

TOEFL[®] Research Report

TOEFL-RR-92

ETS Research Report No. RR-21-06

The Development and Evaluation of Interactional Competence Elicitor for Oral Language Assessments

Evgeny Chukharev-Hudilainen

Gary J. Ockey

December 2021

The *TOEFL*[®] test is the world's most widely respected English language assessment, used for admissions purposes in more than 130 countries including Australia, Canada, New Zealand, the United Kingdom, and the United States. Since its initial launch in 1964, the TOEFL test has undergone several major revisions motivated by advances in theories of language ability and changes in English teaching practices. The most recent revision, the *TOEFL iBT*[®] test, contains a number of innovative design features, including integrated tasks that engage multiple skills to simulate language use in academic settings and test materials that reflect the reading, listening, speaking, and writing demands of real-world academic environments. In addition to the TOEFL iBT, the TOEFL Family of Assessments has expanded to provide high-quality English proficiency assessments for a variety of academic uses and contexts. The TOEFL Young Students Series (YSS) features the *TOEFL Primary*[™] and *TOEFL Junior*[®] tests, designed to help teachers and learners of English in school settings. The *TOEFL ITP*[®] Assessment Series offers colleges, universities, and others an affordable test for placement and progress monitoring within English programs.

Since the 1970s, the TOEFL tests have had a rigorous, productive, and far-ranging research program. ETS has made the establishment of a strong research base a consistent feature of the development and evolution of the TOEFL tests, because only through a rigorous program of research can a testing company demonstrate its forward-looking vision and substantiate claims about what test takers know or can do based on their test scores. In addition to the 20-30 TOEFL-related research projects conducted by ETS Research & Development staff each year, the TOEFL Committee of Examiners (COE), composed of distinguished language-learning and testing experts from the academic community, funds an annual program of research supporting the TOEFL family of assessments, including projects carried out by external researchers from all over the world.

To date, hundreds of studies on the TOEFL tests have been published in refereed academic journals and books. In addition, more than 300 peer-reviewed reports about TOEFL research have been published by ETS. These publications have appeared in several different series historically: TOEFL Monographs, TOEFL Technical Reports, TOEFL iBT Research Reports, and TOEFL Junior Research Reports. It is the purpose of the current TOEFL Research Report Series to serve as the primary venue for all ETS publications on research conducted in relation to all members of the TOEFL Family of Assessments.

Current (2021 – 2022) members of the TOEFL COE are:

| | |
|----------------------|----------------------------------|
| Lorena Llosa – Chair | New York University |
| Beverly Baker | University of Ottawa |
| Tineke Brunfaut | Lancaster University |
| Atta Gebril | The American University of Cairo |
| April Ginther | Purdue University |
| Claudia Harsch | University of Bremen |
| Talia Isaacs | University College London |
| Yasuyo Sawaki | Waseda University |
| Dina Tsagari | Oslo Metropolitan University |
| Koen Van Gorp | Michigan University |
| Wenxia Zhang | Tsinghua University |

To obtain more information about the TOEFL programs and services, use one of the following:

E-mail: toefl@ets.org Web site: www.ets.org/toefl



ETS is an Equal Opportunity/Affirmative Action Employer.

As part of its educational and social mission and in fulfilling the organization's non-profit Charter and Bylaws, ETS has and continues to learn from and also to lead research that furthers educational and measurement research to advance quality and equity in education and assessment for all users of the organization's products and services.

RESEARCH REPORT

The Development and Evaluation of Interactional Competence Elicitor for Oral Language Assessments

Evgeny Chukharev-Hudilainen, & Gary J. Ockey

Iowa State University, Ames, IA

This paper describes the development and evaluation of Interaction Competence Elicitor (ICE), a spoken dialog system (SDS) for the delivery of a paired oral discussion task in the context of language assessment. The purpose of ICE is to sustain a topic-specific conversation with a test taker in order to elicit discourse that can be later judged to assess the test taker's oral language ability, including interactional competence. The development of ICE is reported in detail to provide guidance for future developers of similar systems. The performance of ICE is evaluated on two aspects: (a) by analyzing system errors that occur at different stages in the natural language processing (NLP) pipeline in terms of both their preventability and their impact on the downstream stages of the pipeline, and (b) by analyzing questionnaire and semistructured interview data to establish the test takers' experience with the system. Findings suggest that ICE was robust in 90% of the dialog turns it produced, and test takers noted both positive and negative aspects of communicating with the system as opposed to a human interlocutor. We conclude that this prototype system lays important groundwork for the development and use of specialized SDSs in the assessment of oral communication, which includes interactional competence.

Keywords Interactional competence; second language assessment; paired oral discussion task; spoken dialog system; natural language processing

doi:10.1002/ets2.12319

This paper reports on the development and evaluation of Interaction Competence Elicitor (ICE), a prototype spoken dialog system (SDS) for the delivery of a paired oral discussion task for oral language assessments that target interactional competence. This report aims to present the interdisciplinary community of researchers in language assessment and natural language and speech processing with a case study of how a research prototype of an SDS can be rapidly developed with limited resources for investigating an issue of theoretical importance. The challenge that is addressed by this work stems from the difficulties of developing and deploying large-scale, high-stakes oral language tests that assess interactional competence as part of the construct of oral language ability. Interactional competence, defined as an individual's ability to contribute to the shared understanding of information by orally responding appropriately to a given situation (Ockey, 2017), is arguably a crucial component of oral communication ability and thus should not be ignored by test developers. Generally, tasks that assess interactional competence are believed to require synchronous communication between the test taker and another speaker. The inclusion of another speaker, on the one hand, provides the context and the input necessary to judge a test taker's interactional competence; on the other hand, it also greatly reduces the practicality of the assessment and increases the potential for introducing score variance that results from the characteristics of the other speaker and thereby is construct-irrelevant (Ockey, 2009).

One possible solution for this problem is to utilize an SDS as the "other speaker" in the context of oral language assessment. The SDS then would sustain a task-specific conversation with the test taker and thereby elicit oral discourse that can be judged to assess the test taker's oral language ability, including, crucially, their interactional competence. This approach is taken in the present paper by developing and evaluating ICE. It is important to note that the scope of the present paper does not include evaluating the effect of ICE (vs. a human interlocutor) on the scoring of the test takers' performance. Such empirical evaluation is reported separately (Ockey & Chukharev-Hudilainen, 2021). Our current focus is squarely on the development and evaluation of the performance of ICE in terms of eliciting potentially scorable discourse from the test taker. The empirical evaluation reported in the present paper, therefore, follows two vectors: (a) a technical evaluation of the performance of the system, with a detailed analysis of system failures that occurred within the NLP pipeline in an

Corresponding author: E. Chukharev-Hudilainen, E-mail: evgeny@iastate.edu

experiment, and (b) an analysis of test takers' perceptions of their interactions with the system as opposed to a human interlocutor.

Consistent with the outlined goals, the remainder of the present paper is organized as follows: First, a brief review of the relevant literature is provided, followed by a detailed description of all design decisions that were made in the process of developing ICE with an additional focus on showing how these decisions are grounded in the specifications of the assessment task which the system is designed to deliver. Then, data collection and analysis methodology is detailed, followed by presentation of empirical results and discussion of findings.

Background

The potential of using SDSs for assessment and learning purposes has been widely discussed. Possible benefits of SDSs include the ability to provide fine-grained diagnosis, generate adaptive instructional materials, perform repeated formative evaluation, and scaffold student learning via automated intelligent tutoring (e.g., Graesser et al., 2001; Graesser & McDaniel, 2017). The context of an oral language assessment task is different from content-based assessment and learning in that the specific goal of utilizing an SDS in this context is to evaluate, very narrowly, the test taker's oral language skills (rather than their mastery of content knowledge).

