Int J Artif Intell Educ (2014) 24:284–332
DOI 10.1007/s40593-014-0017-9

BEETLE II: Deep Natural Language Understanding and Automatic Feedback Generation for Intelligent Tutoring in Basic Electricity and Electronics

**Myroslava Dzikovska · Natalie Steinhauser ·
Elaine Farrow · Johanna Moore ·
Gwendolyn Campbell**

Published online: 10 June 2014
© Springer Science+Business Media New York 2014

Abstract Within STEM domains, physics is considered to be one of the most difficult topics to master, in part because many of the underlying principles are counter-intuitive. Effective teaching methods rely on engaging the student in active experimentation and encouraging deep reasoning, often through the use of self-explanation. Supporting such instructional approaches poses a challenge for developers of Intelligent Tutoring Systems. We describe a system that addresses this challenge by teaching conceptual knowledge about basic electronics and electricity through guided experimentation with a circuit simulator and reflective dialogue to encourage effective self-explanation. The Basic Electricity and Electronics Tutorial Learning Environment (BEETLE II) advances the state of the art in dynamic adaptive feedback generation and natural language processing (NLP) by extending symbolic NLP techniques to support unrestricted student natural language input in the context of a dynamically changing simulation environment in a moderately complex domain. This allows contextually-appropriate feedback to be generated "on the

---

M. Dzikovska (✉) · E. Farrow · J. Moore
School of Informatics, University of Edinburgh, 10 Crichton Street, Edinburgh, EH8 9AB, UK
e-mail: m.dzikovska@ed.ac.uk

E. Farrow
e-mail: elaine.farrow@ed.ac.uk

J. Moore
e-mail: j.moore@ed.ac.uk

N. Steinhauser · G. Campbell
Naval Air Warfare Center Training Systems Division, Orlando, FL, USA

N. Steinhauser
e-mail: natalie.steinhauser@navy.mil

G. Campbell
e-mail: gwendolyn.campbell@navy.mil
url: http://groups.inf.ed.ac.uk/beetle

fly" without requiring curriculum designers to anticipate possible student answers and manually author multiple feedback messages. We present the results of a system evaluation. Our curriculum is highly effective, achieving effect sizes of 1.72 when comparing pre- to post-test learning gains from our system to those of a no-training control group. However, we are unable to demonstrate that dynamically generated feedback is superior to a non-NLP feedback condition. Evaluation of interpretation quality demonstrates its link with instructional effectiveness, and provides directions for future research and development.

Keywords Intelligent tutoring systems · Natural language processing · Tutorial dialogue

Introduction

Within Science, Technology, Engineering, and Math (STEM) domains, physics is considered by many to be one of the most difficult topics to master, in part because many of the underlying principles are counter-intuitive. Students' naïve conceptions have proven to be highly resistant to traditional lecture-based methods of instruction and this finding has been the impetus for a tremendous amount of research into alternative, more effective instructional techniques. One approach that has proven effective is called the conceptual change method (Duit and Treagust 2003). This approach relies on two key processes. First, students should be presented with unavoidable and incontrovertible evidence that explicitly contradicts their existing naïve conception(s). Second, they should be guided just enough to allow them to generate an accurate conceptualization of the phenomena and principles within the domain.

This approach is consistent with the basic principles of many learning theories. For example, the second stage relies on the tutor taking on a Socratic role (Seeskin 1987) and leverages what we know about student learning through post-activity reflection and self-explanation (Chi et al. 1994; Katz et al. 2003, 2007; King 1997; Kolb 1984; Lee and Hutchison 1998; Mestre et al. 2011). At its core, this stage recognizes the importance of getting students to generate or construct their own interpretation and understanding of a domain (Duffy and Jonassen 1992; Osborne and Wittrock 1983).

A consideration of the conceptual change approach indicates that an intelligent system capable of implementing this type of science instruction needs to include both a simulation environment for the first stage and a tutorial dialogue capability for the second. Interestingly, while there are many computer-based tutoring systems that use simulation as a training tool (e.g. de Jong and van Joolingen 1998) and tutorial dialogue systems that ask students to provide justifications for problem solving steps (Aleven et al. 2002; Khuwaja et al. 1994; VanLehn et al. 2007) and to explain their conceptual reasoning (Graesser et al. 1999; Jordan et al. 2006; Litman and Silliman 2004), relatively few existing systems combine experimentation in an interactive environment with self-explanation (Pon-Barry et al. 2004; Ros et al. 2004).

In this paper, we tackle the problem of developing an effective curriculum based on the conceptual change method within the context of an Intelligent Tutoring System (ITS) for a sub-topic of physics, electricity and electric circuits. We leveraged existing research on common misconceptions in this domain to create a curriculum with exercises conducted in a simulated circuit workbench, designed to contradict such misconceptions, and post-exercise guided reflective dialogues to help students formulate new conceptions. To our knowledge, the Basic Electricity and Electronics Tutorial Learning Environment (BEETLE II) is the first ITS that integrates the interactive experimentation and reflective dialogue components of the conceptual change approach, using natural language processing to understand student explanations and provide context-specific feedback.

Part of the challenge in building such a system is the requirement to deal with a dynamically changing simulation environment. Student utterances need to be interpreted in the context of the current state of the simulation, and tutoring feedback must also be adapted to the current context. This generally means that feedback must be generated "on the fly", which presents a serious challenge for the natural language processing (NLP) module in a tutoring system.

The BEETLE II system addresses this challenge by providing a natural language interpreter and a tutorial planner and generator that work together to provide dynamic feedback generation. The BEETLE II dialogue component supports context-specific interpretation and diagnosis of natural language student explanations, and implements a set of generic tutoring tactics that can be instantiated in different problem-solving contexts. This allows for dynamic selection and automatic generation of context-adaptive feedback to student answers. Our approach improves upon the state of the art in dialogue ITSs that support automatic feedback generation by tackling more complex language input, while still allowing for integration with an interactive simulation.

After reviewing previous work in NLP for tutorial dialogue, we describe the design and implementation of the BEETLE II system, including the construction of the curriculum and a system architecture to support dynamic feedback generation. We evaluate the overall system's effectiveness in terms of learning gain and user satisfaction and assess the contribution of the NLP module. We then carry out a detailed evaluation of the natural language interpreter, looking at its overall performance and how that correlates with learning outcomes. Finally, we describe an error analysis focusing on the different types of interpretation failures possible in the system, and their relationship with desired outcomes. Based on this evaluation, we discuss future directions for research and development.

Previous Work

It has been solidly established that eliciting self-explanation (Chi et al. 1994) and contentful talk (Litman et al. 2009; Purandare and Litman 2008) from students is correlated with increased learning gain. Therefore, there has been substantial interest in developing tutoring systems in STEM domains that are able to understand and provide feedback to natural language input.

Decisions about how to provide feedback during natural language dialogue can be seen in the more general context of providing formative feedback in an ITS. The types of feedback that an ITS can provide include simple verification feedback that confirms whether or not an answer is correct, correct response (or "bottom-out") feedback that gives away the correct answer, and different forms of elaborated feedback which includes giving general advice on the topic, re-teaching the material, flagging problematic parts of student responses, or giving hints, cues and prompts (Shute 2008). We will focus on the provision of "**informative tutoring feedback**" or "**tutoring tactics**" (Chi et al. 2009; Narciss and Huth 2004; Narciss 2004), that is, feedback that combines verification, error flagging and hints adapted to the context.

In this section, we review work on dialogue ITSs that provide feedback for relatively short (1 to 2 sentence) explanation questions asking students to explain their observations or justify problem-solving steps. Several existing tutorial dialogue systems focus instead on helping students to produce short essays in answer to an explanation-type question (Graesser et al. 1999; Jordan et al. 2006; Litman and Silliman 2004). These systems incorporate NLP approaches that are highly effective in assessing longer texts. However, methods suitable for essay assessment may not perform well when applied to shorter explanations (Ventura et al. 2004).

Systems that provide feedback on short explanations have used both statistical and symbolic methods for answer analysis. The statistical methods employ techniques from textual entailment and paraphrase identification to compare the text of the student answer against sets of possible correct and incorrect answers anticipated by system developers (Jordan et al. 2007; McCarthy et al. 2008; VanLehn et al. 2007). These methods are generally robust to unexpected student inputs, and require less upfront investment in NLP infrastructure than symbolic methods. However, since they rely on matching text strings rather than building semantic representations, they use finite-state machines for dialogue management and require system developers to manually author feedback messages for every machine state. This requires careful attention to authoring tools to avoid redundancy that can cause student confusion (Jordan et al. 2005), and to deal with unanticipated answers which in one study accounted for 30 % of student utterances (Jordan et al. 2009).

In contrast, symbolic methods involve using rule-based parsers and reasoners to build semantic structures representing fine-grained details of utterance content. The availability of such detailed representations supports dynamic adaptive feedback generation because it allows a system to maintain a library of generic tutoring tactics and then choose a tactic at each point of the interaction according to a particular tutoring policy, instantiating it with relevant content from the student answer (Callaway et al. 2006; Glass 2000; Pon-Barry et al. 2004). Symbolic NLP approaches are more brittle to unexpected student input than statistical approaches, and require more upfront investment in parsing and interpretation infrastructure to develop and deploy in new domains. However, the structured semantic representations they produce offer particular advantages for integrating tutoring with simulation-based environments or with environments where problems are dynamically generated, such as cognitive tutors (Aleven et al. 2002). These representations allow such systems to offer feedback that explicitly refers to the current state of the simulation or problem-solving situation.

The development of the BEETLE II system is an attempt to expand the application of the symbolic NLP approach to support dynamic feedback generation in conjunction with a curriculum that embodies best practices in science teaching and requires support for moderately complex natural language input. Compared to other systems that support dynamic feedback generation, the BEETLE II system allows for significantly more complex input than the CIRCSIM-Tutor (Glass 2000), which asks questions requiring 1–2 word answers; and BEETLE II supports explanations of about the same average length as the SCoT-DC tutor (Pon-Barry et al. 2004), but attempts to handle a wider range of topics (SCoT-DC covers three main topics, whereas the BEETLE II curriculum covers 10 main topics, with 14 exercises involving around 150 different natural language questions asked by the system).

Another system that supports symbolic understanding and reasoning about student explanations is the Geometry Explanation Tutor (Aleven et al. 2002), which aims to give feedback on sentence-long explanations for geometry problems. However, unlike BEETLE II, in order to generate feedback it represents all possible partially correct explanations in its domain ontology. The BEETLE II system aims to implement a set of generic tutoring tactics (discussed in the "System Implementation" section) which does not require possible incorrect answers to be specified in advance as concepts in the ontology.

Among the systems that use statistical NLP approaches to understanding, the closest in philosophy to BEETLE II is the AUTOTUTOR system (Graesser et al. 1999). The tutoring in AUTOTUTOR is based around expectations of the answer, expressed as natural language statements. Student input is matched against those expectations using latent semantic analysis. For each missed expectation, the instructional designers author a list of feedback messages that correspond to different tutoring tactics: pumps (or "contentless prompts"), hints and assertions. This provides the system with a collection of tactics it can choose from at each point in the interaction according to the desired tutoring policy. However, since no structured representation is used, the feedback messages cannot be generated automatically and have to be authored for each problem.

The BEETLE II domain was chosen to be more restricted than the AUTOTUTOR domains, so that it allows for building domain reasoners and symbolic models, but is more complex than the systems that use symbolic approaches discussed above.[1] One of our research aims was to investigate whether we can effectively handle more complex language by using a domain adaptation technique originally developed for task-oriented spoken dialogue systems. In contrast to previous approaches to symbolic NLP in tutoring (which rely either on entirely domain-specific processing modules, or on syntactic parsers with domain-specific lexicons), our architecture uses a wide-coverage syntactic parser and a domain-independent lexicon and ontology that is adapted to a given domain with the aid of ontology mapping rules. This

---

[1] Any comparisons based on the number of topics or size of the domain ontology are approximate, because there can be substantial differences in granularity which are difficult to quantify. However, we tried to ensure that the numbers as cited provide a fair comparison based on our understanding of the referenced systems.

architecture has been successfully used to improve portability and robustness of parsing and semantic interpretation in task-oriented dialogue systems (Dzikovska et al. 2008). This gave us reason to believe that it can effectively support understanding explanations in a tutorial dialogue system that combines conceptual learning with experimentation in a simulation environment.

Next, we provide a general description of the curriculum, the high-level tutoring strategies used in the system, and the role that NLP plays in supporting them.

Curriculum Development

As mentioned earlier, physics is considered by many to be an extremely difficult field to master, due in large part to the inclusion of many concepts that appear to be counter-intuitive to our everyday experiences. We focused our curriculum on the sub-topic of electricity and electric circuits. This is a foundational topic for electrical engineering and electricians, and a common part of school physics curricula in order to enable students to understand the electrical appliances and machinery they encounter in everyday life. Our research goal, however, is not to teach circuits per se, but to understand how teaching techniques involving hands-on interaction and reflective dialogue, known to be effective when used by human tutors, can be implemented in a computer system integrating simulation and natural language dialogue.

In order to develop our learning materials, we leveraged the existing research in science education as much as possible. At the heart of over two decades of research in physics education we find a number of studies designed to elicit and identify common misconceptions (Chinn and Brewer 1993; Halloun and Hestenes 1985a; Tallant 1993). From there, researchers went on to develop psychometrically robust tests of conceptual understanding to support the body of research evaluating the effectiveness of a wide variety of different teaching strategies (Carey 1986; Eylon and Linn 1988; Farnham-Diggory 1994; Halloun and Hestenes 1985b; Scott et al. 1992; Vosniadou and Brewer 1987). By tying into this body of research, we were able to establish learning objectives that align with an established set of common misconceptions, apply an instructional strategy that has been demonstrated to be effective, and evaluate our system using tests and conceptual inventories that are known to be reliable and valid.

In the rest of this section, we will elaborate on the specifics of the course that we implemented in BEETLE II, at three levels. We begin by addressing the content, i.e. the specific concepts that our curriculum addresses and how those concepts are sequenced. Next, we address the primary high-level instructional strategy that was used to deliver this content. Finally, we briefly explain how we arrived at the low-level tutoring tactics implemented in the system in order to help a student complete each step required by the instructional strategy.

Instructional Content and Approach

We implemented approximately four hours of instruction, using content and activities drawn from the materials published by the Physics Education Group at the University of Washington (McDermott et al. 1995). These materials are based on years of research in science education conducted by this group and others, and have been iteratively refined through multiple experiences in college classrooms. The curriculum explicitly covers topics such as open and closed paths, voltage, series and parallel circuits, and finding faults in a circuit with a multimeter. In addition, the simulation-based exercises themselves are carefully designed to elicit and counter common misconceptions in this domain. For example, many people intuitively analyze circuits sequentially. They begin by looking at the power source and then follow the circuit with their eyes, anticipating that the electrical phenomena will occur in sequence until there is a disruption in the circuit or they return to the power source. This sequential reasoning underlies several misconceptions, including the misconception that current flows out of a battery and through light bulbs until it reaches a gap, and the misconception that each light bulb consumes a bit of the current that passes through it. Our curriculum includes numerous exercises using a variety of different circuit configurations that consistently demonstrate that students must reason globally about a circuit, because a sequential analysis does not correctly predict electrical phenomena.

