

- Some users mentioned that practicing with friends is more enjoyable and experienced a stronger human connection
- *However practicing with peer is “more laid back because there is no one pressuring you [and that] they usually don’t give great feedback.” - P17*

Discussion

Rethinking the Role of AI in Technical Interview Practice

- Some users mentioned that practicing with friends is more enjoyable and experienced a stronger human connection
- *However practicing with peer is “more laid back because there is no one pressuring you [and that] they usually don’t give great feedback.” - P17*

Future Direction: Leveraging Human-AI Collaboration

“Instead of just relying on the AI ... you can basically just have the interview going on between the two people. And then, the AI interviewer goes over their feedback ... because [the AI feedback] already has a lot of detailed information that extends further into the actual code as well as the talk aloud portion. So I think that helps bring both worlds together because you’re getting both the human interaction as well as the AI portion” - P17

Thank you!

VL/HCC 2025

Designing Conversational AI to Support Think-Aloud Practice in Technical Interview Preparation for CS Students

Taufiq Daryanto, Sophia Stil, Xiaohan Ding, Daniel Manesh, Sang Won Lee, Tim Lee, Stephanie Lunn, Sarah Rodriguez, Chris Brown, Eugenia Rho



VIRGINIA
TECH.

CODE
PATH
*ORG

FIU

Discussion

Design Implications

1. Technical Interview Simulation
 - a. Designing for social presence: turn-taking and emotional reciprocity
2. AI Feedback on Think-Aloud Practice
 - a. Feedback on balancing time between thinking, talking, and coding
 - b. Framing of feedback: AI as a mediator, not authority
3. AI-Generated Think-Aloud Example Dialogue
 - a. Crowdsourcing think-aloud examples through human-AI collaboration