This goal can be approached from two different directions: (a) by repurposing an existing, off-the-shelf SDS, and (b) by developing a specialized SDS for the particular purpose at hand. Previous research has considered both directions. In terms of the efficacy of repurposing an off-the-shelf SDS, findings have been mixed. For instance, Litman et al. (2016) investigated the extent to which three off-the-shelf SDSs could elicit scorable discourse from test takers. Three interactive tasks (searching for a laptop, finding a restaurant, and giving directions), each handled by a separate SDS, were used in the study. Sixty-seven nonnative speakers of English completed the tasks in English and were assigned holistic scores on the Council of Europe Framework (CEFR; Council of Europe, 2001) proficiency-level descriptors by human raters. A major finding of the study was that the SDS did not elicit speech that was representative of natural discourse. In some cases, the test takers had little opportunity to contribute to the conversation, whereas in others, the test takers' discourse was found to be quite unnatural, which may be attributed to the fact that they were trying to accommodate to the non-natural-sounding system-generated discourse.

An alternative to using off-the-shelf SDSs is developing specialized systems geared toward eliciting scorable discourse. Over the past 10 years, multiple systems have been developed for the purposes of second-language learning and assessment (for recent reviews, see Bibauw et al., 2019; Litman et al., 2018). Broadly, these systems can be classified into four categories (Bibauw et al., 2019): narrative, form-focused, goal-oriented, and reactive. In a narrative (or branching) system, the user (i.e., the language learner or the test taker) has to choose from a closed list of utterances expressing different messages, all appropriate and grammatically correct. A form-focused system limits the user, in a context of a meaningful dialog, to selecting and arranging sequences or words or to reading a given utterance out loud without the ability to alter the meaning of the utterance. In a goal-oriented system, the user is encouraged to interact dialogically with the system to attain a specific, clearly defined goal, such as accomplishing a real-world task or answering a set of questions. Finally, a reactive system does not constrain the task or the topic of conversation, leaving the management of the interaction entirely up to the user; the system tries to respond in an appropriate way to every user message. From the four categories, the latter two (i.e., goal-oriented and reactive) can, in principle, deliver a naturalistic dialog experience wherein the test taker interacts with the system as they would with a human interlocutor.

An example of such a specially developed SDS (in particular, goal-oriented) is one developed by Timpe-Laughlin et al. (2017) for the learning and assessment of pragmatic competence. Researchers investigated the potential of short (a few turns) dialogs between the SDS and a test taker. Test takers were given a task that required them to call their boss, set up a meeting time, and review some presentation slides. Of particular importance to the present paper, the researchers investigated the form of the request strategies followed by the users of the system. Requests were sometimes combined and sometimes separate, making it more challenging for the system to respond appropriately. It was also found that, even for this rather simple task, a range in the number of turns was needed to accomplish the task. Natural language understanding (NLU) in this system was implemented through keyword spotting: The user's turn was processed to detect certain task-specific keywords, such as "meeting," "meet," or "slides," which then determined the system's response.

From the standpoint of technical implementation, an SDS can be largely conceptualized as consisting of two components: a task-independent “delivery framework” and a task-specific “dialog manager” (DM). The delivery framework handles all infrastructural logistics, such as acquiring audio from the user, delivering it to the system, processing and storing the audio, and synthesizing the system’s response. One well-established delivery framework, developed specifically with language assessment applications in mind, is HALEF (Ramanarayanan et al., 2015; Ramanarayanan et al., 2019). HALEF integrates a set of open-source components: a telephone server, Asterisk; a voice browser, JVoiceXML; a voice server, Cairo, which allows JVoiceXML to integrate with Asterisk; automated speech recognizers (ASRs), Kaldi and Sphinx-4; text-to-speech (TTS) synthesis engines, Festival and Mary; and an Apache Tomcat-based web server. All these components need to be installed and configured on a server. The HALEF framework, for example, was used to develop and deploy the system described in Timpe-Laughlin et al. (2017), discussed previously.

The first author of the present paper recently developed a minimalistic delivery framework as part of his graduate course on computer-assisted language learning taught at Iowa State University. This framework was subsequently released by the author into the public domain.¹ It is a small application written in full-stack ECMAScript (JavaScript), using Node.js and Express on the server side, that interfaces with cloud-based TTS and ASR (provided by Google Cloud and Amazon Web Services). Because this framework was originally developed with instructional purposes in mind, there have been no publications or presentations about it for research audiences. Because this delivery framework was utilized for the present study, a detailed account of design decisions made while developing this framework is presented in the Methods section below with the intent of providing developers with insight that can be used for replicating our work in their projects.

Once a delivery framework is in place, the development of an SDS amounts to creating a task-specific DM, which in the simplest case could take the form of a text-based chat bot. The DM handles NLU, that is, converting the output of the ASR system into data structures representing the user’s utterance in the task-specific domain; deciding what to say next (i.e., dialog management proper); and natural language generation (NLG), that is, converting “conceptual” data structures into natural language output suitable for the TTS system. For example, in the HALEF framework, the DM can be implemented as a VoiceXML document. For the purposes of the present project, the DM was developed as a simple Perl script (see the Methods section).

In light of this discussion, the present paper aims to report on the design of ICE, a specialized SDS for the assessment of the construct of oral communication ability, which includes interactional competence. The empirical study reported in the paper is guided by two research questions:

Research Question 1. To what extent are the different components of ICE successful in attaining the stated purpose of the system (i.e., sustaining a conversation with the test taker that elicits scorable discourse)?

Research Question 2. What are the test takers’ perceptions of ICE as a conversational partner?

Methods

System Development

The specific language assessment task that was selected as the target task for ICE was part of the institutional English Placement Test, oral communication section (EPT OC), which is administered to incoming international students at Iowa State University. In the current practice, the test taker completes a paired-discussion task with another test taker. The test taker’s goal in this task is to defend a particular point of view in a discussion of a given prompt related to academic life. ICE was developed as a substitute for the other test taker. With the system-delivered task, the test taker is asked to communicate with ICE and argue against a prompt-specific matter of discussion. In the present study, the system was developed and evaluated on two prompts. In Prompt 1, the matter of discussion was group work in college (ICE argued in favor of group work, while the test taker argued against it); in Prompt 2, the discussion was the requirement to take gym classes while in college (ICE argued in favor of mandatory gym classes, while the test taker argued against them).

The following requirements were taken into consideration when deciding on the approach to developing ICE. Firstly, ICE should support a coherent conversation with the test taker about the matter of the discussion and faithfully play the role designated in the task design; that is, arguing in favor of the prompt-specific matter of discussion. Importantly, the conversation should keep going for the predetermined duration of the task (4 min) in order to elicit a sufficient quantity of turns from the test taker. Secondly, the discourse elicited from the test taker should give raters opportunities to judge

aspects of the test taker's interactional competence, such as initiating and expanding on their own ideas, connecting their ideas to the partner's, expanding on the partner's ideas, making relevant comments, taking turns appropriately, asking appropriate questions, disagreeing and/or agreeing politely, and answering questions in an appropriate amount of time in a pragmatically appropriate manner for a given context (Ockey & Chukharev-Hudilainen, 2021). To that end, ICE should generate utterances of sufficient quality, which we define as providing the test taker with meaningful opportunities to demonstrate the various aspects of their interactional competence.

Thus, the design and development of ICE followed a combination of the reactive (responding to the test taker's prior utterance) and the goal-oriented (generating utterances that are useful for eliciting the test taker's interactional competence) approach. Importantly, the goal-oriented component focused not only on the goal of the conversation itself (i.e., defending the prescribed viewpoint), but also on the "meta-goal" of the language assessment in which the task was incorporated; that is, allowing the test taker to demonstrate their interactional competence. This latter consideration is arguably unique to assessment contexts and thus sets our system apart from others.

With these preliminary remarks in mind, in the remaining part of this subsection, we describe the implementation of the delivery framework and task-specific DM scripts.