The sequence of concepts and activities described by the Physics Education Group are nested within the conceptual change instructional approach. (The low-level tactics that we implemented were based on an analysis of human tutoring behavior with this same curriculum, as will be discussed later in this section.) At the core of this approach is an analysis of the fundamental concepts in a scientific domain and a series of activities designed to illustrate these concepts and contradict commonly held misconceptions. Students are taken through several cycles of making predictions based on their existing conceptions, completing activities, and then reflecting on the outcomes and implications for their conceptions of the field. There are repeated demonstrations in the literature of the effectiveness of this approach (Redish et al. 1997; Thacker et al. 1994).

The lessons in BEETLE II apply this instructional approach within the context of a simulated circuit workbench. For each topic, the students are asked to predict the behavior of a given circuit and explain that prediction. Next, the students test their predictions in the circuit simulator by building the circuits and observing their behavior. After the test is complete, the students are asked whether the simulated results matched their predictions. Finally, when predictions were not realized, students are asked to try and generate an explanation for the behavior of the circuit that they just observed. We refer to this general flow as the “Predict-Verify-Evaluate” strategy (PVE). Most of the curriculum is set up in this manner, which encourages students to think deeply about the phenomena that they observe and attempt to infer the underlying explanatory principles in the domain.

It may be informative at this point to compare our strategy to the strategies implemented in other ITSs that address misconceptions, such as Why2-ATLAS (Jordan et al. 2006), ITSPOKE (Litman and Silliman 2004) and AUTOTUTOR (Graesser et al. 1999). These systems have a library of remediations tailored to the assorted misconceptions. They ask students questions, and use natural language interpretation to attempt to diagnose misconceptions present in the answer and to select a matching remediation. With our tutoring strategy we do not use misconception-specific dialogue-based remediations. The simulation-based exercises are designed to elicit and counter common misconceptions. Students who make faulty predictions of circuit behavior based on misconceptions are engaged by the tutor in a dialogue designed to have them explicitly acknowledge the inability of their existing conceptual framework to accommodate the observed phenomena and then to "reverse-engineer" a more appropriate conceptual understanding of the basic principles of electricity and circuits.

We hypothesize that two things are working together to make our curriculum effective in overcoming common misconceptions. The first is our content—the presentation of a carefully designed sequence of concepts. The second is our primary tutoring strategy—the exercises associated with those concepts embedded within Predict-Verify-Evaluate cycle.

In our conceptualization of this strategy, the role of the tutor is to help students accomplish each step within the cycle, i.e. ensure that the student correctly performs the required simulations, that they remember their predictions and are aware of any differences between the predictions and the observations, and that they complete each reflection dialogue with knowledge of the correct explanation. This cannot be fully accomplished without a tutorial dialogue capability that helps students to generate for themselves accurate statements about the underlying principles of basic electricity and electronics.

The challenge for the dialogue component is then to select an appropriate generic tutoring tactic aimed at completing the current step, and then instantiate it, on the fly, within the context created by both the student's statements and the state of the objects in the simulation environment. Our choice of tactics was based on a human-human tutoring study in the context of our curriculum, described in the next section.

Tutoring Tactics

Before implementing our curriculum in the BEETLE II system, we piloted it with three experienced human tutors, each of whom worked through the exercises individually with multiple students, providing feedback during the prediction and reflection stages as necessary. All of the tutors were knowledgeable in the area of electricity and circuits. Two of the tutors had experience providing military technical training and the other had experience teaching college courses and providing private tutoring. A total of thirty students participated in this pilot study for approximately four hours each. The curriculum was presented to the student on a laptop and the student interacted with their tutor via a chat interface. All of the student and tutor questions, answers, and feedback were logged.

After all of the participant data were collected, our team analyzed the transcripts in order to identify effective tactics used by our human tutors. First, we developed a hierarchical coding scheme for the low-level techniques that the tutors used in response to student answers (both correct and incorrect) (Steinhauser et al. 2007). The scheme described micro-level decisions taken by the tutors while trying to help the student perform post-exercise reflection, for example, giving hints to help the student arrive at a better explanation of what they observed, and used eight different tactic codes at the lowest level.

Once all of the transcripts were coded according to this scheme,[2] we turned our focus to evaluating the effectiveness of each tactic and the effectiveness of the most commonly occurring sequences of remediation tactics (when a single tactic was not successful at eliciting the correct answer from a student). Each mini tutorial dialogue that followed each question in the curriculum (across all participants and all tutors) was classified according to its sequence of codes and according to the success of its final result. A mini tutorial dialogue was classified as successful if it resulted in the student generating a full and correct answer and as unsuccessful if the tutor was the one who eventually provided the full and correct answer to the student.

For example, one common sequence found in the transcripts was:

Student:	Partially correct, some missing information
Tutor:	Remediates with generic prompt for more information
Student:	Fully correct
Sequence:	Successful

These sequences were tallied and the most frequently used successful sequences were promulgated for implementation as tutoring tactics in the BEETLE II system. While the ultimate goal was to develop a system that could be used to test a wide variety of tutoring tactics, we felt that this selection would yield a strong initial version of the system. Our analysis also indicated that tutorial dialogues that were not successful after three tutor turns were rarely ever successful. Hence, we recommended that the BEETLE II system default to providing the full and correct answer to the student if three attempts at eliciting the answer were not successful.

Based on this analysis, we developed a scheme to structure the tactics generated by a computer tutor. The list of tactics, found in the Appendix, reflects a collection of the various tactics that our tutors used. The details of how tactics are chosen and instantiated at each step of the dialogue are discussed in the "Tutorial Planner and Natural Language Generation" section. The Appendix also illustrates how the PVE strategy and the dialogue remediation tactics work together in the BEETLE II ITS to address student misconceptions. It contains examples of students making incorrect predictions in the beginning and arriving at the correct answer during reflection dialogue, either by themselves after observing the discrepancy in simulation, or with the aid of the system.

Assessment

To allow us to test the effectiveness of our instruction, we developed pre- and post-tests by leveraging published conceptual inventories in the domain of electricity and electronics, particularly the "Determining and Interpreting Resistive Electric Circuits Concepts Test" (Engelhardt and Beichner 2004) and the "Electric Circuit Diagnostic Test" (Sokoloff 1996). After selecting a subset of questions that were relevant to our curriculum, we conducted pilot testing of these items with 106 students at the US

---

[2] Three independent raters achieved kappa scores ranging from 0.69 to 0.88 across the levels of the coding hierarchy, indicating agreement ranging from moderate to substantial.

Navy Training Command Great Lakes. In this pilot test, we followed each multiple choice question with the open-ended question, "Why?" The primary objective of this pilot effort was to determine the extent to which the distractors were valid representations of common misconceptions in the field. In other words, we evaluated the extent to which the students' written explanations for selecting a distractor were consistent with the misconception that the distractor was purported to represent. The results of this analysis provided positive evidence for the validity of the test items. In addition, these data allowed us to assemble two different tests that were well-matched in terms of both difficulty and content coverage. We used one test, with 22 questions, for our pre-test and the other, with 21 questions, for our post-test in the current study. Note that further evidence for the equality of these two tests is presented later, through the performance of the no-training control condition of our study.

System Implementation

Architecture

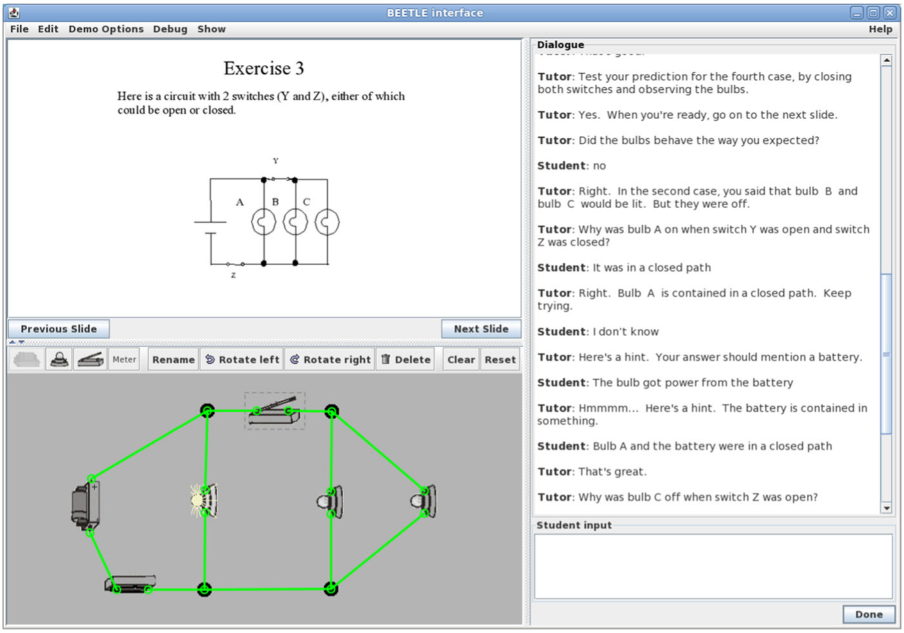
A screenshot of the system is shown in Fig. 1. The student interface includes an area to display reading material, a circuit simulator, and a dialogue history window. All interactions with the system are typed to avoid issues with speech recognition.[3] Students read pre-authored curriculum slides and carry out exercises which involve experimenting with the circuit simulator. They are asked to explain observed behavior by answering questions such as "Why was bulb C off when switch Z was open?". The system also asks high-level questions, such as "Under what circumstances will a switch affect a bulb?". The graphical user interface is integrated with the open-source GNUCAP circuit simulator,[4] which is used to correctly display bulb brightness and provide accurate voltage and current readings for arbitrary circuits.

An excerpt from the example dialogue in Fig. 1, formatted for readability, is shown in Fig. 2. Students are intended to learn that a bulb will light up when it is in a closed path with a battery. Thus, when the student says that the bulb is contained in a closed path, the system treats the answer as partially correct but incomplete. It restates the student answer to confirm the correct part, a tactic that is effective in human-human tutoring (e.g. Dzikovska et al. 2008), and then prompts the student to complete the answer. When the student is unable to continue correctly, the system gives two different hints until the student arrives at a correct answer. We will use the first exchange (turns 2–3) of this dialogue as a running example to illustrate the system's capabilities. Some additional examples demonstrating system behavior and strategies used, excerpted from our evaluation corpus, are provided in the Appendix.

---

[3] We are currently engaged in data collection to obtain speech recordings necessary to build language models and support spoken interaction (Bell et al. 2012).

[4] http://www.gnucap.org



Exercise 3

Here is a circuit with 2 switches (Y and Z), either of which could be open or closed.

Tutor: Test your prediction for the fourth case, by closing both switches and observing the bulbs.

Tutor: Yes. When you're ready, go on to the next slide.

Tutor: Did the bulbs behave the way you expected?

Student: no

Tutor: Right. In the second case, you said that bulb B and bulb C would be lit. But they were off.

Tutor: Why was bulb A on when switch Y was open and switch Z was closed?

Student: It was in a closed path

Tutor: Right. Bulb A is contained in a closed path. Keep trying.

Student: I don't know

Tutor: Here's a hint. Your answer should mention a battery.

Student: The bulb got power from the battery

Tutor: Hmmmm... Here's a hint. The battery is contained in something.

Student: Bulb A and the battery were in a closed path

Tutor: That's great.

Tutor: Why was bulb C off when switch Z was open?

Fig. 1 Screenshot of the BEETLE II system

The system architecture is shown in Fig. 3. The system behavior can be seen as an instance of the two-loop approach from VanLehn (2006). The outer loop consists of the system presenting a series of high-level activities to the user. Each activity is

1 Tutor: Why was bulb A on when switch Y was open and switch Z was closed?

2 Student: It was in a closed path

3 Tutor: Right. Bulb A is contained in a closed path. Keep trying.

4 Student: I don't know

5 Tutor: Here's a hint. Your answer should mention a battery.

6 Student: The bulb got power from the battery

7 Tutor: Hmmmm... Here's a hint. The battery is contained in something.

8 Student: Bulb A and the battery were in a closed path

9 Tutor: That's great. Why was bulb C off when switch Z was open?

Fig. 2 Example dialogue with the system, excerpted from Fig. 1

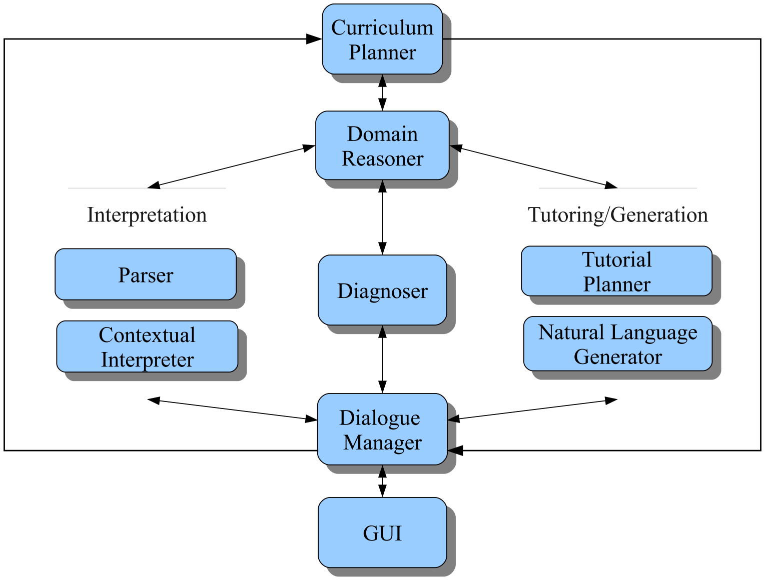


Fig. 3 System architecture diagram

made up of a series of questions, which roughly correspond to steps within Vanlehn's inner loop.[5] Both the activities and the questions are generated by a curriculum planner. For each question, the system provides a dialogue-based feedback service. The tutor processes student input (either natural language or a circuit submission) and gives feedback and hints until the correct answer is established, either by the student answering correctly, or by the tutor deciding that further remediation will be unproductive and therefore telling the student the correct answer and moving on to the next question.

To provide feedback on natural language answers, the feedback service uses a standard interpretation pipeline, with domain-independent parsing and generation components supported by domain-specific components for tutorial decision-making. Student answers are first analyzed by a deep parser. A contextual interpreter is then used to provide reference resolution and to transform the parser output into the domain-specific representation used by the domain reasoner. The diagnoser checks the student answer representation for factual correctness and matches it against the expected answer, returning a detailed diagnosis containing a list of correctly mentioned objects and relations, those that are missing, and those that contradict the expected answer. Next, the diagnosis is passed to the tutorial planner, which decides on an appropriate response, including which tutoring tactics to use. Finally, the natural language generator converts the tutorial planner's high-level decision into the text of a response to be presented to the student.

---

[5] The analogy is imperfect, since the focus of our activities is on teaching concepts rather than solving multi-step problems. Questions asked by the system often build on each other to help students build up their understanding of a complex concept, but are not as inter-dependent as problem solving steps.

Curriculum Planning and Exercises

The high-level lesson flow, including the presentation of reading material and exercise sequencing, is managed by the curriculum planner. For each exercise, the planner stores the reading material to be presented and the questions the system will ask the student, together with the associated circuits and reference answers (described later in the "Domain Reasoning and Diagnosis" section). These are used by the dialogue manager to load the appropriate circuits into the simulator and set up the dialogue context at the beginning of each exercise.

At present, exercises and questions are presented to all students in the same fixed sequence, based on a lesson plan specified by the curriculum designers. We are considering implementing a more flexible approach in the future, allowing the curriculum planner to choose both exercises and questions to be presented based on a student model and an adaptive planner (e.g. Cho 2000). The fixed lesson plan was selected as the simplest possible option, since the focus of our work has been on adaptive feedback generation in the feedback service. However, the dialogue manager we have implemented in the system is fully capable of handling a dynamic lesson plan, and an adaptive curriculum planner could be substituted in without any changes to the rest of the system's architecture.