User Interface Implementation

The user interface (i.e., the means by which the test taker interacts with ICE), was implemented in the delivery framework as a client-server, full-stack JavaScript, single-page web application, with the server side implemented using Node.js² and Express.³ The system assigned each dialog session (i.e., a conversation session between a single user and ICE) a unique identifier on the server side. The client side of the application (JavaScript code running within the user's browser) then opened a WebSocket connection to the server, which was used throughout the dialog session to maintain real-time communication between the client and the server side. Node.js provides the benefit of the asynchronous and efficient handling of multiple client connections by a single instance of the server-side application.

Upon receiving the user's permission to access the microphone, the client side began to continually acquire the audio signal through the HTML5 AudioContext interface as a stream of raw audio samples. This stream was split into chunks of 1,024 sample-frames, downsampled as needed to conserve the internet connection bandwidth, and transmitted to the server side over the WebSocket connection as a sequence of binary objects (blobs). The server reassembled the chunks of audio data received from the client. The resultant audio stream was then concurrently piped into two destinations: (a) the ASR system for automatic speech-to-text conversion and (b) the filesystem for long-term storage (files were initially stored in the raw audio format, and then compressed to the MP3 format in the background, using the FFmpeg tool⁴).

The ASR system returned a plain-text output to the server. This output was stored in a log file on the server side and was then passed to the DM script, as described below. The DM script analyzed the test taker's utterance and generated the system's response, using an algorithm that was specific to the task and the prompt. The response was returned, as plain text, to the server.

The server stored the plain-text response in the log file and sent it to the TTS system for automatic text-to-speech conversion. The TTS system returned an audio stream, which was then piped by the server into two destinations: (a) an HTTP endpoint to be retrieved by the client side (i.e., the user's web browser) and (b) the server filesystem for long-term storage. When the TTS-generated audio stream was ready, the server notified the client by sending a message over the WebSocket connection. Upon receiving this notification, the client side accessed the HTTP endpoint, retrieved the audio stream, and played it back to the test taker using an HTML5 Audio element. While the system-generated utterance was playing, the user's microphone could be picking up the sound of this playback. To avoid erroneous interpretation of such audio signal as part of the test taker's own utterance, the transmission of audio from the client to the server was suspended during the playback of the system-generated utterance. A limitation of this approach was that if the test taker were to interrupt the system in the middle of the system's turn, this interruption would be ignored by the system. However, a scenario where the system interrupts the test taker's turn could still be implemented.

With the present approach, the test taker was expected to experience a naturalistic, low-latency, full-duplex conversation with ICE. In addition, after each dialog session, the long-term server-side storage would contain the following files:

- (a) a plain-text log file with both ASR-generated transcriptions of test taker's utterances and system-generated responses,
- (b) MP3 recordings of all of the test taker's utterances, and
- (c) TTS-generated MP3 files containing the system's utterances.

TTS and ASR Implementation

As it is evident from the above description, the server side of ICE maintained an interface to an ASR service, which received the audio stream of the user's utterance and converted it into plain text for further processing by the system. Instead of using installed ASR software, such as Dragon or Apple Dictation, ICE employed a cloud-based speech recognition service, Google Speech-To-Text.⁵ This service was selected due to its low cost (at the time of writing, \$0.004 for 15 s of audio input, or an estimated \$0.032 for a 4-min conversation), low latency, and high accuracy of speech recognition.⁶

The server-side application also maintained an interface to a TTS engine, which converted system-generated utterances from plain text into an audio stream that was then delivered to the client side for playback to the test taker. In the present approach, synthesized rather than prerecorded natural speech was used because it enabled ICE to generate utterances on the fly and to reuse some language from the test taker's utterances. This capability was important for the purpose of assessing interactional competence because it provided the ability for the system to initiate repairs of communication breakdowns. For example, ICE may prompt the test taker to give a clarification by asking them, "What do you mean by X?," where X is a word that the test taker has just spoken. After reviewing several cloud-based TTS services, Amazon Polly⁷ was utilized due to its low cost (at the time of writing, \$4.00 per 1 million characters, or an estimated \$0.0056 for a 4-min conversation), high quality, and fast performance.

Google Speech-To-Text and Amazon Polly do not allow for customization, such as implementing custom acoustic or speech models; however, their off-the-shelf versions proved to deliver high accuracy while maintaining a low latency (ICE had an end-to-end latency less than 1 s).

Dialog Management Implementation

The above-described delivery framework handles the logistics of the web-based client-server connection, interfacing with the ASR and TTS systems, maintaining many concurrent dialog sessions, and creating audio and plain-text logs of each session. With this infrastructure in place (which, in itself, is task-independent), the development of ICE (i.e., a particular task-specific SDS for the present project) amounted to creating a DM script that accepted plain-text utterances from the user and returned plain-text system responses; this is the same as developing a text-based dialog system. Specifically, two scripts were developed, one for each of the two prompts supported by the system as described in the present paper.

The DM scripts receive as input plain-text test-taker utterances and output plain-text system-generated responses. The DM scripts can be implemented in any programming language as they are only loosely coupled with the server side of the delivery framework. The DM scripts are placed in a filesystem location that is accessible by the server. The filesystem path to the DM script is then provided in the configuration of the server, and the server interacts with the DM script in the following manner.

When a new dialog session with ICE is initiated, the server spawns a new instance of the DM script and supplies, as a command-line parameter, a unique identifier of the dialog session. The DM script opens a Transmission Control Protocol (TCP) socket connection to the server and transmits the dialog session identifier. This allows the server to match the DM instance to the dialog session identifier.

The server then enters a loop, described in more detail above, wherein it obtains ASR-generated transcriptions of test taker's utterances, transmits these transcriptions as plain text message over the TCP connection to the DM script, awaits a plain-text response from the DM script, and finally, converts the plain-text response to an audio stream using TTS. The DM script is expected to implement a matching loop. In this loop, the DM script first awaits a plain text input from the server over the TCP connection. Once the input is received, it is processed in a task-specific way to generate a system response to the test taker's utterance. Finally, the plain-text response is transmitted to the server over the TCP connection.

The development of the DM scripts for ICE was conducted in two steps. First, a small seed corpus of task-specific conversations was collected and qualitatively analyzed to generate insights into possible dialog management strategies for the system. Second, strategies were enumerated and implemented. The following subsections provide details on the implementation of each step.

Seed Corpus Analysis

Members of the research team recorded a series of pretend conversations, with one member playing the role of ICE (and arguing in favor of the prompt-specific matter of discussion) and the other playing the role of the test taker (and arguing against the matter). These conversations formed what we called a “seed corpus.” The size of the seed corpus was about 170 utterances per prompt. A qualitative analysis of the seed corpus was conducted to (a) enumerate the possible arguments in favor and against the prompt-specific matter and (b) identify potential keywords signaling the arguments against the matter.

For example, for Prompt 1, the following possible arguments in favor of group work were identified:

- Group work encourages creativity.
- It allows for dividing responsibilities among group members.
- It helps improve communication skills.
- It provides opportunities for interacting with new people.
- It increases student motivation.

The arguments against group work were the following:

- Group work may lead to within-group conflicts.
- There are benefits to working independently.
- Group work makes meeting deadlines more challenging.
- It may be difficult to schedule group meetings.
- Students may need to deal with unhelpful group members.
- Group work makes it easier to get distracted.
- Group work may be more time-consuming than individual work.

Along the same lines, arguments for and against required gym classes in college were identified for Prompt 2.

Dialog Management Strategies

The analysis of the seed corpus and the requirements of the language-assessment application determined a set of strategies that ICE was to follow to sustain a conversation with the test taker. Broadly, these strategies fell into two categories: (a) strategies to maintain a coherent dialog with the test taker and (b) strategies to give the test taker an opportunity to demonstrate features of interactional competence. The former category was represented by the following two strategies: (a) refuting the test taker’s arguments against the matter of discussion and (b) presenting arguments in favor of the matter of discussion. The latter category was represented by the following strategies: (a) interrupting the test taker, (b) producing an unintelligible utterance, and (c) unexpectedly changing the topic. The three strategies in the latter category intentionally disrupted the flow of conversation with the test taker to give them an opportunity to demonstrate their ability to handle such disruptions, for example, by asking for a clarification or appropriately reacting to the abrupt topic change.