The curriculum includes four different types of questions:

- **Identify:** Object and attribute identification questions, e.g. "In this circuit, which bulbs will be on and which bulbs will be off?" These questions ask the student to identify one or more objects or attributes. They require short answers which are much simpler to interpret than explanation questions, but still require the system to correctly resolve references to objects on the screen ("The bulb in 1"), ellipsis ("all off"), etc., requiring proper interpretation and integration with the domain reasoner.
- **Explain:** Explanation and definition questions, e.g. "What are the conditions for a bulb to light?" These questions are key to supporting self-explanation, and student answers to these are the most difficult to process.
- **Multiple choice:** Multiple choice questions, requiring students to answer "yes" or "no", or to choose from a set of alternatives. These questions can be processed straightforwardly using simple template matching.
- **Number:** Number questions, either a measurement report ("What voltage reading do you get?") or a count ("How many closed paths can you find in diagram 1?"). These questions require some post-processing, but again, can be dealt with straightforwardly using a simple grammar.

In the remainder of this section, we discuss the interpretation and dynamic feedback generation pipeline, focusing on the processing necessary to produce automatically generated feedback for "identify" and "explain" questions.

Dialogue Management

The interpretation and feedback generation pipeline is coordinated by the dialogue manager. Our approach is based on the information-state update approach to dialogue management (Larsson and Traum 2000), in which the dialogue manager keeps the central representation of the state of the dialogue, and individual components consult it to support context-specific interpretation and generation, and update the state with the results of their processing. The dialogue manager calls the components in the feedback service pipeline (interpretation, domain reasoning and diagnosis, then tutorial planning and generation) to process the student answer and generate appropriate feedback, and decides when to pass control to the curriculum planner to move on to the next question.

Dialogue state is represented by a cumulative answer analysis which tracks, over multiple turns, the correct, incorrect, and missing parts of the answer. Every time the student attempts an answer, the objects and relations mentioned are added to the dialogue state. Content from the hints produced by the tutor is also added. This allows the system to track the answer over multiple turns. For example, if the system asks “Which components are in a closed path”, expecting the answer “the bulb and the battery”, the student may say “the bulb.” When prompted to say more, they may complete their answer by saying “the battery.” The dialogue manager will combine the information from the two answers and conclude that the complete answer has been provided, allowing the student to move on to the next question.

Next we describe the implementation of the pipeline components.

Interpretation Components

To parse student utterances, the system uses a two-stage process. First, the TRIPS dialogue parser (Allen et al. 2007) produces a domain-independent semantic representation including high-level word senses and semantic role labels. Next, the contextual interpreter uses a reference resolution approach similar to Byron (2002) and a set of hand-written rules specific to our domain to transform the domain-independent parser output into the domain-specific representation employed by the domain reasoner, using the ontology mapping mechanism described in Dzikovska et al. (2008).

Example output from the TRIPS parser and the contextual interpreter is shown in Fig. 4. The details of the parser representation are beyond the scope of this paper. Note, however, that it identifies a referential pronoun “it”, and provides some additional syntactic information (not shown here for brevity, see Allen et al. (2007) for details) that serves as the input to the reference resolution algorithm. It also identifies that the word “in” is used in its spatial, and not temporal, sense.

The contextual interpreter takes the parser output and resolves the pronoun to a specific bulb in the environment, `LightBulb-4-1`, using information from the domain reasoner about the objects visible on screen, and the dialogue history which records the most salient object based on the last question asked.[6] The meaning of

---

[6] The interpreter also performs ellipsis resolution. For example, if the student replies “all off” in answer to the question “If switch Y is closed and switch Z is open, which bulbs will be on, and which bulbs will be off?”, the interpreter can determine that it can be taken to mean “all of the bulbs will be off”.

Student: It was in a closed path

(a) Parser output:

```
(SPEECHACT V1 SA_TELL :CONTENT V2)
    (F V2 (LF::HAVE-PROPERTY BE) :PROPERTY V3 :THEME V4
          :TMA ((TENSE PAST)))
    (PRO V4 (LF::REFERENTIAL-SEM IT))
    (F V3 (LF::SPATIAL-LOC IN) :OF V4 :VAL V5)
       (A V5 (LF::ROUTE PATH) :MODS (V6))
          (F V6 (LF::OPENNESS-VAL CLOSED) :OF V5))
```

(b) Interpreter output:

```
(ID1 LightBulb LightBulb-4-1)
(ID2 Path _P1) (ID3 is-closed _P1 t)
(ID4 contains _P1 LightBulb-4-1)
```

Fig. 4 **a** The domain-independent semantic representation produced by the TRIPS parser, and (**b**) the corresponding domain semantic representation produced by the BEETLE II interpreter for turn 2 in our example dialogue (Fig. 2). Some details are omitted for readability

the word "in" is further refined from "spatial location" specified by the parser. The interpreter determines that it corresponds to the knowledge base concept `contains`, used to represent the relationship between a component and a path (and not, for example, `state-of`, as it would in the sentence "the two terminals are in the same state").

This two-stage interpretation approach has been adapted from previous work on multi-domain task-oriented dialogue systems (Dzikovska et al. 2008) as a means to improve system robustness to unexpected answer phrasings. One of the major sources of interpretation failures in systems that use symbolic interpretation methods is missing lexical entries and grammar rules from hand-coded grammars and lexicons for the domain. The use of a domain-independent parser and semantic representation allows the system to recognize syntactic variations which represent the same meaning, e.g. "The bulb is contained in a closed path" and "There is a closed path containing a bulb." The domain-independent ontology used by the TRIPS parser allows the system to recognize synonyms. For example, if the student says "Terminals 1 and 2 are linked", the system will identify "linked" as an instance of the `LF::Attach` domain-independent sense, which is a synonym for "connected". Both words will be automatically mapped to the `connected-to` concept understood by the domain reasoner by writing a single mapping rule that matches the `LF::Attach` and `connected-to` concepts in the domain-independent and domain-specific ontologies.

The TRIPS grammar and lexicon incorporate information from existing wide-coverage resources (Dzikovska et al. 2004; Swift 2005) and have been used in different dialogue domains over many years, further extending their coverage. Therefore, lexical entries and grammar rules are less likely to be missing compared to domain-specific grammars used in symbolic semantic interpreters for earlier dialogue

ITSs, though occasional gaps in coverage are unavoidable. In addition, interpretation may fail where concepts that have similar meanings in a specific limited domain are not synonymous in a broader context. For example, if a student says “a broken path” instead of “an open path”, the equivalence may not be recognized because “broken” and “open” are not synonyms in a domain-independent ontology. This issue is addressed in part by writing additional rules to cover domain-specific semantic information not encoded in the domain-independent ontology, and in part by using an error recovery policy that points out the specific terms in an utterance that the interpreter fails to recognize, discussed later in this section.

Domain Reasoning and Diagnosis

The domain reasoner uses the Knowledge Machine system (Clark and Porter 1999; Dzikovska et al. 2006) to answer factual questions about the state of the world (e.g. whether a given switch is open, or which bulbs are lit in a given circuit). It uses a description logic formalism and incorporates a knowledge base with an ontology of the objects that can be seen on the screen, their properties, and relations between them (currently 14 object types and around 50 properties and relations). The reasoner supports contextually-appropriate interpretation and generation, and, most importantly, diagnosis of answer correctness.

While student answers to multiple choice and number questions are either right or wrong, for “identify” and “explain” questions the situation is more complicated. The diagnoser compares the objects and relations in the student answer against the reference answer to determine the correct, contradictory, missing and irrelevant answer parts (Dzikovska et al. 2008). At present, the system uses a heuristic matching algorithm to classify answer parts into the appropriate category. In future we will consider using a statistical classifier similar to Nielsen et al. (2008). For “identify” questions, reference answers are automatically computed based on the information provided by the domain reasoner. For “explain” questions, reference answers are provided by instructional designers to ensure that they match the curriculum goals.

Each student explanation is checked on two levels, verifying both (a) *factual* and (b) *explanation* correctness. Going back to our example, we must check (a) that the bulb identified as `LightBulb-4-1` is indeed in a closed path, and (b) that being in a closed path is an acceptable explanation for the bulb being lit. Different remediation tactics are needed depending on whether the student made a factual error (i.e. they misread the diagram and the bulb is not in a closed path) or produced an incorrect or incomplete explanation (i.e. the bulb is indeed in a closed path, but they failed to mention that a battery needs to be in the same closed path for the bulb to light). The diagnoser verifies factual correctness by querying the domain reasoner and explanation correctness by matching the student answer against the reference answer. An example of the output from the diagnoser is given in Fig. 5. In this case, everything the student said was correct with respect to the reference answer, and there were two missing parts (corresponding to “the battery must be in the same closed path”).

Student: It was in a closed path

```
(:MATCHED
    (ID1 LightBulb LightBulb-4-1)
    (ID2 Path _P1) (ID3 is-closed _P1 t)
    (ID4 contains _P1 LightBulb-4-1))
(:CONTRADICTORY ())
(:EXTRA ())
(:MISSING
    (?ID10 Battery ?BT) (?ID11 contains ?P ?BT))
```

Fig. 5 The diagnosis produced for turn 2 in our example dialogue

Tutorial Planner and Natural Language Generation

The system generates tutoring feedback automatically based on the diagnosis of the student answer. In order to generate a response, the tutorial planner first makes a high-level decision on which tactic or set of tactics to use, and then passes them to a natural language generator to realize as text.

The tutorial planner supports a set of generic tutoring tactics and implements a top-level tutoring policy for choosing appropriate tactics at each point of the interaction. An example textual summary of a decision rule is shown in Fig. 6. It says that early in the dialogue, if the student is on the right track but some parts of the answer are missing, the system should give positive feedback, acknowledge the correct portion of the student's statement, and go back for more information, using a generic "keep going" prompt to try to elicit the missing parts of the student's answer.

Figure 7a shows the tutorial planner output for turn 3 in our example, produced based on the diagnosis from Fig. 5 and the rule in Fig. 6. The tactic decisions are expanded into planning operators using relevant portions of the diagnosis structure, specifying the content to be acknowledged. This output is passed on to the natural language generator. The system uses a domain-specific sentence planner to determine how to allocate content to sentences and to choose syntactic structures and lexical items. The sentence planner output is shown in Fig. 7b. It shows, for example, that the system decided to use the `ACKNOWLEDGE-CORRECT` template for confirming

```
IF (:CONTRADICTORY is empty)
   AND (:MATCHED is not empty)
   AND (:MISSING is not empty)
   AND attempt-number=1
THEN try Feedback-Positive, Acknowledge-good-bits, Remediate-GoBackForMore-Keep-Going
```

Fig. 6 Pseudocode for the rule used by the tutorial planner to decide on a sequence of tactics in turn 1 in our example dialogue. Tutorial planner output from application of this rule is shown in Fig. 7a

(a) Tutorial Planner output

```
((ID101 (JOIN ID102 ID103 ID104))
 (ID102 (CONTENT-FREE-PROMPT CORRECT))
 (ID103 (REINFORCE-CONTENT
        (ID1 LightBulb LightBulb-4-1)
        (ID2 Path _P1) (ID3 is-closed _P1 t)
        (ID4 contains _P1 LightBulb-4-1)))
 (ID104 (CONTENT-FREE-PROMPT KEEP_TRYING)))
```

(b) Sentence Planner output

```
((ID201 (JOIN ID202 ID203 ID204))
 (ID202 (ACCEPT ACKNOWLEDGE-CORRECT))
 (ID203 (NSP-CONTENT
            ((ADDPROP LightBulb-4-1 BULB-ELECTRICAL "bulb A")
             (ADDPROP V1 PATH)
             (DETAIL V1 CLOSED-STATE)
             (PROP-PROPERTY CONTAINING V1 LightBulb-4-1)
             (FOCUS LightBulb-4-1))))
 (ID204 (ASSERT CORRECT NOT_COMPLETE2)))
```

(c) Realized text

Tutor: Right. Bulb A is contained in a closed path. Keep trying.

Fig. 7 Tutorial planner output resulting from the application of the tutoring rule in Fig. 6, along with the sentence planner output and the realized text

correctness (e.g. saying “Right”). When reinforcing the correct portion of the student answer, the sentence planner decided that it was most appropriate to use the phrase “bulb A”, as opposed to “the bulb” or “it”, and that the focus of the sentence should be the bulb, rather than the path (in the latter case, the same content would be realized as “There is a closed path containing bulb A”.) This representation is finally passed to an existing wide-coverage surface realizer, FUF/SURGE (Elhadad and Robin 1992), which outputs the corresponding text.

The tutorial planner typically has multiple possible tactics available at each point in the interaction. In our example, instead of making the decision to reinforce correct student content and use a contentless prompt, the tutorial planner could have used the content from the `missing` field in the diagnosis to generate hints. Some example hints generated by the system are shown in turns 5 and 7 in Fig. 2. For a low specificity hint the system typically selects an as-yet unmentioned object and hints at it, in our case, saying that the answer should mention the battery. For high-specificity hints, it attempts to hint at a two-place relation, in our example saying that the battery must be contained in something, with the hope that the student can fill in the blank by remembering the concept of a closed path.

The choice of which of the applicable prompt, hint or restatement tactics to use is governed by a rule-based tutoring policy. In our current policy, if remediation is necessary, the system starts by giving a contentless prompt (optionally restating the correct part of the answer, if any), then proceeds to more specific hints and suggestions for reading slides as appropriate for the context. Finally, if the student is unable to arrive at the correct answer after three to four attempts, the system gives away the answer using the bottom-out tactic. The specific decisions about which prompt or hint to use at each step take into consideration the error type (by checking matched, contradictory and missing parts of the diagnosis, as shown in Fig. 6), the number of incorrect answers received in response to the current question, and the number of previous answers that the system failed to interpret.[7]

In addition to remediation tactics after flawed answers, the system implements different tactics for responding to correct answers, to predictions and to interpretation difficulties. If the student gives an acceptable answer, the tutorial planner can choose to either accept it and move on, or to accept it, but restate the answer using better terminology, based on the policy discussed in Dzikovska et al. (2008). For predictions, the system initially accepts the prediction neutrally (without acknowledging correctness) and then either confirms that the student was correct, or re-iterates the difference between the prediction and the actual outcome in the evaluation stage (see dialogue with Participant 9 in the Appendix). The list of tactics and example dialogues annotated with tactic names are given in the Appendix.

Finally, the tutorial planner implements an error recovery policy (Dzikovska et al. 2009) to deal with non-interpretable utterances. As discussed in the “Interpretation Components” section, the domain independent parser and ontology support alternative syntactic phrasings and synonyms, but interpretation may fail, for example, when the student uses terms that are related in the domain but not labeled synonymous in the domain-independent ontology. To help mitigate interpretation failures, the tutorial planner attempts to produce a message that describes the problem but without giving away the answer. For example, if the student refers to a broken or damaged path, the system will say, “I’m sorry, I’m having a problem understanding. I don’t understand when you say that paths are damaged. Batteries or bulbs can be damaged, but not paths”.[8] The help message is accompanied by a hint at the appropriate level, again depending on the number of previous incorrect and non-interpretable answers. In task-oriented spoken dialogue systems, this method has been used successfully to help users learn to phrase their requests in a way that is understood by the system (Hockey et al. 2003). We evaluate its effectiveness in the context of tutorial dialogue in the next section.

---

[7] The system was designed to support experimentation with different policies. The decisions about which tactics to use and which factors to take into account in BEETLE II were based on the data analysis described in the “Curriculum Development” section. The same tutorial planner component has been used in a different system, where other factors such as student confidence were taken into account (Callaway et al. 2007).

[8] The content of such a message is computed automatically from the ontology.

Evaluation

Setup

A key decision in any system evaluation is the nature of the control condition(s). In this study, we included two control conditions. The first was a no-training control condition, where participants completed the pre-test and the post-test, separated by an hour of distractor tasks (a lesson on differential equations). The purpose of including this condition was to generate the data needed to calculate effect sizes for the training systems of interest using Cohen's d.

Next, given that the focus of this research is on developing the NLP capability necessary to support tutorial dialogue during the reflection phase of the PVE cycle, we determined that our second control condition should be a system that applied the same curriculum and only differed from our dynamic, adaptive, NLP-enabled ITS in the nature of the follow-up dialogue. The BEETLE II tutor we have described in this paper aims to elicit the correct answer from the student over multiple turns through adaptive natural language feedback guiding them towards the correct answer. We will refer to this as the "elicit answer" system (ELICIT). In addition, we built another version of the system which immediately tells the student the correct answer without analyzing the answer or providing any explicit corrective feedback (the "tell answer" system, TELL).[9] The user interface, lesson materials, activities and even the tutor questions were exactly the same in the two versions. The only thing that differed was the type of feedback the students received after answering a question (including all identify, explain, multiple choice, and number questions discussed in the previous section).

The ELICIT version of BEETLE II processed student input and produced a diagnosis and dynamically generated feedback and hints as described in the previous section. In the TELL version of BEETLE II, the system did not attempt to provide any explicit feedback on the accuracy of student answers. Instead, each time a student answered a question the tutor would simply give them a neutral acknowledgment, followed by a statement of the correct answer (bottom-out). An example interaction with the TELL system is given in Fig. 8. The system moves on to the next question without attempting remediation. It is left entirely up to the student to notice whether their answer matches the answer given by the system, and to determine whether any differences in phrasings and terminology are meaningful.

Procedure

After reviewing the informed consent paperwork, all participants filled out a demographic questionnaire and took the pre-test. The participants were then introduced to BEETLE II and given a brief demonstration of its functionality. The students spent the majority of the experimental session working through two lessons with either the TELL or the ELICIT system. Students typically spent 2–3 hours working through the

---

[9] This distinction follows the terminology from Chi et al. (2009).

TELL1 Tutor: Why was bulb A on when switch Y was open and switch Z was closed?

TELL2 Student: It was in a closed path

TELL3 Tutor: OK. One good way to answer the question is: Bulb A is still contained in a closed path with the battery. Why was bulb C off when switch Z was open?

Fig. 8 Example dialogue with the TELL system, corresponding to turns 1–3 with the ELICIT system in Fig. 2

lessons. The difference in the type of feedback provided by the ELICIT versus TELL system was the only difference between these two conditions.

After the students completed the lessons, they took a post-test which included 21 multiple choice questions and filled out a usability and satisfaction questionnaire. The REVU-IT (Report on the Enjoyment, Value, and Usability of an Intelligent Tutoring System) (Dzikovska et al. 2011) Questionnaire was used to assess usability and satisfaction with BEETLE II. REVU-IT asked participants to rank their agreement, on a 5-point Likert scale, to both positive and negative statements regarding the lesson materials (10 questions), the circuit-building workspace (13 questions), the computer tutor (35 questions), and their overall reaction to BEETLE II (5 questions). They were then debriefed, thanked and paid. The full study took 3–4 hours per participant.

In the no-training condition, participants first reviewed an informed consent document and then completed the same demographic survey and pre-test as the participants in the other two conditions. Next, they spent an hour completing a distractor task, which involved working through a lesson on differential equations within the context of modeling predator and prey relationships. This was followed by the post-test. They were then debriefed, thanked and paid. The no-training group took 2 hours to complete the study.

Corpus

Guided by VanLehn et al. (2007), we conducted a power analysis and determined that a total of 120 participants (40 per condition) would give us reasonable statistical power (0.85) to reject the null hypothesis ($alpha = 0.05$), assuming an effect size of 0.75 between our conditions. Participants were recruited from a Southeastern University in the US using the University's online recruiting tool. The recruiting system advertised for students that had no prior knowledge of electronics and electricity and participants had to confirm the lack of prior knowledge before they were enrolled in the study. A total of 122 participants took part in the study. The no-training group consisted of 41 participants, who ranged in age from 18 to 38 years old ($M = 21.4$). Participants in the training conditions were randomly assigned to the ELICIT or TELL condition. There were 41 participants in the ELICIT condition and 40 participants in the TELL condition. Data from eight participants in the training conditions was dropped: four participants due to technical errors with the system, three who were outliers with respect to learning gain, and one outlier with respect to interpretation problems.[10] Outliers were defined as being three standard deviations from the mean for those variables.

Our analysis data set therefore includes 35 participants in the ELICIT condition, whose ages ranged from 18 to 37 years ($M = 21$); and 38 participants in the TELL condition, aged from 18 to 42 years ($M = 21.5$). In both training conditions, students were required to work through 26 exercises with a total of 215 questions asked by the system, 150 of which required natural language answers (the remaining questions required building circuits only). In the ELICIT condition, an average of 422 student turns were taken ($SD = 32$), with an average of 230 natural language turns ($SD = 24$). Transcripts contained on average 1417 student words ($SD = 217$). In the TELL condition, there was an average of 335 student turns ($SD = 6.25$), with an average of 156 language turns ($SD = 2.04$). Transcripts contained on average 900 student words ($SD = 218$).

The differences in the number of turns and in the number of words were significant between conditions (average number of student language turns: $t(71) = 18.78$, $p < 0.0001$; average number of student words: $t(71) = 10.14$, $p < 0.0001$). This is as expected, given that the students in the ELICIT condition were asked to revise their answers if they were incorrect, while the students in the TELL condition were allowed to move on regardless of answer correctness.

Results

We begin by presenting descriptive and inferential statistics on our two primary outcome measures, student learning and student satisfaction. Student learning is measured both in terms of individual gain scores and effect size by condition. While other studies have found that student satisfaction is not always directly related to learning outcomes (Jackson et al. 2009; Papadopoulos et al. 2009), we include it as an important outcome measure because it contributes to a student's willingness to continue to use a training system.

Next, we focus on one of our primary system characteristics—natural language interpretation of student answers. Accurate language understanding is a prerequisite for providing accurate feedback, and was therefore the main focus of our initial evaluation. Evaluating other aspects of the interaction, in particular the quality and appropriateness of system feedback, is the next step in our planned research program and is discussed in more detail in the "Future Work" section.

In our evaluation of the BEETLE II interpretation component, we first describe how human coders judged the correctness of student answers. Then we provide a number of measures of system accuracy in interpreting student answers, when compared to this human assessment. Finally we look more deeply into the impact of interpretation quality on our two primary outcome measures (student learning and satisfaction). These final analyses point to different strategies for improving

---

[10] After examining the transcript of the single participant who was an outlier in terms of frequency of interpretation problems, it seemed likely that they either refused to engage altogether or had some problem with English that prevented them from interacting with the system in a sensible manner.

the system's performance and the effects they may have on the desired learning outcomes.

Overall System Effectiveness

As expected, there was no learning gain in the no-training group. The participants in this condition had a mean pre-test score of 0.39 ($SD = 0.13$) and mean post-test score of 0.38 ($SD = 0.14$), with an average normalized learning gain of -0.04 ($SD = 0.22$). Learning gain was calculated using the formula: $\frac{posttest-pretest}{1-pretest}$. The maximum score on both the pre and post-tests was 1.0 (i.e. 100 %); thus the maximum learning gain using this calculation is 1.0.

In contrast, both the ELICIT and TELL versions of BEETLE II were highly effective at teaching basic electricity and electronics concepts. Participants learned significantly between pre- and post-test. The ELICIT condition had a mean pre-test score of 0.35 ($SD = 0.13$) and mean post-test score of 0.75 ($SD = 0.12$). The average normalized learning gain was 0.61 in ELICIT ($SD = 0.15$), with a significant increase between pre- and post-test, (paired t-test, $t(34) = -19.15$, $p < 0.00001$). The TELL condition had a mean pre-test score of 0.34 ($SD = 0.13$) and mean post-test score of 0.77 ($SD = 0.14$). The average normalized learning gain was 0.65 in TELL ($SD = 0.21$), again a significant increase between pre- and post-test (paired t-test, $t(37) = -17.38$, $p < 0.00001$). However, there was no significant difference in normalized learning gain between conditions, $t(71) = -0.863$, $p = 0.39$.

To determine the effect of the ELICIT and TELL conditions, effect sizes, using Cohen's d, were calculated as $\frac{M_{exp}-M_{notrain}}{\sigma_{pooled}}$, where M_{exp} is the mean of the corresponding experimental condition (ELICIT or TELL), $M_{notrain}$ is the mean of the no-training condition, and σ_{pooled} is the pooled standard deviation for the no-training condition and the relevant experimental condition. Both experimental conditions had a large effect on student learning. When compared to the no-training condition, the ELICIT system had an effect size of $d = 1.72$, and the TELL system had an effect size of $d = 1.69$.

With respect to student satisfaction, we focus on the two sections of the REVU-IT questionnaire that are related to tutoring quality.[11] The Tutor score represents the subset of questions assessing satisfaction with the natural language tutor; for example, "I felt that the tutor understood me well" or "I found our dialogues to be boring." The Overall score covers the five questions addressing overall satisfaction with the system, e.g. "I would use this system again in the future to continue to learn about electricity." The full questionnaire, and the divisions into specific subsets, are presented in Dzikovska et al. (2011). The mean Overall satisfaction scores were 3.4 out of 5 ($SD = 0.87$) for ELICIT and 3.67 out of 5 ($SD = 0.85$) for TELL. There was no significant difference between conditions ($t(71) = -1.31$, $p = 0.20$). In contrast, the Tutor satisfaction score was significantly higher in TELL than in ELICIT

---

[11] The 4 sections are: reading material, simulation interface, tutoring and overall satisfaction. Assessment of reading material and simulation interface are outside the scope of this paper.

(TELL: $M = 3.33$, $SD = 0.65$, ELICIT: $M = 2.57$, $SD = 0.61$; $t(71) = -5.13$, $p < 0.0001$).

Since the only difference between the ELICIT and TELL systems is the natural language feedback used in ELICIT to elicit the correct response from the student, we surmised that the quality of the natural language interpretation may affect the learning outcomes. This led us to carry out a detailed evaluation of the performance of the NLP module in the ELICIT system.

Evaluating NLP Performance

Creating The Gold Standard

Evaluating NLP components for tutorial dialogue systems is complicated, because the representations used by current systems are domain specific, and therefore there are no shared data sets or standard methodologies for evaluating system performance. Ideally, we would like to have a "gold standard" semantic representation for each student answer, and then compare how well the representations generated by the system match against such standard representations. However, this requires human annotation with good inter-rater reliability, which is labor intensive and impractical on a large scale. Therefore, to evaluate the interpretation component, we use a previously established methodology based on comparing the correctness judgments made by the BEETLE II diagnoser with accuracy codes assigned to the same utterances by human raters (Dzikovska et al. 2012). This approach simplifies creation of the gold standard.[12]

The outputs from the BEETLE II diagnoser are mapped into a 5-class annotation scheme as shown in Table 1. This scheme is based on the DEMAND coding scheme for assessing correctness of student answers in human-human dialogue (Campbell et al. 2009). All student utterances in the corpus were manually labeled with DEMAND labels ($\kappa = 0.69$, see Steinhauser et al. 2010) and automatically converted into our 5-class annotation scheme.[13] The resulting corpus consists of all student utterances in the data, each associated with two labels: a gold standard label based on the manual annotation, and an automatic label based on the output of the diagnoser. Examples of gold standard labels assigned to student utterances in the corpus are given in Table 2. The system is evaluated by comparing the labels and computing standard classification metrics.

Evaluation Data Set

The evaluation data set consists of all student utterances typed in response to tutor questions, together with labels automatically assigned by the diagnoser, and manually

---

[12] The disadvantage of this approach is that it does not distinguish between the errors made by the interpreter and the diagnoser. However, increased reliability in human gold standard labels is a major advantage.

[13] The motivation and details for the conversion are given in Dzikovska et al. (2012) and Dzikovska et al. (2012).

Table 1 Label set used for evaluating interpretation accuracy

Label	Description
non_domain	Metacognitive, social and nonsense expressions.
correct	The student answer is fully correct.
pc_incomplete	The student said something correct, but incomplete, i.e. some parts of the expected answer are missing
contradictory	The student's answer contains something that contradicts the expected answer, rather than just an omission.
irrelevant	The student's statement is correct in general, but it does not answer the question.

annotated gold standard labels as discussed earlier. We report evaluation scores for the ELICIT system only. The TELL system did not attempt any remediation. The entire data set covers the 150 questions requiring natural language answers asked by the system, and 8004 student answers. In our evaluation, we built three main test sets, based on the question types discussed in the "Curriculum and Exercise Management" section.[14]

- `All questions`: a test set containing every student response submitted to the system, together with the manual label and the automatically assigned class.[15]
- `Explain`: student responses to all explanation and definition questions, except those which asked for a prediction.
- `Identify`: all student responses to object or attribute identification questions, except those which ask for a prediction.

Table 3 shows the characteristics of our evaluation sets in terms of the number of student answers (total number of items in the set), and the number of unique questions from which the evaluation set items were drawn.

Evaluation Metrics

We focus on several different aspects of the system's performance. First, we need to have a metric that reflects the student's overall experience: if a student is interacting with the system, how accurately is the system interpreting their answers overall, and, correspondingly, how appropriate is the feedback that they are getting? Second, since

---

[14] We did not separately evaluate the answers to multiple choice and number questions, because they are processed through a separate template-based mechanism which is sufficiently simple to implement with basic NLP tools.

[15] Student answers to prediction questions are always accepted neutrally (without acknowledging correctness) – no remediation is given until the evaluation stage. Because the system does not attempt remediation, it skips some interpretation steps (such as ellipsis resolution with respect to question text), which can sometimes lead to an answer being incorrectly labeled as pc_incomplete instead of correct. These answers are included in our `All questions` data set, and thus scores computed for it reflect a conservative estimate of system performance. However, we decided to exclude answers to prediction questions from our `Explain` and `Identify` subsets, in order to focus on the system's performance on questions for which it attempts to give adaptive feedback.

Table 2 Examples of gold standard annotations for student answers in our evaluation corpus

Question	What are the conditions that are required to make a bulb light up?	
Reference Answer	There is a closed path containing both the bulb and a battery	
Annotated Answers	• The light bulb must be in a closed path with a connected battery.	correct
	• closed path, connected to a battery	pc_incomplete
	• battery is in a holder	irrelevant
	• I dont know	non_domain
Question	Explain why you got a voltage reading of 1.5 for terminal 1 and the positive terminal.	
Reference Answer	Terminal 1 and the positive terminal are separated by the gap	
Annotated Answers	• because there was not direct connection between the positive terminal and bulb terminal 1	correct
	• Because terminal 1 is connected to the positive battery terminal	contradictory
	• tell me the answer	non_domain

we are particularly interested in supporting open-ended explanation questions, we want to evaluate the system's performance on these questions separately. Finally, we want to establish baseline performance for comparing more advanced versions of the system against this initial version, and for comparing different NLP approaches to the natural language interpretation task.