The primary goal of the DM script was to orchestrate these five strategies in a way that would be helpful for eliciting scorable discourse from the test taker. With this goal in mind, we prioritized the second category of strategies over the first one. The DM script was programmed to engage in strategies from the second category early in the course of the conversation, as shown in Table 1.

Strategy 1: Refuting the Test Taker’s Arguments Against the Matter of Discussion

In order to logically and coherently refute the test taker’s argument, it was necessary for the system to first understand the argument that had been made. The NLU functionality of the DM script was implemented as a two-step procedure.

The first step was to differentiate between two types of test-taker utterances based on their length. As demonstrated by the analysis of the seed corpus, short utterances (operationalized as shorter than 10 words) were either incomplete utterances followed by an excessive hesitation pause (e.g., “I could be but ...” [Dialog 5175, turn 16U]) or brief responses not making or developing an argument (e.g., “I don’t think so” [Dialog 8415, turn 15U]). Long utterances (operationalized as 10 words or longer), on the other hand, would substantially contribute to the conversation by making or developing

Table 1 ICE Dialog Management Strategies

| Turn | Strategy employed |
|-----------|---------------------------------------|
| 1 | None (hard-coded opening utterance) |
| 2 | 1 and 2 (maintaining coherent dialog) |
| 3 | 3 (interruption) |
| 4 | 1 and 2 (maintaining coherent dialog) |
| 5–6 | 4 (unintelligible utterance) |
| 7 | 3 (interruption) |
| 8 | 5 (unexpected topic change) |
| 9 et seq. | 1 and 2 (maintaining coherent dialog) |

an argument or by countering the argument that had previously been made by ICE. The 10-word threshold was derived through an expert judgment based on the authors' observation of the pilot data. Strategy 1 was only employed to generate a response to a long utterance produced by the test taker. If the test taker's utterance was short, then Strategy 2 was employed instead (see the Strategy 2 subsection).

The second step of the NLU procedure was to determine which particular argument (or arguments) was the focus of the test taker's long utterance. This step was accomplished by matching the test taker's utterance against a manually engineered set of regular expressions. Each of the regular expressions in the set identified the presence of a certain prompt-specific keyword in the utterance that corresponded to arguments that the test taker might be making against the matter of discussion.

Regular expressions (Table 2) were developed based on the seed corpus and then improved through a semistructured iterative process. At each iteration of this process, the current set of regular expressions was run on the seed corpus, with each of the utterances being matched against the regular expressions in the set, and the list of keywords identified in the utterance was output. Then, this output was manually analyzed to identify a list of misses (i.e., keywords that were present in the utterance from the corpus but passed undetected by the regular expressions) and false alarms (i.e., keywords corresponding to the arguments that were not actually present in the utterance but were mistakenly detected by a regular expression). Based on this analysis, regular expressions were revised first to avoid false alarms (i.e., to increase the precision of detection) and then, whenever possible, to avoid misses (i.e., to increase the recall of detection). Precision was prioritized over recall because a false alarm was expected to have a higher chance of derailing the conversation than a miss. The revised set of regular expressions was then the starting point for the next iteration.

Iterations continued until subjectively acceptable levels of precision and recall were reached. We found that, for each prompt, only three iterations were required to develop the regular expressions. Two research assistants, an undergraduate student in linguistics and a graduate student in applied linguistics, did most of the work with developing regular expressions under the supervision of the first author.

If no keywords were detected in a test taker's utterance, then the DM script would not be able to carry on with Strategy 1 and would fall back to Strategy 2 (see below). Otherwise, the next step was to generate a coherent system response to the test taker's utterance. Two banks of templates were created during the development phase to help the system accomplish this step: (a) a bank of keyword-specific responses containing at least two utterances per keyword, each providing a counter-argument to the test taker's argument and (b) a bank of six generic challenges not specific to any particular keyword. For example, for the keyword CONFLICT, one of the specific responses was written as "Conflict is not a pleasant experience, but conflict resolution strategies from group projects are good skills to have for life." An example of a generic challenge from the latter bank was "Do you think that would be the case for everyone?"

The system would first check if there was a previously unused keyword-specific response in the former bank. If one was present, then it would be used as the SDS response to the test taker for the current turn. If no specific arguments were present, however, then the system would draw from the latter bank of non-keyword-specific challenges. Each template was only allowed to be used once during a single dialog session to avoid unnatural repetitiveness in the utterances generated by the SDS. If several keywords were detected in the test taker's utterance, one of them was chosen at random for the purposes of generating a response.

For example, consider the following utterance made by a test taker: "<...> sometimes there are always in each group of people there are some **people who** are not familiar with some brainstorming and dealing with new ideas and might kill

very creative ideas but if you are working **on your own** you can you can easily pursue your objectives and your goals and your creative ideas” (Dialog 1280, turn 3U; emphasis added). The words “people who” matched the regular expression for the keyword BAD_PEOPLE, and the words “on your own” matched the regular expression for the keyword INDEPENDENCE. Thus, the system correctly understood that the test taker was making two points against group work: (a) the threat of running into unhelpful group members (keyword BAD_PEOPLE) and (b) the benefits of working independently (keyword INDEPENDENCE). In this particular dialog session, from the two keywords, INDEPENDENCE happened to be randomly chosen as the target for the system response. The following was extracted from the bank of templates and used as the final text of the system utterance: “Working independently is a useful skill, but working with groups is more important because teamwork is so important for careers later on” (Dialog 1280, turn 3C).

Strategy 2: Presenting Arguments in Favor of the Matter of Discussion

This strategy would be employed in cases when the test taker’s utterance could not be coherently responded to by following Strategy 1: either (a) the utterance was short or (b) it did not contain any detectable keywords. In such cases, the system could not reliably determine whether the test taker was actively putting forward an argument against the matter of discussion; therefore, it did not “know” what (if anything) it should be refuting.

To maintain coherence of the conversation, Strategy 2 allowed the system to put forward and develop its own argument in favor of the matter of discussion. An argument to be presented was selected randomly from the list of possible arguments that had been derived from seed corpus analysis. The selected argument, then, was considered to be the *current topic* of the conversation until the user or the system changed the topic. On the one hand, *topic change by the user* would be detected if the user spoke a long utterance that contained a detectable keyword; such an utterance would mean that the user presented an argument against the matter of discussion, thereby changing the topic. In such cases, the system would revert to Strategy 1 in generating its responses to the test taker. On the other hand, *topic change by the system* would be initiated if or when the system exhausted all relevant templates pertaining to the current topic, so it could no longer sustain a conversation about this topic. (Note that since Strategy 2 kept track of information about the current topic, it was a *stateful* strategy, as opposed to Strategy 1, which was *stateless* as it was based solely on reacting to the test taker’s most recent utterance.)

While the topic remained current, the DM script would execute the following sequence. First, if the test taker’s most recent turn (i.e., before the current topic was selected by the system) was a long utterance, then the system would begin by using an explicit topic-changing template like “I think I see what you mean. I want to move on to a somewhat different point, such as {topic}.” Here, “{topic}” was replaced by a noun phrase naming the topic selected by the system. A bank of five topic-changing templates was created for this purpose. As evident from the example just given, the templates of this kind were neither topic-specific nor prompt-specific. The reason why it was important to provide the test taker with an explicit signal of topic change was that, without such a signal, the logical flow of the conversation might be disrupted.

However, if the test taker’s most recent turn was a short utterance, then an explicit change of topic would not be needed and the system would move directly to the next step, which was to provide a general statement of the argument without referring to personal experiences. Afterward, as long as the conversation stayed on the current topic, the system would be providing more statements that used the system’s fictitious “personal experiences” to provide specific support for the argument being made.