From the "student experience" point of view, the most relevant metric is overall interpretation accuracy (referred to as simply "accuracy" in Tables 5 and 8). It is defined as the relative frequency of instances in which the label assigned by the system matches the gold standard label: $accuracy = \frac{\#matching}{\#total} = 1 - \frac{\#errors}{\#total}$. Thus the accuracy value on the `All questions` set reflects the total number of times the system makes a correct decision about which tutoring tactic to use.

However, accuracy scores do not take into account potential skewness of answer class distribution. Table 4 shows that some classes are much more common than others in the corpus. For example, 60 % of all student answers are rated as correct, but only 14 % are labeled pc_incomplete and 2 % overall are rated as irrelevant. In such unbalanced class distributions the overall accuracy metric often favors systems that focus on the most frequent classes at the expense of ignoring less frequent ones. In our case, it is possible to achieve 60 % interpretation accuracy simply by classifying every student utterance as correct and never attempting any remediation. From the point of view of establishing the baseline and comparing different system versions, we need to ensure that the system is doing well across all classes, and not just the

Table 3 Characteristics of our evaluation sets

Test set	Student answers	Unique questions
`All questions`	8004	150
`Explain`	3510	48
`Identify`	1319	26

Table 4 Gold standard label distribution in our evaluation sets. Proportion of total label count in parentheses

	`All questions`	`Explain`	`Identify`
correct	4814 (0.60)	1549 (0.44)	880 (0.67)
pc_incomplete	1097 (0.14)	804 (0.23)	149 (0.11)
contradictory	1655 (0.21)	788 (0.22)	263 (0.20)
irrelevant	140 (0.02)	118 (0.03)	15 (0.01)
non_domain	298 (0.04)	251 (0.07)	12 (0.01)

most common ones. Therefore, following Dzikovska et al. (2012), we report per-class F_1 scores for each class, to show how well each class is recognized, and the macro-average F_1 score, to characterize overall system performance.

For each class, the F_1 score can be defined as[16]

$$F_1(c) = \frac{2 * truepositive(c)}{2 * truepositive(c) + falsenegative(c) + falsepositive(c)}$$

The macro-average F_1 score is defined as an unweighted average of individual class scores,

$$F_1(macro) = \frac{1}{N_c} \sum_c F_1(c)$$

where N_c is the number of classes (5 in our case).

We report three other evaluation metrics related to student experience. First, we report a score based on a binary "accept/reject" decision as proposed in Dzikovska et al. (2012). Mismatches between system and human judgments are labeled as errors when computing the overall accuracy score but do not necessarily mean that the system will give inappropriate feedback. For example, in the dialogue with participant 24 in the Appendix, student turn `msg255` is misdiagnosed by the system as irrelevant, when human annotators assigned a pc_incomplete label. Ideally, the system should acknowledge that the student's answer is largely correct. Instead, the system gives hedged feedback ("Hmmm...") followed by a hint. Although not optimal, this feedback is still reasonable and does help the student to improve their answer. Therefore we computed an alternative accuracy measure, "accept-reject accuracy".[17] Using this metric, an answer counts as accepted if the diagnoser labeled it as correct. An answer counts as rejected if the diagnoser labeled it as flawed[18] or the interpreter rejected it as non-interpretable. In other words, an accepted answer is one where the system tells the student that it is correct and moves on, and a rejected answer is one where the system tells the student that there is a problem, and either asks the student to improve it or provides a correct alternative itself. This metric effectively assesses how well the system is able to decide when to accept the student answer as "good enough" and when to attempt a remediation. It does not consider whether the assigned label

---

[16] In information-theoretic terms, $2\frac{precision*recall}{precision+recall}$.

[17] This metric was introduced in Dzikovska et al. (2012), called "corrective feedback decision" there.

[18] We will use "flawed" to denote all answers identified as not entirely correct, i.e. pc_incomplete, contradictory and irrelevant.

will allow the tutorial planner to choose an adequate remediation tactic.[19] Comparing this metric against the overall accuracy shows how the interpreter's performance is affected by confusions between different types of answer flaws (as opposed to differentiating between correct and flawed answers).

Finally, we break down the errors that contribute to overall accuracy to look separately at the relative frequency of non-interpretable and misinterpreted utterances. It is well known from spoken dialogue system research that there is a trade-off between the frequency of misunderstandings (i.e. situations where the system incorrectly understands the user and takes the wrong actions) and non-understandings (i.e. situations where the system asks the user to repeat or rephrase the utterance rather than risk taking an incorrect action). Most dialogue systems make a decision to reject certain utterances as non-interpretable when it is not possible to find an interpretation that is reliable enough to use in response generation (Bohus and Rudnicky 2005). Such rejections can be frustrating to users. However, giving feedback based on a misinterpreted utterance can mislead and confuse the students as well. We observed many dialogue breakdowns caused by misinterpretation in early system pilots, and therefore developed a policy for dealing with non-understandings, where the system acknowledges its lack of understanding and gives a message describing the source of the problem (discussed in more detail in the next subsection). Separating out the non-interpretable from the misinterpreted utterances provides additional information about the misunderstanding vs. non-understanding trade-off within the BEETLE II system.

Evaluation Results

We report the values of our evaluation metrics for the complete set of student utterances, and for the `Explain` and `Identify` subsets separately. Overall accuracy, "accept-reject" accuracy, macro-average F_1, and relative frequencies of non-interpretable and misinterpreted utterances are reported in Table 5. Table 6 breaks down the macro-average F_1 into individual class metrics. The evaluation scores show that the system makes largely correct decisions about whether to accept a student answer or to request that it be rephrased, as reflected in its high "accept-reject" accuracy; this is also reflected in a high F_1 score for the correct class. However, the system often makes incorrect decisions about which class a flawed answer belongs to. This is reflected in lower scores for classes other than correct, and correspondingly lower macro-average F_1 scores. Unsurprisingly, explanation questions are the most difficult to interpret, and thus the evaluation scores for this subset, on all metrics, are much lower compared to the overall system average. We first provide an intrinsic evaluation of system performance, and return to the relationship between overall metrics and learning outcomes in the next two sections.

For intrinsic evaluation of classifier accuracy, a reasonable comparison is to the "majority class" baseline, in which the interpreter would label every student utterance

---

[19] For human annotations, an accepted answer is one labeled as correct, and a rejected answer is one labeled with any other category.

Table 5 Overall evaluation scores for the ELICIT system

Metric	`All questions`	`Explain`	`Identify`
macro-average F_1	0.49	0.46	0.74
overall accuracy	0.66	0.47	0.88
"accept-reject" accuracy	0.85	0.80	0.96
non-interpretable frequency	0.16	0.29	0.07
misinterpreted frequency	0.18	0.24	0.05

using the most frequent class according to the gold standard. As can be seen from Table 4, this would mean labeling every student utterance as correct. Table 7 shows the comparison between this baseline and the BEETLE II interpreter. The baseline has a macro-averaged F_1 score of 0.15 and an accuracy score of 0.60 on our `All questions` data set, and macro-averaged F_1 of 0.20 and accuracy of 0.44 on the `Explain` subset. The BEETLE II interpreter outperforms the majority class baseline in both cases, and by a particularly large margin on macro-averaged F_1. The differences between the interpreter evaluation scores and the majority class baseline scores are statistically significant for all three data sets, and for both macro-averaged F_1 and accuracy scores, with $p < 0.01$.[20]

Our discussion in this paper focuses on the overall interpretation metrics. We also conducted further system evaluation looking at confusion matrices and examining how different confusions correlate with learning gain. The confusion matrices and evaluation results are provided in Dzikovska et al. (2012).

Another question is how the system's accuracy compares to other state-of-the-art NLP methods. Direct comparison to other tutorial dialogue systems is difficult because of the lack of systematic reporting of standardized interpretation quality measures in the existing literature. However, the data from our corpus was used as part of a shared task on student response analysis at the 7th International Workshop on Semantic Evaluation (SemEval-2013), with the goal of investigating the performance of state-of-the-art NLP approaches in our setting. The participants were provided with a data set of question-answer pairs based on explanation questions extracted from the BEETLE II evaluation corpus, and challenged to build a system that directly classifies student answers into the same 5 classes that we used in our evaluation. The participating teams used approaches based on semantic similarity scores, recognizing textual entailment, domain adaptation, machine translation, combining different features, and supervised classification methods. The systems and results are discussed in detail in Dzikovska et al. (2013). For the 5-way classification task, the results ranged from 0.44 to 0.71 accuracy (0.31 to 0.62 macro-averaged F_1), and for the 2-way "accept-reject" task the accuracy ranged between 0.64 and 0.84.

These results are not directly comparable to those reported in Table 5, because the evaluation corpora in the SemEval task were split into training and test sets in order

---

[20] We used the approximate randomization significance test with 1000 iterations for testing for significant differences in F_1 scores, and the McNemar test for testing for significant differences in classification accuracy, as recommended by Dietterich (1998) and Yeh (2000).

Table 6 Per-class F_1 scores for the ELICIT system

Metric	`All questions`	`Explain`	`Identify`
F_1 correct	0.86	0.73	0.97
F_1 pc_incomplete	0.53	0.48	0.85
F_1 contradictory	0.63	0.32	0.81
F_1 irrelevant	0.15	0.17	NA
F_1 non_domain	0.31	0.59	0.35
macro-average F_1	0.49	0.46	0.74

to support training and testing of statistical classifiers. However, they show that our system, which was developed based on an analysis of the transcripts from 8 pilot sessions, performs in the same range as state-of-the-art statistical NLP approaches trained on data collected from 73 students interacting with the system (combining the data from both the ELICIT and TELL conditions). It also shows that the analysis task is intrinsically difficult, with none of the systems reaching ceiling. We describe avenues for performance improvement in the Discussion section.

Impact of Interpretation Quality

While there is much scope for improvement in interpretation quality, it is also important to understand whether such improvements will contribute to improved outcomes. It is reasonable to hypothesize that improved interpretation should lead to improved feedback and correspondingly improved overall learning gain and satisfaction. There is a long tradition of investigating the importance of different interaction parameters by correlating system performance characteristics with desired outcomes in spoken dialogue systems within the PARADISE evaluation framework (Walker et al. 2000). Similar studies have been carried out for dialogue-based ITSs (Aleven et al. 2004; Litman and Forbes-Riley 2005; Pon-Barry et al. 2004). Although such correlational analyses are not proof of causality, they can help develop actionable hypotheses about system improvement that can then be tested in user experiments (Rotaru and Litman 2009). We therefore carried out an exploratory analysis investigating the relationship between interpretation quality and learning gain in our data set.

Table 7 Comparison in evaluation scores between the ELICIT system and a majority class baseline

Metric	`All questions`	`Explain`	`Identify`
BEETLE II macro-average F_1	0.49	0.46	0.74
Baseline macro-average F_1	0.15	0.20	0.16
BEETLE II overall accuracy	0.66	0.47	0.88
Baseline overall accuracy	0.60	0.44	0.67
BEETLE II “accept-reject” accuracy	0.85	0.80	0.96
Baseline “accept-reject” accuracy	0.60	0.44	0.67

We first computed the interpretation quality scores listed in Table 5 for each student separately, and then computed the correlations between these individual scores and learning gain and user satisfaction for that student. As argued above, explanation questions constitute an important subset of our data with respect to both tutoring and to difficulties in natural language processing. Therefore, we also investigated the relationship between system performance on `Explain` questions for individual students, and their learning outcomes. The correlations we found are summarized in Table 8.[21]

The results show that overall interpretation accuracy is significantly correlated with both learning gain ($r = 0.38$, $p = 0.02$) and Tutor satisfaction ($r = 0.37$, $p = 0.03$). Recall from the previous section that the errors in the overall accuracy computation are split into two types: non-interpretable utterances, in which the system finds no possible interpretations; and misinterpretations, where the system assigns an incorrect class to the utterance. Our evaluation shows that they occur at similar rates (see Table 5). However, the results in Table 8 show that only the frequency of non-interpretable utterances correlates with learning gain and user satisfaction: (learning gain $r = -0.40$, $p = 0.02$; Overall satisfaction $r = -0.35$, $p = 0.04$; Tutor satisfaction $r = -0.48$, $p = 0.004$). In contrast, misinterpretations are not significantly correlated with either learning gain or user satisfaction. This suggests that students are sensitive to the system explicitly acknowledging interpretation failures, but may be less sensitive to non-optimal feedback.

When looking only at explanation questions, overall interpretation accuracy was not correlated with learning gain ($r = 0.24$, $p = 0.15$), but was correlated with Tutor satisfaction ($r = 0.33$, $p = 0.05$). And when error types were examined separately, this correlation was only significant for non-interpretable utterances, similar to the pattern observed in the `All Questions` dataset. This underscores the need to better understand the impact of non-interpretable utterances on learning outcomes. We examine this issue further in the next subsection.

Impact of Non-interpretable Utterances

Our results clearly show that the frequency of non-interpretable utterances is negatively correlated both with learning gain and with user satisfaction. As with the overall interpretation quality scores, one inference is that if the interpreter can be improved, this will lead to the overall reduction of error scores, and hopefully to a correlated improvement in learning gain. We outline some possibilities for improving interpretation robustness in the "Future Work" section. However, even with technology improvements we can expect that some proportion of student utterances will be difficult or impossible to interpret. This has been observed in spoken dialogue system research in general. Tutoring introduces additional difficulties for interpretation

---

[21] Macro-averaged F_1 score was not significantly correlated with any of the desired outcome metrics, and is therefore not included in the table.

Table 8 Correlations between interpretation quality metrics and outcome metrics in ELICIT. Significant correlations (with $p < 0.05$) in bold

	overall accuracy		non-interpretable		misinterpreted	
	`All questions`	`Explain`	`All questions`	`Explain`	`All questions`	`Explain`
Learning gain	**0.38**	0.24	**−0.40**	−0.27	−0.23	−0.13
Overall satisfaction	0.31	0.26	**−0.35**	−0.12	−0.12	−0.11
Tutor satisfaction	**0.37**	**0.33**	**−0.48**	**−0.46**	−0.07	−0.11

because when students struggle with unfamiliar domain terminology they sometimes say things that even human tutors find difficult to understand and evaluate.[22] Moreover, different students have different degrees of success in mimicking the terminology used by the system (Steinhauser et al. 2011). Therefore, it is important to evaluate the impact of non-interpretable utterances on learning outcomes, to gain better understanding of appropriate error recovery policies.

As discussed in the evaluation metrics section, in developing a dialogue system there is always a trade-off between the rate of non-interpretable and misinterpreted utterances. It is possible to build a system that never reports any non-interpretable utterances by having the interpreter assign a pre-determined class (e.g. correct or irrelevant) to every utterance it fails to interpret, and then letting the tutorial planner make tutoring decisions based on that analysis. However, this will mean that some utterances previously treated as non-interpretable will be misinterpreted instead, resulting in an increase in inappropriate feedback decisions. In early system pilots, we observed situations where misinterpretations caused considerable confusion for the students. We were particularly concerned about situations in which the system misinterpreted correct answers as flawed and produced feedback that further confused the students and derailed the dialogue (Dzikovska et al. 2009).

In light of the pilot results, we decided that rejecting some student utterances as non-interpretable was preferable to certain misunderstandings. Therefore, as discussed in the "System Implementation" section, we implemented an error recovery policy that gives students information about terms that were not understood by the system when an interpretation failure occurs. We also hoped that these help messages would teach students to use terminology correctly, which can in turn improve learning outcomes, since higher levels of lexical cohesion with the tutor can be positively correlated with learning gain (Ward and Litman 2006). However, given that we observed significant negative correlations between the frequency of non-interpretable utterances and both learning gain and user satisfaction, it appears that the error recovery policy implemented in the system was not effective.