Thus, for each of the arguments in favor of the matter that were identified through the analysis of the seed corpus, two banks of templates were developed: (a) a bank of at least two general utterances, presenting the argument without reference to personal experience, and (b) a bank containing at least one specific utterance citing the system’s “personal experience” in support of the argument. For example, for the argument that group work encouraged interacting with new people, one general utterance was “Don’t you think that group work is a great opportunity for interacting with new people?” and one specific utterance was “Last year when I was assigned a group project I was upset at the beginning, but then I met some great people that I didn’t know, and I became good friends with them.”

Strategies 1 and 2 were always used in conjunction, with the choice between the strategies based on the analysis of the test taker’s turn. Strategies 3, 4, and 5, described below, were added specifically to enhance the ability of the system to elicit features of interactional competence from the test taker. Thus, they were not developed based on the analysis of the seed corpus, but rather implemented according to the specifications of interactional competence features in the present assessment project.

Table 2 Regular Expressions (in PCRE Syntax) for Prompt 1

| Keyword | Meaning | Regular expression |
|--------------|--|---|
| CONFLICT | Group work can lead to conflicts among group members | /conflict/ |
| INDEPENDENCE | There are benefits of working independently | /independ alone on (your my) own by \w+self/ |
| DEADLINES | Group work may make it challenging to meet deadlines | /deadline/ |
| SCHEDULING | Group activities are difficult to schedule | /schedul/ |
| BAD_PEOPLE | Groupmates may turn out to be “bad” (unhelpful) people, some group members might be doing all the work, etc. | /(bad stupid incompetent inapt annoying slow irresponsible distracting)(person people student peer classmate)/ /(person people student peer classmate)\w*(who that which)/ /\bdo(ing es) (nothing everything all most)\b/ |
| DISTRACTING | It’s easier to get distracted while doing group work | /(off focus off track distract socializ youtube social media)/ |
| WASTE_TIME | Group work is a waste of time | /waste/ && /time/ /(much lots? use) of((the their his her my your))? time/ |

Strategy 3. Interrupting the Test Taker

On the technical level, the strategy of interrupting the test taker was implemented by passing an instruction from the DM script to the server side of the delivery framework, specifying the number of seconds or words in the user’s utterance after which the utterance should be interrupted and control should be passed back to the DM script. Time-based interruptions were implemented by using a countdown timer within the delivery framework that would be started after the ASR engine reported that the test taker had produced the first recognizable word in the utterance. Word-based interruptions would be triggered after the ASR engine reported that the test taker had produced the specified number of recognizable words.

The interruption strategy was deployed at Turns 3 and 7. At Turn 3, the system would interrupt the test taker at 1 s after the onset of their utterance and produce the following turn: “Sorry, could you please speak louder?” At Turn 7, the interruption would occur after the test taker spoke the first 10 words of their utterance, and the system would use a template from a bank, for example: “Sorry, I didn’t understand what you just said.” This latter version of the interruption would not happen if the test taker’s Turn 7 was a short utterance. In this case, Strategy 2 would be deployed for responding to the short utterance, and the interruption strategy would be deferred until the subsequent turn.

Strategy 4: Producing an Unintelligible Utterance

Regardless of the test taker’s input, the system’s Turn 5 was drawn from a bank of templates that were intentionally developed as partially unintelligible and thus creating an opportunity for the test taker to ask for a clarification. Each of these templates contained two nonce words (each repeated twice) that followed the phonotactic patterns of English. The nonce words were generated by recombining letter *n*-grams from a large dictionary using a script written by the first author. An example of such an unintelligible utterance is “I think that the **agliteration** of **fairfowlets** is crucial for success in group work. What do you think about the **agliteration** of **fairfowlets**?” (the nonce words are in bold).

The test taker, then, was expected to ask a clarification question. The test taker’s response was, however, not analyzed by the DM script, and the system’s Turn 6 was always generated using the template: “What I mean is that {topic} can be really important in group work” (for Prompt 1), where “{topic}” was replaced by the noun phrase corresponding to the currently active topic of conversation that had been selected following Strategy 2. If no topic was currently active, then a topic was chosen at random.

Strategy 5: Unexpectedly Changing the Topic

This strategy was implemented at Turn 8 by using the following template (for Prompt 1): “I just realized that when you engage in group work, you are encouraged to interact with others, and the nature of that interaction may allow you to discover things about yourself that you wouldn’t be aware of otherwise. So, in a sense, group work might serve as a window into your own cognition and personality. What is your take on that?” This template was written to make sure that it would not align with any topic that the interlocutors would be expected to be discussing, so that the topic change was always unexpected. A similar template was written for Prompt 2.

Question Avoidance Strategy

A notable limitation of the five dialog management strategies outlined above is that they would not assist the SDS in answering the test taker’s questions logically and coherently. NLU was only involved in implementing Strategy 1 and was limited to identifying keywords corresponding to arguments that the test taker was expected to make against the matter of discussion. Thus, a “question avoidance strategy” was required. This was implemented by writing all templates in a way that would sound reasonably coherent in response to both a statement and a question from the test taker. Specifically, words like “yes” and “of course” were avoided at the start of all templates because they might be inappropriate after a yes/no question. Instead, more neutral segues like “I think I see what you mean” and “All right” were utilized.

System Evaluation

ICE was evaluated in an experiment with participants representative of the target population of test takers. Forty-four international students at Iowa State University took the test in two conditions: in the human condition, the test takers were paired for the oral discussion task, which is “business as usual” for the EPT OC assessment; in the ICE condition, the test takers were speaking with ICE instead of a human interlocutor. The group of participants was diverse in regard to first language background, year in school (first-year undergraduate to advanced graduate student), gender (21 females and 23 males), country of origin, and English language proficiency (intermediate [*TOEFL iBT*[®] scores as low as 71] to advanced [students had obtained university degrees in English-medium universities]). All consented to participate in the study and were compensated for their efforts.

All participants completed the test task in the human and ICE conditions (which were randomized to different prompts), a 12-item questionnaire, and a semistructured interview. Transcripts and audio recordings of all system-generated and test-taker utterances were collected. Interview data was available for 44 participants. Two of the participants did not complete the questionnaire, and their transcript and audio-recording data were also excluded. These two types of data were, therefore, available for 42 out of 44 participants. A sample transcript of a conversation with one participant is presented in Appendix A.

A data set of all system-generated utterances produced in the 42 recorded dialog sessions was created. The ICE system always started the conversation with a predetermined utterance that was the same for each prompt. These utterances were excluded from the analysis that is reported below, resulting in a final data set of 525 system-generated utterances (an average of 12.5 per dialog session).

The data set was manually reviewed by a research assistant who took part in the development of the system. Failures of ASR (i.e., incorrectly recognized words) were manually coded, and in all such instances, manual orthographic transcription of the words actually spoken by the test takers were performed. Failures of NLU (i.e., instances where regular expressions misidentified keywords in the utterances) were coded in a similar fashion. Finally, each system-generated utterance was coded as useful versus not useful for eliciting scorable discourse from the test taker. The utterances coded as not useful were the ultimate DM failures. Because this final coding relied on subjective judgment, it was independently repeated by a second coder who was also a member of the research team but did not participate in the development of ICE.

The codes were analyzed, in combination, to provide insight into the performance of the system, the sources of system failures, and the possibility of addressing these failures in an updated version of the system.

Test takers’ perceptions of ICE, in comparison with the human interlocutor, were evaluated using a 12-item questionnaire and a semistructured interview. Immediately after completing each testing condition (ICE or human), the test takers

completed a five-point Likert item questionnaire (Appendix B). Descriptive statistics were used to analyze trends in the data. After completing both testing conditions and the two questionnaires, test takers met individually with one of the data collectors and completed a video-recorded semistructured oral interview. The interviews were manually coded in NVivo 12 according to the following main themes: (a) which partner test takers preferred, (b) which partner—human or computer—would be better for a “real” assessment, and (c) the pros and cons of ICE and a human partner on the test.

Results

System Performance Evaluation

As evident from the technical description of the system provided in the previous section, the NLP pipeline where failures could occur included the ASR, NLU, and DM components. Each downstream component in this pipeline was designed to be robust to failures of the upstream component; however, the “garbage-in-garbage-out” effect (i.e., a situation where an upstream failure directly caused a downstream failure; e.g., NLU failed due to inaccurate ASR) was still expected as with any system that involves NLP.

System failures were counted based on the codes assigned during the manual review of the data set. Cause-and-effect relationships between the failures of individual components were established on a case-by-case basis through the expert judgment of the first coder and the first author of the paper, both of whom took part in system development. A summary of failure counts is presented in Table 3.

Failures Originating in ASR

ASR failed to recognize at least one word correctly in 266 of the 525 utterances (51%). However, these ASR failures only affected the performance of NLU in 19 cases, and the ultimate performance of DM in 4 cases. This is because our approach to implementing NLU and DM was robust to upstream failures: NLU only reacted to certain keywords and disregarded the rest of the message, whereas DM implemented multiple strategies that did not rely on ASR or NLU. Thus, out of the 266 instances of initial ASR failure, only four (1.5%) resulted in an issue with the overall system performance that propagated down to the DM stage. These four cases are described in detail in the following paragraphs.

In one of these cases, the ASR misunderstood the test taker’s final words in the utterance as “free shipping” (Dialog 7594, Turn 12C). This understanding was clearly an ASR failure because the context of conversation (Prompt 2) had nothing to do with shipping or the cost thereof. However, when members of our research team listened to the recording of this utterance, they could not understand the test taker’s words either and transcribed them as <unintelligible>. Thus, this failure of ASR does not suggest its poor performance relative to the human gold standard.

In two other cases, the ASR mistakenly recognized the word “cost” when the test taker actually said the word “class.” Finally, in one case, the words “the conflict” were incorrectly recognized as “the time that” (Dialog 8272, turn 15C).

Failures Originating in NLU

These system failures occurred when regular expressions either did not detect a relevant keyword (resulting in a “miss”) or flagged a keyword that was not present in the test taker’s utterance (resulting in a “false alarm”). NLU was considered the first point of failure in the processing pipeline only if the NLU failure was not causally related to an upstream ASR failure. Specifically, an ASR misrecognition in a part of the utterance other than that which was directly targeted by a regular expression was not considered to be the cause of the failure of that regular expression.

There was a total of 56 failures originating in NLU, but only two (4%) of them resulted in an issue with the overall system performance because the DM implementation was robust to NLU failures. The two failures are discussed in detail below.

In one of these cases, the regular expression that was intended to match the word “free” (in relationship to the cost of gym classes vs. independent physical exercises in Prompt 2) was written without regard to the right-hand word boundary, so it also matched the word “freedom” in the context “freedom to control their body” (Dialog 4157, Turn 14C). This match triggered a system response related to the cost of taking a gym class, which was irrelevant to the current context of conversation. This issue can be easily addressed by revising the problematic regular expression (adding right-hand word boundary) in the next version of the system.

Table 3 Frequencies of System Failures

| First point of failure in the pipeline | ASR failures | NLU failures | DM failures |
|--|--------------|--------------|-------------|
| ASR | 266 | 19 | 4 |
| NLU | | 56 | 2 |
| DM | | | 46 |

Note. ASR = automated speech recognizer; NLU = natural language understanding; DM = dialog manager. Number of utterances, $N = 525$.

In the other case, the test taker's utterance was not captured by the regular expression for the keyword BAD_PEOPLE for Prompt 1: "not necessary if some people are more they don't have a good work ethic that could be dragged down the ability to get things done" (Dialog 1949, Turn 3C). As a result, the SDS behaved like someone who did not understand their interlocutor but pretended that they knew what was going on instead of asking for clarification. This issue is harder to address as the test taker's utterance does not include an easily detectable word pattern that could be classified as referring to BAD_PEOPLE.

Failures Originating in the DM

These were failures specific to the chosen implementation of the DM component of the system. The coding of DM failures was performed independently by two coders as detailed in the Methods section. Both coders agreed on assigning the DM failure code to seven utterances. In addition, the second coder also flagged 49 DM failures that were not flagged by the first coder. The first author of the present paper reviewed and adjudicated these 49 instances of disagreement between the original coders, and in 39 of these cases, the DM failure code was assigned. All in all, this process resulted in a set of 46 instances of DM failures, which are further analyzed in the following paragraphs.

The most frequent cause of DM failure was the production of irrelevant segues, which happened in 21 cases. For example, the system would say "I see your point" when the test taker did not make a substantive point or ask "What does that mean?" in response to a simple "yes." This shortcoming can be mitigated in the next version of the system by rewriting the banked templates and adjusting their selection process based on the length of the test taker's utterance. Additionally, in seven cases, the system did not react to the test taker's requests to repeat the previous utterance (e.g., "Can you repeat what you just said?"). Responses to such requests would have been easy to implement; however, they were overlooked by the authors in the process of system development.

In 12 instances, the system did not allow the test taker to finish their utterance and misinterpreted a long hesitation pause as the end of the turn. This process was an apparent limitation of the ASR system that was relied upon for detecting the end of the turn. In four instances, the system failed to respond to the test taker's direct question, resulting in a disruption of the conversation flow. For example, in Dialog 4157, the system said "What I mean is that being forced to do things that are good for us can be really important" (Turn 7C), to which the test taker responded with "Why do you think like that?" (Turn 8U). The system's response, however, was "I think I see what you mean. I want to move on to a somewhat different point, such as the advantage of meeting new people in gym class." (Turn 8C). The decision to avoid processing and responding to test taker's questions was deliberate, and the few instances of suboptimal question avoidance may be considered an acceptable trade-off.

Each of the following two issues only appeared one time: (a) the system switched topics two times in a row due to lack of templates in the bank (this can be addressed by writing more templates); and (b) when a test taker chose to side with the system on one of the arguments, the system restated much of what the test taker just said (this resulted from the combination of somewhat unexpected test taker's behavior and coincidence).

User Experience Evaluation

The results from the 42 test takers on the 12-item questionnaire are presented in Figure 1. In responding to the questionnaires, the test takers were generally positive about all aspects of the test under both the human and ICE conditions. Almost all categories had values of 3 or higher on the 5-point Likert scale items. It can also be seen that the test takers evaluated

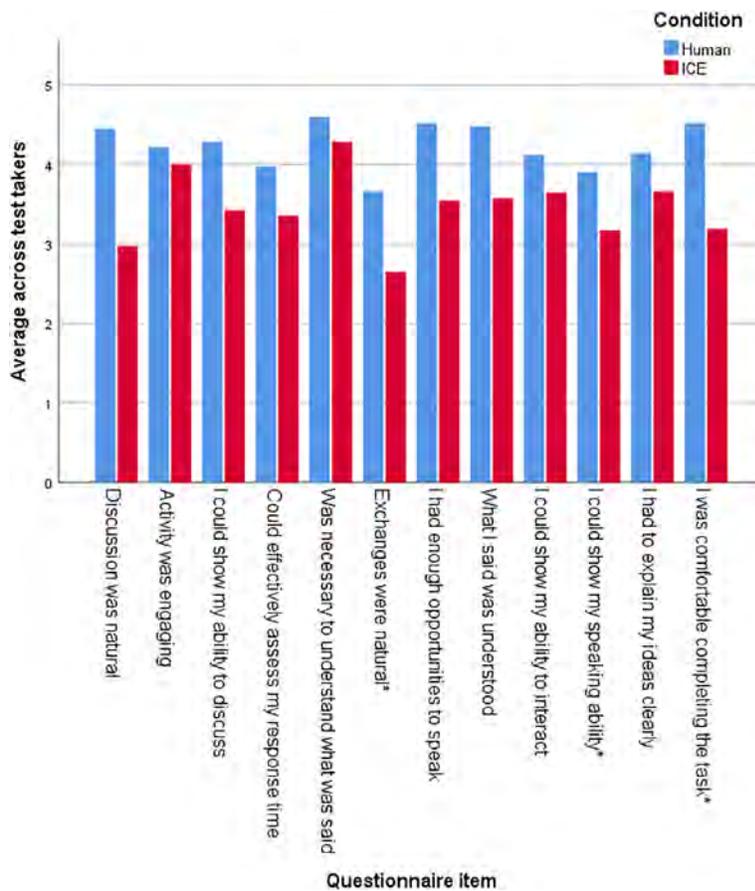


Figure 1 Test-taker perceptions of ICE versus a human partner based on questionnaire data. *Note.* An asterisk indicates items were reversed.