---

[22] It is not possible to quantify the proportion of such utterances given our data; however, the evidence of such difficulties is occasional requests from human tutors for the student to rephrase, in our human-human corpus, and the substantial but less than perfect inter-rater agreement on evaluating the accuracy of student contributions.

In order to improve our recovery policy implementation, we performed a more detailed analysis of the features of the utterances that the system failed to interpret, and of the correlations between different help message types and learning outcomes. The first thing worth noting is that non-interpretable utterances show a markedly different distribution of answer classes from interpretable utterances, which can be seen from Table 9. In particular, recall that in the corpus as a whole 60 % of utterances are labeled correct, and 21 % are labeled contradictory. But looking at the utterances that the system fails to interpret, only 25 % are labeled contradictory, with 38 % contradictory. Among the interpretable answers, 67 % are correct, and 17 % are contradictory. In fact, answers labeled contradictory were disproportionately likely to be non-interpretable (overall, $P(nonint|\text{contradictory}) = 0.30$, while $P(nonint|\text{correct}) = 0.07$). This difference in distributions is statistically significant: $\chi^2(4) = 875.5$, $p < 0.0001$.

This analysis shows that flawed student answers are intrinsically more difficult to interpret, probably because they are more likely to contain vague or incorrect terminology that the system (and in some cases even human tutors) may find difficult to understand. We can also conclude that, relatively speaking, there is little risk involved in treating non-interpretable utterances as flawed and requesting that the student improve them.[23] Thus, our initial concern about the consequences of treating difficult-to-interpret utterances as flawed was not supported by the data. But clearly something different is needed in place of the error recovery policy that we are currently using.

We further examined the impact of interpretation failures by taking the fine-grained classification of interpretation problems developed in Dzikovska et al. (2010), and identifying four broad classes based on the cause of the problem and the way the system deals with it.

- `no-semantic-analysis-possible`: the system cannot find a full parse and a reasonable fragment combination that covers the input is also impossible; or else the system can find a parse, but does not recognize the meaning of any of the content words;[24] or (in rare instances) a crash in one of the components is preventing feedback from being generated. The system tells the student that they were not understood and asks them to rephrase (possibly with a hint).
- `identifiable-failure-point`: The system knows most of the words in the sentence, but there is an unknown word or word combination that interferes with finding a complete interpretation;[25] or otherwise there is a pronoun that the system cannot resolve. The system points out the word or word combination it hasn't understood and optionally gives a hint.

---

[23] With the obvious safeguard of providing extra reading suggestions or other helpful tutoring tactics if possible, and bottoming out and moving on after a number of failed attempts, in the same way as the system already does for utterances it diagnoses as flawed.

[24] If only some of the content words are unknown, the system decides either that they can be ignored and proceeds with a partial interpretation, or produces a more specific error message, using the approach described in Dzikovska et al. (2009).

[25] These are typically words that are in the parser's lexicon, but which the interpreter fails to understand because of the limitations of the domain-independent semantic ontology, as discussed in the "Interpretation Components" section.

Table 9 Gold standard label distribution for interpretable vs. non-interpretable subsets. Proportion of subset total in parentheses

	`All questions`		`Explain`		`Identify`	
	interpretable	non-interp.	interpretable	non-interp.	interpretable	non-interp.
correct	4480 (0.67)	334 (0.25)	1303 (0.52)	246 (0.24)	853 (0.69)	27 (0.30)
pc_incomplete	806 (0.12)	291 (0.22)	540 (0.22)	264 (0.26)	145 (0.12)	4 (0.04)
contradictory	1155 (0.17)	500 (0.38)	424 (0.17)	364 (0.35)	221 (0.18)	42 (0.47)
irrelevant	85 (0.01)	55 (0.04)	74 (0.03)	44 (0.04)	8 (0.007)	7 (0.08)
non_domain	159 (0.02)	139 (0.10)	144 (0.06)	107 (0.10)	3 (0.002)	9 (0.10)

- `wrong-input-form`: the student is submitting a circuit when the system is expecting them to type, or else are submitting an unexpected answer, e.g. typing a long sentence in response to a question requiring them to name a circuit or component, phrased in a way that leaves the answer ambiguous. The system then tells the student what answer format it is expecting to see, e.g. “Sorry, this isn’t the form of answer that I expected. I’m looking for the name of an object”.
- `restriction-failure`: student input violated knowledge base expectations. They are told what expectation is violated, with an example of a correct phrasing, e.g. “I don’t understand when you say that circuits are lit. Bulbs can be lit, but not circuits”.

The frequency of different problem types is shown in Table 10. When a problem occurs, the system attempts to provide a targeted help message that indicates the specific portion of the student input causing the problem. This can be done for all cases except `no-semantic-analysis-possible`, where there is insufficient information about the underlying cause. In addition, for `wrong-input-form` and `restriction-failure` the system is able to provide some guidance with respect to acceptable inputs.

A correlational analysis revealed that these four types of errors pattern differently with respect to learning gain and user satisfaction. The results of the correlational analyses are presented in Table 11. Students clearly found the situations where the system could give no help frustrating: there was a significant negative correlation between the frequency of `no-semantic-analysis` errors and Tutor and Overall satisfaction (Tutor: $r = -0.48$, $p = 0.004$; Overall: $r = -0.36$, $p = 0.04$). However, the frequency of such problems was not related to learning gain ($r = -0.29$, $p = 0.09$). In the two situations where the system was able to give help and clearly articulate its expectations, there was no significant correlation with user satisfaction. However, the frequency of such errors was negatively correlated with learning gain (`wrong input form`: $r = -0.38$, $p = 0.02$; `restriction failure`: $r = -0.39$, $p = 0.02$), indicating that students who repeatedly failed to phrase their answers in a way that the system could process also tended to learn less.

Table 10 Frequencies of different interpretation problems in the corpus

Error category	Count
`no-semantic-analysis-possible`	328
`identifiable-failure-point`	572
`wrong-input-form`	100
`restriction-failure`	325

These results suggest that different factors may be influencing our two primary outcome metrics, learning gain and user satisfaction. To improve user satisfaction with the system, it may be best to concentrate on reducing the number of cases where the system is unable to give any help. This is best achieved by improving overall system robustness. However, in order to improve learning gain, the focus should be placed on the `wrong input form` and `restriction failure` cases. There are two possibilities for what is going wrong here. First, it may be that the feedback currently produced is ineffective and needs to be improved. A second possibility is that such cases indicate students who are confused (perhaps struggling with domain terminology), or simply inattentive, and therefore unable to modify their answer even when they are given clear guidance about how to do so (this problem has previously been reported in tutorial dialogue Glass and Evens 2008). With such students, it may be counterproductive for the system to take responsibility for "interpretation failures"; instead, it may be helpful to devise strategies to recapture the student's attention. Determining which of these two hypotheses is correct is planned as part of our future work.

This analysis also suggests that in developing and evaluating interpretation modules for tutorial dialogue systems, different metrics may be important depending on the desired target outcomes. Most current systems place the emphasis on improving learning gain. However, improving user satisfaction is important for keeping students engaged long-term. Thus, in evaluating interpretation quality, system developers may benefit from more fine-grained analyses such as the one described in this section, in order to better target the error types related to the desired outcome metric.

Table 11 Correlations between individual interpretation failure types and learning outcomes

	`no-semantic-analysis-possible`	`identifiable-failure-point`	`wrong-input-form`	`restriction-failure`
Learning gain	−0.29	−0.03	**−0.38**	**−0.48**
Overall satisfaction	**−0.36**	−0.25	−0.02	−0.11
Tutor satisfaction	**−0.48**	−0.21	−0.30	−0.23

Discussion

We began this paper with the argument that a cyclical process of experimentation followed by explanation is an effective method for teaching conceptual material in STEM domains, but that it poses a significant challenge for the development of an NLP-enabled Intelligent Tutoring System. Our primary results have certainly borne out both pieces of this argument. Our curriculum, embedded in both versions of the system (TELL and ELICIT), yielded impressive effect sizes of close to two sigma when compared to a no-training control condition. Still, analyses of interpretation quality showed that there is significant room for improvement in processing student answers.

The next logical step is to carefully consider the evaluation data and identify those areas that appear most promising and most important for further development. First, however, we will briefly revisit our hypothesis that the NLP capability is an important component of this system and see what, if anything, our results to date have to say on this point. Recall that the second stage of the conceptual change approach uses a tutorial dialogue with the student to elicit reflection on the observed phenomena and to guide the student to generate an accurate conceptualization of the underlying principles in the domain. One of our control groups (the TELL system) was designed explicitly to test our hypothesis that the effectiveness of our system would be improved by the use of an NLP component, and our results did not yield statistically significant support for that hypothesis.

Obviously, we would have been delighted if this first version of a system using our adaptive NLP feedback (ELICIT) had proven to yield a significantly larger learning gain than the TELL system. While that was not the case, we believe that it would be premature to draw any conclusions about the potential value added by an NLP capability. First, we would argue that the results of the evaluation of the interpretation module demonstrate that the current instantiation of NLP in our system is not yet robust enough to provide a fair test of the hypothesis that incorporating NLP capabilities will improve the effectiveness of a training system using the conceptual change approach embedded in the exercise sequence.

Next, recall from the introduction that the importance of student reflection and generation is integral to several different learning theories that have been supported with multiple bodies of empirical evidence. While it is possible that properties of our domain and curriculum limit the benefits of natural language interaction, no single study would ever be sufficient to justify abandoning that component of the instructional process. Thus, improving the system's NLP capabilities remains relevant in the context of supporting research into ways of implementing effective instructional approaches in STEM.

Finally, while we did see substantial learning taking place, our students did not reach ceiling. Mean gain scores were 61 % and 65 % in ELICIT and TELL respectively—indicating that there is still room for improvement. While it is certainly possible to make changes to the static content of the TELL system, the ELICIT system affords a larger variety of manipulations. Basic principles of system development, such as the law of diminishing returns, suggest that gains are more likely to be achieved by focusing on improving weak system components and adding new capabilities than they are by fine tuning those aspects of a system that are already well developed.

Our evaluation points at a set of issues and limitations within the natural language processing module that can be addressed and evaluated in the future as part of improving the NLP components, which we discuss in the next section.

Future Work

One of the key outcomes of our evaluation is a rich data set which can be used for system improvement. The system design and implementation was informed by the analysis of dialogues from a human-human study; however, the full richness of human interaction is beyond the reach of current NLP technology, and thus we inevitably had to choose a subset of tutoring tactics that the system can use. Moreover, there are phenomena in human-computer dialogue that are not present in human-human dialogue. These include the need to deal with interpretation failures from the ITS and also negative metacognitive and social utterances from the students which may be suppressed in human-human communication because of politeness effects (Dzikovska et al. 2010; Steinhauser et al. 2010). We therefore cannot rely on human-human data collection as a guide for system behavior in those situations. Now that the system architecture and initial versions of system modules have been implemented and evaluated with a sufficient number of participants, we can use the collected data to improve the system.

The BEETLE II system uses symbolic NLP to dynamically generate feedback adapted to the problem, the state of the simulation environment and the previous dialogue history, based on a library of generic tutoring tactics. In our first evaluation, we have shown that the system is effective, but have not been able to demonstrate that the adaptive feedback in ELICIT leads to better outcomes than the TELL control condition in which the students are always shown the correct answer, without adaptive feedback.

Our evaluation focused on natural language interpretation, since it is the first stage of student input processing, and thus interpretation problems are likely to cause problems in feedback generation as well. However, there are at least four sources of potential problems in the system's behavior, and correspondingly four main areas for improvement:

- Interpretation quality
- The appropriateness of the feedback automatically generated by the system
- The choice and implementation of the generic tutoring tactics included in the library
- The tutoring policy employed in choosing the best possible tactic

While our evaluation demonstrates that interpretation quality is correlated both with learning gain and user satisfaction, it does not account for the impact of the automatically generated feedback. Two aspects of feedback quality need to be taken into account: the appropriateness of the feedback produced by the tactics already implemented in the system, and the choice of the implemented tactics. To evaluate the appropriateness of the feedback already produced by the system, tutor utterances need to be annotated for appropriateness by human raters. Similar studies have previously been carried out in tutorial dialogue (Aleven et al. 2004) and task-oriented spoken dialogue systems (Mller et al. 2007). We plan to develop an annotation scheme to enable us to annotate and evaluate feedback quality as part of our future work.

In addition, the number and variety of tutoring tactics available can affect the quality of instruction offered by the system. In devising our library of tactics, we examined the tactics that our human tutors used in the pilot study and selected the subset that were most frequently employed with successful outcomes. Our architecture is capable of supporting a larger range of tactics than is currently used by the system. For example, early on we implemented the tactic of pointing out a counter-example to a student's flawed answer based on the circuits visible in the simulation environment. We did not use this tactic in our final system because it was not part of the most frequently successful remediation sequences; however, this and other additional tactics can be implemented and evaluated in the future.

The challenge with implementing a large number of tactics is deciding which to apply if more than one tactic is applicable at a given time, as is usually the case. We chose to use a fixed policy for applying the tactics, based on common suggestions in the literature and on what we saw our human tutors do: start with contentless prompts, progress on to more specific hints, and finally give away the answer if the student is struggling. Recent work suggests that significant improvements in learning outcomes can be achieved by using statistical data analysis, either to detect common situations in the data where the current remediation policy is failing and additional tutoring can be beneficial (Forbes-Riley and Litman 2011), or by using reinforcement learning to learn the best policy (Chi et al. 2011). Our system is particularly well-placed for applying reinforcement learning techniques. Reinforcement learning methods typically require a set of actions (i.e. tutoring tactics) to be available at each point of the interaction and optimize the expected outcome by repeatedly trying different actions in similar situations in order to learn the best action to take. Since our system can dynamically generate different feedback messages in every situation, this makes it an excellent platform for reinforcement learning, which we plan to undertake in future work.

With respect to interpretation quality, the data we have collected can help in improving system performance. We recently carried out encouraging experiments showing that our symbolic interpreter can be combined with a statistical classifier to improve robustness while retaining the benefits of dynamically generated feedback in most cases. The resulting system performs significantly better on our evaluation data set than either the interpreter or the stand-alone statistical classifier (Dzikovska et al. 2013a, b). In addition, approaches have been developed to data-mine automatically parsed data sets in order to identify frequent word sequences that lead to parse failures and thus to rapidly improve coverage of grammar-based systems (Sagot and de La Clergerie 2006; van Noord 2004). These techniques can be adapted for our symbolic interpreter, thus using the collected data to efficiently improve the symbolic interpretation.

We are currently working on improving both the interpretation and the feedback generation, as discussed above, to develop and evaluate an improved version of the system, in order to investigate whether improvements in intrinsic evaluation scores will translate into improved learning gain and user satisfaction.

In addition, recall from the "Instructional Approach" section that the two key components of our approach are careful curriculum and exercise design intended to address common misconceptions, and the capability to conduct tutorial dialogue to support reflection and generation of correct explanations. Our TELL control condition does not separate out the first component. We therefore developed another "stripped-down" baseline that provides students with the same reading material and has them build the same circuits, but does not require them to make predictions or generate any explanations. We plan to evaluate this system with human participants and compare it to the system versions described in this paper, in order to separate out the contribution made by instructional design to the overall effectiveness of the BEETLE II system.