the human partner condition more positively than ICE across the 12 items. Items that were reversed are indicated with an asterisk. The results from the semistructured interviews with test takers provide insight into the reasons for test takers’ opinions about ICE compared to a human partner. All 44 test takers completed the interviews after taking the test.

Preferred Partner for a Speaking Test

When asked which partner they preferred when taking a speaking test, 18 test takers preferred speaking with ICE, 18 preferred speaking with a human, and eight indicated no preference. Twenty test takers thought grammar and vocabulary were better assessed with ICE. Thirteen test takers felt that ICE was better for assessing pronunciation, and 12 thought it was better for assessing fluency because they had to respond more quickly. Eleven test takers thought that ICE understood them better, seven felt that ICE required them to listen carefully, five appreciated that ICE was standardized, and three felt ICE was more authentic because it was formal and at an appropriate difficulty level.

Reasons for preferring a speaking test with a human partner included the following: naturalness of situation (15), partner could accommodate to their level of fluency by slowing down and allowing longer response time (11), presence of body language (5), and easier to talk to someone with similar ability (3).

Pros and Cons of ICE as a Partner

Pros of having ICE as a partner included the following: it’s easier to understand ICE (19), ICE has more ideas (14), it’s more comfortable speaking with ICE (6), ICE speaks at an appropriate speed (6) and difficulty level (5), ICE is convenient because you do not need another human (4), and ICE is standardized (3).

Negative comments about ICE included the following: ICE does not always understand what the test taker says (12), the conversation is not completely natural (12), ICE interrupts too much (10) and speaks too fast (4), and it is strange speaking with a computer (4) and difficult to converse without body language (3).

Pros and Cons of a Human as a Partner

Test takers felt that a human partner is an advantage in regard to naturalness (18) and body language (9) to aid comprehension, ease of exchanging ideas (7), and a human partner's ability to understand better (6).

On the negative side, 18 test takers felt that a human can be more difficult to understand than ICE, while 13 felt that a human partner could be at a different ability level. This difference could mean that their partner is at a higher level, which would lower their confidence, or that their partner is at a lower level, which could lower their score. Six test takers said they might feel uncomfortable with a human speaker because their partner judges their language ability. Four said that conversations with a humans are often repetitive, and four felt that either they or their partner may not share allotted time evenly.

Discussion and Conclusion

The results are, overall, positive. ICE was able to generate task-appropriate (i.e., useful for eliciting the test takers' interactional competence features) utterances in 90% of cases. Only 1% of utterances represented system failures that originated in the ASR and NLU, while 9% of utterances failed to be useful due to issues with DM. Of these, almost half constituted the production of irrelevant segues, such as saying "I see your point" when reacting to a very short utterance by the test taker that does not make any substantive point that would license such a confirmation. These failures can be mitigated by rewriting the bank of segues and differentiating them based on a simple yet robust parameter: the length of the test taker's previous utterance. Other issues with the performance of the system, similarly, can be addressed in a future version. Even without these improvements, however, the system was very successful at maintaining a task-appropriate dialog with the test taker.

Notably, the performance of a cloud-based commercial ASR (Google Speech-To-Text) was sufficient for the successful implementation of ICE. The benefits of using an off-the-shelf, cloud-based ASR include low deployment and maintenance costs and robust performance. The overall latency of the system was not negatively perceived by the test takers. In addition, the rule-based approach to developing the DM script proved successful for the task at hand. Keyword spotting by regular expressions, the key procedure in the NLU component of the system, was largely implemented by two research assistants on the team (students in linguistics who had no computer science expertise). This finding points to the potential of rapid, low-cost development of similar systems for other prompts in the future. The skills that the research assistants utilized amounted to simple, qualitative discourse analysis and writing regular expressions.

The robustness of the system (and especially its ability to recover failures within its processing pipeline) is largely explained by the alignment of the system's DM strategies with the assessment task. Unlike typical goal-oriented systems, it is not essential for ICE to accomplish a real-life task successfully. Notably, the performance of the system was not evaluated based on its ability to accomplish the goal of "winning the debate" with the test taker by "convincing" them to change their viewpoint, which might be a relevant goal in a real-life debate. In the context of an oral language assessment, however, the operational goal of the system was *discourse elicitation* from the test taker. Thus, simplifications in the NLU and NLG components, which would have been detrimental for a typical goal-oriented SDS, proved to be acceptable (and productive) trade-offs in the case of ICE. In fact, some of the failures of the system components were ultimately helpful for attaining its goal. For example, whereas a failure in the ASR or the NLU might lead to a situation of "misunderstanding," demonstrating the ability to detect and repair misunderstandings in a conversation is, in fact, necessary for the test taker to receive an appropriate score for interactional competence on the test.

The proposed design and implementation of the system, while successful as an initial rapid prototype, is not without limitations. Unlike HALEF (Ramanarayanan et al., 2015), the underlying delivery framework in ICE only allows for web-based delivery and is not ready for large-scale deployments. The use of TTS instead of voice-talent recordings may have contributed, in part, to the perceived unnaturalness of the system. The keyword-spotting approach, while robust for the present purposes, does not account for the intricacies of semantics or the context of speech. Score data from an external test were not available for comparing how test takers with differing ability levels performed with the system. Hopefully,

despite these limitations, the present report lays out groundwork that may be useful for future developers of specialized SDSs for assessing oral communication ability.

Acknowledgments

This research was funded by Educational Testing Services (ETS) under a Committee of Examiners and the Test of English as a Foreign Language research grant. ETS does not discount or endorse the methodology, results, implications, or opinions presented by the researcher(s). The authors are grateful to research assistants Nazlınur Göktürk Tüney and MacKenzie Novotny for their contributions to the development of the SDS presented in this paper.

Notes

- 1 <https://github.com/chukharev/sds>
- 2 <https://nodejs.org/>
- 3 <https://expressjs.com/>
- 4 <https://ffmpeg.org/>
- 5 <https://cloud.google.com/speech-to-text/>
- 6 Latest publicly reported word error rate 4.9% (e.g., <https://venturebeat.com/2017/05/17/googles-speech-recognition-technology-now-has-a-4-9-word-error-rate/>).
- 7 <https://aws.amazon.com/polly/>

References

- Bibauw, S., François, T., & Desmet, P. (2019). Discussing with a computer to practice a foreign language: Research synthesis and conceptual framework of dialogue-based CALL. *Computer Assisted Language Learning*, 32(8), 827–877. <https://doi.org/10.1080/09588221.2018.1535508>
- Council of Europe. (2001). *Common European framework of reference for languages: Learning, teaching, assessment*. Cambridge University Press.
- Graesser, A., & McDaniel, B. (2017). Conversational agents can provide formative assessment, constructive learning, and adaptive instruction. In C. A. Dwyer (Ed.), *The future of assessment* (pp. 85–112). Routledge. <https://doi.org/10.4324/9781315086545-4>
- Graesser, A. C., VanLehn, K., Rose, C. P., Jordan, P. W., & Harter, D. (2001). Intelligent tutoring systems with conversational dialogue. *AI Magazine*, 22(4), 39–51. <https://ojs.aaai.org/index.php/aimagazine/article/view/1591/1490>
- Litman, D., Young, S., Gales, M., Knill, K., Ottewell, K., van Dalen, R., & Vandyke, D. (2016). Towards using conversations with spoken dialogue systems in the automated assessment of non-native speakers of English. In R. Fernandez, W. Minker, G. Carenini, R. Higashinaka, R. Artstein, & A. Gainer (Eds.), *Proceedings of the 17th annual meeting of the special interest group on discourse and dialogue* (pp. 270–275). Association for Computational Linguistics. <https://doi.org/10.18653/v1/W16-3635>
- Litman, D., Strik, H., & Lim, G. S. (2018). Speech technologies and the assessment of second language speaking: Approaches, challenges, and opportunities. *Language Assessment Quarterly*, 15(3), 294–309. <https://doi.org/10.1080/15434303.2018.1472265>
- Ockey, G. J. (2009). The effects of group members' personalities on a test taker's L2 group oral discussion test scores. *Language Testing*, 26(2), 161–186. <https://doi.org/10.1177/0265532208101005>
- Ockey, G. J. (2017). Approaches and challenges to assessing oral communication on Japanese entrance exams. *JLTA Journal*, 20, 3–14. https://doi.org/10.20622/jltajournal.20.0_3
- Ockey, G. J., & Chukharev-Hudilainen, E. (2021). Human versus computer partner in the paired oral discussion test. *Applied Linguistics*, 2021, 1–61. <https://doi.org/10.1093/applin/amaa067>
- Ramanarayanan, V., Suendermann-Oeft, D., Ivanov, A. V., & Evanini, K. (2015). A distributed cloud-based dialog system for conversational application development. In A. Koller, G. Skantze, F. Jurcicek, M. Araki, & C. Penstein Rose (Eds.), *Proceedings of the 16th annual meeting of the special interest group on discourse and dialogue*. Association for Computational Linguistics. <https://doi.org/10.18653/v1/w15-4658>
- Ramanarayanan, V., Evanini, K., & Tsuprun, E. (2019). Beyond monologues: Automated processing of conversational speech. In K. Zechner & K. Evanini (Eds.), *Automated speaking assessment* (pp. 176–191). Routledge. <https://doi.org/10.4324/9781315165103-11>
- Timpe-Laughlin, V., Evanini, K., Green, A., Blood, I., Dombi, J., & Ramanarayanan, V. (2017). Designing interactive, automated dialogues for L2 pragmatics learning. In V. Petukhova & Y. Tian (Eds.), *Proceedings of the 21st Workshop on the Semantics and Pragmatics of Dialogue* (pp. 143–152). SEMDIAL. http://semdial.org/anthology/Z17-Timpe-Laughlin_semdial_0018.pdf

Appendix A

Sample Conversation of a Test Taker with ICE

(Dialog ID: 1280. Prompt 1.)

Notes. “C” indicates the system’s turn; “U” indicates the test taker’s turn. Inline comments are in angle brackets. The transcription is verbatim.

1C: I believe that group work is very beneficial because it helps people improve their communication skills and also helps them to somehow learn the content better.

2U: I disagree with this, because sometimes people are not that much responsible and who do their job correctly and the that’s my, that’s causing result in the getting some privileges of, for the things that they have not done in the group. They just use others’ effort to get the credit from that work, or although they have not done much.

2C: I think I see what you mean. I want to move on to a somewhat different point, such as creativity.

3U: I do not agree with this creativity, because sometimes there are always in each group of people there are some people who are not familiar with some brainstorming and dealing with new ideas might kill very creative ideas. But if you are working on your own, you can you can easily pursue your objectives and your goals and your creative ideas.

3C: Working independently is a useful skill, but working with groups is more important because teamwork is so important for careers later on.

4U: I do not <interruption>

4C: Sorry, could you please speak louder?

5U: I do not see any positive point regarding the future job because you are evaluated for your own, when you want to apply for a job, and.

5C: That’s fair. Would you like to talk about improving communication skills?

6U: Yeah we can.

6C: I think that the aglitteration of fairfowlets is crucial for success in groupwork. What do you think about the aglitteration of fairfowlets?

7U: Excuse me, I did not get what you meant. I mean, can you rephrase what you said?

7C: What I mean is that improving communication skills can be really important in group work.

8U: I do agree with this, but sometimes this is very time-consuming, too. Sometimes it’s a waste of time. When people had to communicate and, as I said, some people are not very responsible and professional for doing their jobs on job, and you might waste your time communicating with them and at the end you will gain nothing.

8C: Group work actually reduces the amount of time you spend on a project because each person only has to do a smaller task.

9U: I am not agree with this. I do not agree with.

9C: Sorry, I did not understand what you just said.

10U: I do not agree with this, because I think that’s.

10C: I just realized that when you engage in group work, you are encouraged to interact with others, and the nature of that interaction may allow you to discover things about yourself that you would not be aware of otherwise. So, in a sense, group work might serve as a window into your own cognition and personality. What is your take on that?

11U: I do agree with you because always communication helps you to improve your personal characteristics, but sometimes you are not, you do not want to work on your own characteristics, you just want to do a project and you have limited time, so, so.

11C: OK, let us talk about something else. I believe interacting with new people is really important.

<The conversation is stopped at this point by the data collector.>

Appendix B Questionnaire

| Test taker | 1 | 2 | 3 | 4 | 5 |
|--|-------------------|----------|---------------------------|-------|----------------|
| Discussion with a human | Strongly disagree | Disagree | Neither agree or disagree | Agree | Strongly agree |
| 1_1. I felt like the discussion was natural | | | | | |
| Comments: | | | | | |
| 1_2. The activity was engaging/interesting | | | | | |
| Comments: | | | | | |
| 1_3. I was able to demonstrate my ability to discuss a topic | | | | | |
| Comments: | | | | | |
| 1_4. I was required to respond quickly, like is expected in a natural discussion | | | | | |
| Comments: | | | | | |
| 1_5. It was necessary to understand what my partner said to complete the task | | | | | |
| Comments: | | | | | |
| 1_6. There were unnatural speaking exchanges because of my partner's limited English ability | | | | | |
| Comments: | | | | | |
| Test taker | 1 | 2 | 3 | 4 | 5 |
| Discussion with a human | Strongly disagree | Disagree | Neither agree or disagree | Agree | Strongly agree |
| 1_7. My partner gave me enough opportunities to say what I wanted to | | | | | |
| Comments: | | | | | |
| 1_8. My partner seemed to understand what I said | | | | | |
| Comments: | | | | | |
| 1_9. I was able to show my ability to interact/communicate in English | | | | | |
| Comments: | | | | | |
| 1_10. I was NOT able to show my true speaking ability in the discussion | | | | | |
| Comments: | | | | | |
| 1_11. My partner effectively made sure I explained my ideas clearly | | | | | |
| Comments: | | | | | |
| 1_12. I was NOT comfortable completing the task because of the person I was talking with | | | | | |
| Comments: | | | | | |

Suggested citation:

Chukharev-Hudilainen, E., & Ockey, G. J.. (2021). *The development and evaluation of Interactional Competence Elicitor for oral language assessments* (TOEFL Research Report No. RR-92). ETS. <https://doi.org/10.1002/ets2.12319>

Action Editor: John Norris

Reviewers: This report was reviewed by the Research Subcommittee of the TOEFL Committee of Examiners.

ETS, the ETS logo, and TOEFL IBT are registered trademarks of Educational Testing Service (ETS). All other trademarks are property of their respective owners.

Find other ETS-published reports by searching the ETS RESEARCHER database at <http://search.ets.org/researcher/>