Improving the effectiveness of STEM-based Intelligent Tutoring Systems is both a high priority goal and a significant challenge for researchers and system developers. While there is clearly still much work to be done, we believe that the BEETLE II system advances the state of the art by providing natural language interpretation and generation components which integrate a simulation environment with dynamic, adaptive tutorial feedback generation, in the context of an instructional approach that requires understanding natural language input as part of reflective dialogue.

Acknowledgments This work has been supported in part by US Office of Naval Research grants N000141010085 and N0001413WX20227. We would like to thank our sponsors from the Office of Naval Research, Dr. Susan Chipman and Dr. Ray Perez, and former Research Associates who worked on this project (Peter Bell, Simon Caine, Charles Callaway, Katherine Harrison, Amy Isard, Charles Scott, and Leanne Taylor) for their contributions to this effort. The views expressed herein are those of the authors and do not necessarily reflect the official position of the organizations with which they are affiliated.

Appendix

Sample Dialogues

The sample dialogues below were all collected during system evaluation. They illustrate interaction patterns between the students and the system. For student utterances, we show the automatically annotated correctness label as described in Table 1, to give an indication of how the student answer was interpreted by the system. For tutor utterances, we provide a set of labels corresponding to tutoring tactics used. Labels are described in the next subsection.

Participant 34

A dialogue early in the process demonstrates the student making an incorrect prediction accompanied by an incorrect self-explanation. The system understands the student's answer and, when they fail to notice the discrepancy between their prediction and the actual circuit behavior, brings it to their attention. The incorrect explanation is corrected during the reflection dialogue.

MsgId	Agent	Text	System code
msg40	Tutor:	In which diagram(s) do you think that the bulbs will light up?	
msg41	Student:	1, 2, and 3	contradictory
msg42	Tutor:	OK, we'll check your prediction in a moment.	*Fdbk-Accept-Prediction, Move-On*
msg42	Tutor:	Explain your reasoning.	
msg43	Student:	The bulbs are in closed circuits	Non-Interpretable (restriction-failure)
msg44	Tutor:	I'm sorry, I'm having a problem understanding. I don't understand when you say that circuits are closed. Switches or paths can be closed, but not circuits. Please try again.	*Fdbk-Negative-Meta-computation, Ack-Bad-Bits, GBFM-Try-Again*
msg45	Student:	The paths are closed for the bulbs	pc_incomplete
msg46	Tutor:	OK, you will see what happens in a moment. When you are ready, go on to the next slide.	*Fdbk-Accept-Prediction, Move-On*
msg48	Tutor:	Build circuit 1 to test your prediction. Press the submit button when you are done. [Student builds circuit 1, and then circuits 2–5 as requested by the tutor – omitted for brevity]	
msg70	Tutor:	Did the circuit behavior match your predictions?	
msg71	Student:	Yes	contradictory
msg72	Tutor:	Actually, there was a difference. You said circuit 1 and circuit 3 correctly, but you guessed circuit 2 incorrectly. When you are ready, go on to the next slide.	*Fdbk-Negative, Ack-Bad-Bits, Call-Out-Prediction-Errors*
	Student	[Presses next slide button. The slide topic changes to "Discussion"]	
msg74	Tutor:	What are the conditions that are required to make a bulb light up?	
msg75	Student:	A closed path with a battery and bulb	correct
msg76	Tutor:	Very good. When you are ready, go on to the next slide.	*Fdbk-Positive, Move-On*

Participant 12

A fragment of reflective dialogue focusing on the same question as Participant 34 above, but where the student needs help formulating the correct answer. The system attempts two tactics (restatement and a contentless prompt in msg78 and hinting at missing object in msg80). The system keeps track of the context, and the hint in msg80 suggests a battery, which is the only object in the expected answer not yet mentioned in either of the previous turns. At the end of the dialogue, all the relevant concepts (bulb, battery, closed path) have been mentioned, but the student answer has not pulled them together into a single explicit statement. The system decides that further remediation will not be effective and instead restates the complete and correct answer at the end of the dialogue (msg82).

MsgId	Agent	Text	System code
msg76	Tutor:	What are the conditions that are required to make a bulb light up?	
msg77	Student:	There must be a closed path.	pc_incomplete
msg78	Tutor:	Yes.There must be a closed path. And then?	*Fdbk-Positive,* *Ack-Good-Bits,* *GBFM-Keep-Going*
msg79	Student:	There must be a bulb.	pc_incomplete
msg80	Tutor:	Hmmmm... Here's a hint. Your answer should also mention a battery.	*Fdbk-Hedged,* *GBFM-Hint-Object*
msg81	Student:	There must be a closed path and a battery.	pc_incomplete
msg82	Tutor:	Good so far. Here's the answer. The bulb and the battery must be contained in the same closed path. When you are ready, go on to the next slide.	*Fdbk-Hedged,* *Bottom-Out*

Participant 24

Demonstrates a hint (msg255) followed by an acceptance of a reasonable, but not ideal, answer and a restatement (msg258), which we observed in human-human tutoring (Dzikovska et al. 2008)

MsgId	Agent	Text	System Code
msg254	Tutor:	How does a damaged bulb compare to an open switch?	
msg255	Student:	both dont allow for a closed path	irrelevant
msg256	Tutor:	Hmmmm... Here's a hint. Your answer should mention a gap.	*Fdbk-Hedged,* *GBFM-Hint-Object*
msg257	Student:	both create a gap	correct
msg258	Tutor:	Very good. Both a damaged bulb and an open switch create a gap and they cause bulbs to turn off. When you are ready, go on to the next slide.	*Fdbk-Positive,* *Reinforce-ModelBetter*

Participant 9

Demonstrates an incorrect prediction (all predictions in msg399, msg401 and msg402 are incorrect). The student notices the discrepancy, but needs help in formulating the correct explanation for their results. In the reflective dialogue, the system uses a different hint, hinting at a relationship instead of the object (msg418), which helps the student arrive at the correct explanation.

MsgId	Agent	Text	System Code
msg398	Tutor:	What voltage reading do you think you would get if you compared bulb terminal 2 to bulb terminal 1?	
msg399	Student:	1.5 V	contradictory
msg400	Tutor:	OK, we'll check your prediction in a moment.	*Fdbk-Accept-Prediction, Move-On*
msg400	Tutor:	What voltage reading do you think you would get if you compared bulb terminal 4 to bulb terminal 1?	
msg401	Student:	1.5 V	contradictory
msg402	Tutor:	OK, we'll check your prediction in a moment.	*Fdbk-Accept-Prediction, Move-On*
msg402	Tutor:	What voltage reading do you think you would get if you compared bulb terminal 6 to bulb terminal 1?	
msg403	Student:	0.0 V	contradictory
msg404	Tutor:	OK, we'll check your prediction in a moment. When you are ready, go on to the next slide.	*Fdbk-Accept-Prediction, Move-On*
msg406	Tutor:	Measure the voltage between bulb terminal 1 and bulb terminal 2. [Student makes measurements for terminals that they predicted–omitted for brevity]	
msg414	Tutor:	Are those the voltage readings you expected?	
msg415	Student:	No	correct
msg416	Tutor:	Correct. In your first prediction, you said that the voltage would be 1.5 volts. But the voltage was 0.0 volts. In your second prediction, you said that the voltage would be 1.5 volts. But the voltage was 0.0 volts. In your third prediction, you said that the voltage would be zero volts. But the voltage was 1.5 volts.	*Fdbk-Positive, Ack-Bad-Bits, Call-Out-Prediction-Errors*

msg416	Tutor:	Explain why you got a voltage reading of 0 for terminal 1 and terminal 2.	
msg417	Student:	Because there was no difference	irrelevant
msg418	Tutor:	Hmmmm... Here's a hint. A terminal is connected to something.	*Fdbk-Hedged, GBFM-Hint-Relation*
msg419	Student:	Because the terminals are connected	correct
msg420	Tutor:	Very good.	*Fdbk-Positive, MoveOn*
msg420	Tutor:	Explain why you got a voltage reading of 0 for terminal 1 and terminal 4.	
msg421	Student:	the terminals are connected	correct
msg422	Tutor:	Perfect.	*Fdbk-Positive, MoveOn*

List of Tutoring Tactics

The table below lists high-level tutoring tactics used by the system. The tactics are typically combined – for example, most system utterances will use one *Feedback* tactic and either a *Reinforce* or *Remediate* tactic depending on the evaluation of the student answer.

Group	Tactic label	Description
Remediate		Tactics used after the student made a mistake.
	GBFM-Keep-Going	Go Back For More information, with a context-free prompt implying that they are on the right track but need to amend their answer.
	GBFM-Try-Again	Go Back For More information, with a context-free prompt implying that the student answer needs to be completely rephrased.
	GBFM-Hint-Object	Go Back For More information, giving a hint about an object that needs to be mentioned in the answer.
	GBFM-Hint-Relation	Go Back For More information, giving a hint about a relationship between objects that needs to be mentioned in the answer.
	GBFM-Hint-Badbits	Go Back For More information, hinting at the problematic part of the answer (but without explicitly pointing out the error).

	GBFM-Activity	Suggest that the student conduct an activity which will help them answer the question (currently, read additional instructional material).
	Call-Out-Prediction-Errors	Point out correct and incorrect predictions that the student made earlier.
	Bottom-Out	Give the correct answer to the student, after telling them that their answer is not fully acceptable.
Reinforce		Accept the student's answer as correct and choose either to move on or to follow-up with content-laden utterances.
	Move-On	Accept the answer without any other follow-up.
	Model-Better	Accept the answer, but repeat it, it in some way, e.g. by using better terminology.
Feedback (Fdbk)		Explicit evaluation of correctness of student answer.
	Positive	Indicate that the answer is correct.
	Negative	Indicate that the answer is wrong.
	Accept-Prediction	Accept the answer (a prediction) without giving direct positive or negative feedback.
	Hedged	Indicate that the student is partially correct or partially incorrect, but not explicitly, e.g. saying "pretty much".
	Negative-Metacomputation	Explicitly inform the student that the system doesn't understand what they said.
Acknowledge (Ack)		Explicitly point out a specific aspect of the student's answer as being correct or incorrect.
	Good-bits	Call out the correct aspect of a student's answer.
	Bad-bits	Call out the part of the answer that is incorrect.

References

Aleven, V., Ogan, A., Popescu, O., Torrey, C., Koedinger, K. (2004). Evaluating the effectiveness of a tutorial dialogue system for self-explanation. In Lester, J.C., Vicari, R.M., Paraguaçu, F. (Eds.) *Proceedings of the international conference on intelligent tutoring systems* (Vol. 3220, pp. 443–454). Berlin: Springer.

Aleven, V., Popescu, O., Koedinger, K.R. (2002). Pilot-testing a tutorial dialogue system that supports self-explanation. In *Proceedings of the 6th international conference on intelligent tutoring systems* (pp. 344–354). Springer.

Allen, J., Dzikovska, M., Manshadi, M., Swift, M. (2007). Deep linguistic processing for spoken dialogue systems. In *Proceedings of the ACL-07 workshop on deep linguistic processing* (pp. 49–56). Association for Computational Linguistics.

Bell, P., Dzikovska, M., Isard, A. (2012). Designing a spoken language interface for a tutorial dialogue system. In *Proceedings of 13th annual conference of the international speech communication association (INTERSPEECH 2012)*. ISCA.

Bohus, D., & Rudnicky, A. (2005). Sorry, I didn't catch that! - An investigation of nonunderstanding errors and recovery strategies. In *Proceedings of 6th SIGdial workshop on discourse and dialogue*. Lisbon: Association for Computational Linguistics.

Byron, D.K. (2002). *Resolving pronominal reference to abstract entities (Unpublished doctoral dissertation)*. University of Rochester.

Callaway, C., Dzikovska, M., Farrow, E., Marques-Pita, M., Matheson, C., Moore, J.D. (2007). The Beetle and BeeDiff tutoring systems. In *Proceedings of the SLaTE workshop on Speech and Language Technology in Education (SLaTE'07)*.

Callaway, C., Dzikovska, M., Matheson, C., Moore, J., Zinn, C. (2006). Using dialogue to learn math in the LeActiveMath project. In *Proceedings of the ECAI workshop on languageenhanced educational technology* (pp. 1–8).

Campbell, G.C., Steinhauser, N.B., Dzikovska, M.O., Moore, J.D., Callaway, C.B., Farrow, E. (2009). The DeMAND coding scheme: A "common language" for representing and analyzing student discourse. In *Proceedings of 14th International Conference on Artificial Intelligence in Education (AIED), poster session*.

Carey, S. (1986). Cognitive science and science education. *American Psychologist*, *10*, 1123–1130.

Chi, M., Jordan, P., VanLehn, K., Litman, D. (2009). To elicit or to tell: Does it matter? In *Proceedings 14th International Conference on Artificial Intelligence in Education (AIED)*. Brighton, UK.

Chi, M., VanLehn, K., Litman, D., Jordan, P. (2011). Empirically evaluating the application of reinforcement learning to the induction of effective and adaptive pedagogical strategies. *User Modeling and User-Adapted Interaction*, *21*, 137–180.

Chi, M.T.H., de Leeuw, N., Chiu, M.-H., LaVancher, C. (1994). Eliciting self-explanations improves understanding. *Cognitive Science*, *18*(3), 439–477.

Chinn, C.A., & Brewer, W.F. (1993). The role of anomalous data in knowledge acquisition: a theoretical framework and implications for science instruction. *Review of Educational Research*, *63*(1), 1–49.

Cho, B.-I. (2000). *Dynamic planning models to support curriculum planning and multiple tutoring protocols in intelligent tutoring systems (Unpublished doctoral dissertation)*. Illinois Institute of Technology.

Clark, P., & Porter, B. (1999). KM (1.4): Users manual [Computer software manual]. Retrieved from http://www.cs.utexas.edu/users/mfkb/km.

de Jong, T., & van Joolingen, W.R. (1998). Scientific discovery learning with computer simulations of conceptual domains. *Review of Educational Research*, *68*(2), 179–201.

Dietterich, T.G. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, *10*(7), 1895–1923.

Duffy, T.M., & Jonassen, D.H. (1992). In *Constructivism and the technology of instruction: A conversation*. Hillsdale: Lawrence Erlbaum Associates.

Duit, R., & Treagust, D.F. (2003). Conceptual change: A powerful framework for improving science teaching and learning. *International Journal of Science Education*, *25*(6), 671–688.

Dzikovska, M.O., Allen, J.F., Swift, M.D. (2008). Linking semantic and knowledge representations in a multi-domain dialogue system. *Journal of Logic and Computation*, *18*(3), 405–430.

Dzikovska, M.O., Bell, P., Isard, A., Moore, J.D. (2012). Evaluating language understanding accuracy with respect to objective outcomes in a dialogue system. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics* (pp. 471–481). Avignon: Association for Computational Linguistics.

Dzikovska, M.O., Callaway, C.B., Farrow, E. (2006). Interpretation and generation in a knowledge-based tutorial system. In *Proceedings of EACL-06 workshop on Knowledge and Reasoning for Language Processing (KRAQ'06)*. Trento: European Association for Computational Linguistics.

Dzikovska, M.O., Callaway, C.B., Farrow, E., Moore, J.D., Steinhauser, N.B., Campbell, G.E. (2009). Dealing with interpretation errors in tutorial dialogue. In *Proceedings of the SIGDIAL 2009 Conference: The 10th annual meeting of the special interest group on discourse and dialogue* (pp. 38–45). Association for Computational Linguistics.

Dzikovska, M.O., Campbell, G.E., Callaway, C.B., Steinhauser, N.B., Farrow, E., Moore, J.D., . . ., Matheson, C. (2008). Diagnosing natural language answers to support adaptive tutoring. In *Proceedings of the 21st International FLAIRS Conference*. Coconut Grove, Florida.

Dzikovska, M.O., Farrow, E., Moore, J. (2013a). Improving interpretation robustness in a tutorial dialogue system. In *Proceedings of the 8th workshop on innovative use of NLP for Building Educational Applications (BEA-8)*. Atlanta: Association for Computational Linguistics.

Dzikovska, M.O., Farrow, E., Moore, J.D. (2013b). Combining semantic interpretation and statistical classification for improved explanation processing in a tutorial dialogue system. In *Proceedings of the 16th international conference on Artificial Intelligence in Education (AIED 2013).* Memphis, TN, USA.

Dzikovska, M.O., Moore, J.D., Steinhauser, N., Campbell, G. (2010). The impact of interpretation problems on tutorial dialogue. In *Proceedings of the 48th annual meeting of the Association for Computational Linguistics(ACL-2010)* (pp. 43–48). Association for Computational Linguistics.

Dzikovska, M.O., Moore, J.D., Steinhauser, N., Campbell, G. (2011). Exploring user satisfaction in a tutorial dialogue system. In *Proceedings of the SIGDIAL 2011 conference* (pp. 162–172). Portland: Association for Computational Linguistics.

Dzikovska, M.O., Nielsen, R., Brew, C., Leacock, C., Giampiccolo, D., Bentivogli, L., . . ., Dang, H.T. (2013). Semeval-2013 task 7: The joint student response analysis and 8th recognizing textual entailment challenge. In *Proceedings of the 6th International Workshop on Semantic Evaluation (SEMEVAL-2013).* Atlanta: Association for Computational Linguistics.

Dzikovska, M.O., Nielsen, R.D., Brew, C. (2012). Towards effective tutorial feedback for explanation questions: A dataset and baselines. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT12)* (pp. 200–210). Association for Computational Linguistics.

Dzikovska, M.O., Steinhauser, N.B., Moore, J.D., Campbell, G.E., Harrison, K.M., Taylor, L.S. (2010). Content, social, and metacognitive statements: An empirical study comparing human-human and human-computer tutorial dialogue. In *Sustaining TEL: From innovation to learning and practice - 5th European Conference on Technology Enhanced Learning (EC-TEL 2010)* (pp. 93–108). Barcelona, Spain.

Dzikovska, M.O., Swift, M.D., Allen, J.F. (2004). Building a computational lexicon and ontology with FrameNet. In *Proceedings of LREC workshop on building lexical resources from semantically annotated corpora*. Lisbon, Portugal.

Elhadad, M., & Robin, J. (1992). Controlling content realization with functional unification grammars. In Dale, R., Hovy, E., Rösner, D., Stock, O. (Eds.) *Proceedings of the sixth international workshop on natural language generation*, (pp. 89–104). Berlin: Springer.

Engelhardt, P.V., & Beichner, R.J. (2004). Students' understanding of direct current resistive electrical circuits. *American Journal of Physics*, *72*(1), 98–115.

Eylon, B.S., & Linn, M.C. (1988). Learning and instruction: An examination in four research perspectives in science education. *Review of Educational Research*, *58*(3), 251–301.

Farnham-Diggory, S. (1994). Paradigms of knowledge and instruction. *Review of Educational Research*, *64*(3), 463–477.

Forbes-Riley, K., & Litman, D.J. (2011). Designing and evaluating a wizarded uncertainty-adaptive spoken dialogue tutoring system. *Computer Speech & Language*, *25*(1), 105–126.

Glass, M.S. (2000). Processing language input in the CIRCSIM-Tutor intelligent tutoring system. In *Papers from the 2000 AAAI fall symposium* (pp. 74–79). Available as AAAI technical report FS-00-01.

Glass, M.S., & Evens, M.W. (2008). Extracting information from natural language input to an intelligent tutoring system. *Far Eastern Journal of Experimental and Theoretical Artificial Intelligence*, *1*(2).

Graesser, A.C., Wiemer-Hastings, K., Wiemer-Hastings, P., Kreuz, R. (1999). Autotutor: A simulation of a human tutor. *Cognitive Systems Research*, *1*, 35–51.

Halloun, I.A., & Hestenes, D. (1985a). Common sense concepts about motion. *American Journal of Physics*, *53*(11), 1056–1065.

Halloun, I.A., & Hestenes, D. (1985b). The initial knowledge state of college physics students. *American Journal of Physics*, *53*(11), 1043–1055.

Hockey, B.A., Lemon, O., Campana, E., Hiatt, L., Aist, G., Hieronymus, J., . . ., Dowding, J. (2003). Targeted help for spoken dialogue systems: intelligent feedback improves naive users' performance. In *Proceedings of the tenth conference of European chapter of the association for computational linguistics* (pp. 147–154). Association for Computational Linguistics.

Jackson, G.T., Graesser, A.C., McNamara, D.S. (2009). What students expect may have more impact than what they know or feel. In *Proceedings of the 14th international conference on Artificial Intelligence in Education (AIED).* Brighton, UK.

Jordan, P., Albacete, P., VanLehn, K. (2005). Taking control of redundancy in scripted tutorial dialogue. In *Proceedings of international conference on Artificial Intelligence in Education (AIED-2005)* (pp. 314–321).

Jordan, P., Hall, B., Ringenberg, M., Cue, Y., Rosé, C. (2007). Tools for authoring a dialogue agent that participates in learning studies. In *Proceedings of the 2007 Conference on Artificial Intelligence in Education: Building technology rich learning contexts that work* (pp. 43–50). IOS Press.

Jordan, P., Litman, D., Lipschultz, M., Drummond, J. (2009). Evidence of misunderstandings in tutorial dialogue and their impact on learning. In *Proceedings of the 2009 conference on artificial intelligence in education: building learning systems that care: from knowledge representation to affective modelling* (pp. 125–132). IOS Press.

Jordan, P., Makatchev, M., Pappuswamy, U., VanLehn, K., Albacete, P. (2006). A natural language tutorial dialogue system for physics. In *Proceedings of the 19th International FLAIRS Conference* (pp. 521–527).

Katz, S., Allbritton, D., Connelly, J. (2003). Going beyond the problem given: how human tutors use post-solution discussions to support transfer. *International Journal of Artificial Intelligence in Education*, *13*(1), 79–116.

Katz, S., Connelly, J., Wilson, C. (2007). Out of the lab and into the classroom: An evaluation of reflective dialogue in andes. In *Proceedings of the 2007 Conference on Artificial Intelligence in Education: Building technology rich learning contexts that work* (pp. 425–432). Amsterdam: IOS Press.

Khuwaja, R.A., Evens, M.W., Michael, J.A., Rovick, A.A. (1994). Architecture of CIRCSIM-tutor (v.3): A smart cardiovascular physiology tutor. In *Proceedings of the 7th annual IEEE computer-based medical systems symposium* (pp. 158–163). IEEE Computer Society Press.

King, P.M. (1997). The reflective judgment model: transforming assumptions about knowing. *College Student Development and Academic Life: Psychological, Intellectual, Social, and Moral Issues*, *4*, 141.

Kolb, D.A. (1984). *Experiential learning: Experience as the source of learning and development* (Vol. 1). Englewood Cliffs: Prentice-Hall.

Larsson, S., & Traum, D. (2000). Information state and dialogue management in the TRINDI dialogue move engine toolkit. *Natural Language Engineering*, *6*(3–4), 323–340.

Lee, A.Y., & Hutchison, L. (1998). Improving learning from examples through reflection. *Journal of Experimental Psychology: Applied*, *4*(3), 187.

Litman, D., & Forbes-Riley, K. (2005). Speech recognition performance and learning in spoken dialogue tutoring. In *Proceedings of EUROSPEECH-2005* (p. 1427).

Litman, D., Moore, J., Dzikovska, M., Farrow, E. (2009). Using natural language processing to analyze tutorial dialogue corpora across domains and modalities. In *Proceedings of 14th International Conference on Artificial Intelligence in Education (AIED)*. Brighton, UK.

Litman, D., & Silliman, S. (2004). ITSPOKE: an intelligent tutoring spoken dialogue system. In *Demonstration papers at HLT-NAACL 2004* (pp. 5–8). Boston: Association for Computational Linguistics.

McCarthy, P.M., Rus, V., Crossley, S.A., Graesser, A.C., McNamara, D.S. (2008). Assessing forward-, reverse-, and average-entailment indices on natural language input from the intelligent tutoring system, iSTART. In Wilson, D., Sutcliffe, G. (Eds.), *Proceedings of the 21st International Florida Artificial Intelligence Research Society Conference* (pp. 165–170).

McDermott, L.C., Shaffer, P.S., Rosenquist, C.M., Physics Education Group Univ. Washington (1995). *Physics by inquiry: An introduction to physics and the physical sciences* (Vol. 2). New York: Wiley.

Mestre, J.P., Docktor, J.L., Strand, N.E., Ross, B.H. (2011). Conceptual problem solving in physics. In Mestre, J.P., Ross, B.H. (Eds.), *Cognition in education* (Vol. 55, p. 269–298). Academic Press.

Mller, S., Smeele, P., Boland, H., Krebber, J. (2007). Evaluating spoken dialogue systems according to de-facto standards: a case study. *Computer Speech & Language*, *21*(1), 26–53.

Narciss, S. (2004). The impact of informative tutoring feedback and self-efficacy on motivation and achievement in concept learning. *Experimental Psychology*, *51*(3), 214–228.

Narciss, S., & Huth, K. (2004). How to design informative tutoring feedback for multimedia learning. In Niegemann, D.L.H., & Brunken, R. (Eds.) *Instructional design for multimedia learning*, (pp. 181–195). Waxmann: Muenster.

Nielsen, R.D., Ward, W., Martin, J.H. (2008). Learning to assess low-level conceptual understanding. In *Proceedings 21st International FLAIRS Conference* (pp. 427–432). Coconut Grove, Florida.

Osborne, R.J., & Wittrock, M.C. (1983). Learning science: a generative process. *Science Education*, *67*, 489–508.

Papadopoulos, P.M., Demetriadis, S.N., Stamelos, I. (2009). The impact of prompting in technology-enhanced learning as moderated by students motivation and metacognitive skills. In Cress, U., Dimitrova, V., Specht, M., (Eds.), *Learning in the synergy of multiple disciplines, proceedings of the*

4th european conference on technology enhanced learning, (EC-TEL 2009) (Vol. 5794, pp. 535–548). Springer.

Pon-Barry, H., Clark, B., Bratt, E.O., Schultz, K., Peters, S. (2004). Evaluating the effectiveness of SCoT: A spoken conversational tutor. In Mostow, J., Tedesco, P. (Eds.), *Proceedings of the ITS 2004 workshop on dialog-based intelligent tutoring systems* (pp. 23–32).

Pon-Barry, H., Clark, B., Schultz, K., Bratt, E.O., Peters, S. (2004). Advantages of spoken language interaction in dialogue-based intelligent tutoring systems. In Lester, J.C., Vicari, R.M., Paraguaçu, F. (Eds.), *Proceedings of Intelligent Tutoring Systems Conference (ITS-2004)* (Vol. 3220, pp. 390–400). Springer.

Purandare, A., & Litman, D. (2008). Content-learning correlations in spoken tutoring dialogs at word, turn and discourse levels. In *Proceedings of the 21st International FLAIRS Conference*.

Redish, E.F., Saul, J.M., Steinberg, R.M. (1997). On the effectiveness of activeengagement microcomputer-based laboratories. *American Journal of Physics*, *65*, 45–54.

Ros, C.P., Torrey, C., Aleven, V., Robinson, A., Wu, C., Forbus, K. (2004). CycleTalk: Toward a dialogue agent that guides design with an articulate simulator. In Lester, J.C., Vicari, R.M., Paraguaçu, F. (Eds.) *Proceedings of the intelligent tutoring systems conference* (Vol. 3220, pp. 401–411). Berlin: Springer.

Rotaru, M., & Litman, D.J. (2009). Discourse structure and performance analysis: beyond the correlation. In *Proceedings of the SIGDIAL 2009 Conference: The 10th annual meeting of the special interest group on discourse and dialogue* (pp. 178–187). Association for Computational Linguistics.

Sagot, B., & de La Clergerie, E. (2006). Error mining in parsing results. In *Proceedings of the 21st international conference on computational linguistics and 44th annual meeting of the association for computational linguistics* (pp. 329–336). Sydney: Association for Computational Linguistics.

Scott, P.H., Asoko, H.M., Driver, R. (1992). Teaching for conceptual change: A review of strategies. In Duit, R., Goldberg, F., Niedderer, H. (Eds.) *Research in physics learning: Theoretical issues and empirical studies*, (pp. 310–329). Kiel: IPN.

Seeskin, K. (1987). *Dialogue and discovery: A study in socratic method*. SUNY Press.

Shute, V.J. (2008). Focus on formative feedback. *Review of educational research*, *78*(1), 153–189.

Sokoloff, D.R. (1996). Teaching electric circuit concepts using microcomputer-based current/voltage probes. In Tinker, R.F. (Ed.) *Microcomputer-based labs: Educational research and standards* (Vol. 156, pp. 129–146). Berlin: Springer.

Steinhauser, N.B., Butler, L.A., Campbell, G.E. (2007). Simulated tutors in immersive learning environments: Empirically-derived design principles. In *Proceedings of the 2007 interservice/industry training, simulation and education conference*. Orlando, FL.

Steinhauser, N.B., Campbell, G.E., Harrison, K.M., Taylor, L.S., Dzikovska, M.O., Moore, J.D. (2010). Comparing human-human and human-computer tutorial dialogue. In *Proceedings of the 32nd Annual Conference of the Cognitive Science Society poster session*.

Steinhauser, N.B., Campbell, G.E., Taylor, L.S., Caine, S., Scott, C., Dzikovska, M.O., Moore, J.D. (2011). Talk like an electrician: Student dialogue mimicking behavior in an intelligent tutoring system. In *Proceedings of the 15th International Conference on Artificial Intelligence in Education (AIED-2011)* (pp. 361–368). Berlin: Springer.

Swift, M. (2005). Towards automatic verb acquisition from VerbNet for spoken dialog processing. In *Proceedings of the interdisciplinary workshop on the identification and representation of verb features and verb classes*. Saarbrucken, Germany.

Tallant, D. (1993). A review of misconceptions of electricity and electrical circuits. In Novak, J. (Ed.) *Proceedings of the third international seminar on misconceptions and educational strategies in science and mathematics*, (pp. 111–120). New York: Cornell University.

Thacker, B., Kim, E., Trefz, K., Lea, S.M. (1994). Comparing problem solving performance of physics students in inquiry-based and traditional introductory physics courses. *American Journal of Physics*, *62*, 627–633.

VanLehn, K. (2006). The behavior of tutoring systems. *International Journal of Artificial Intelligence in Education*, *16*(3), 227–265.

VanLehn, K., Graesser, A.C., Jackson, G.T., Jordan, P., Olney, A., Ros, C.P. (2007). When are tutorial dialogues more effective than reading? *Cognitive Science*, *31*(1), 3–62.

VanLehn, K., Jordan, P., Litman, D. (2007). Developing pedagogically effective tutorial dialogue tactics: Experiments and a testbed. In *Proceedings of SLaTE Workshop on Speech and Language Technology in Education*.

van Noord, G. (2004). Error mining for wide-coverage grammar engineering. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL04)* (pp. 446–453). Barcelona, Spain.

Ventura, M.J., Franchescetti, D.R., Pennumatsa, P., Graesser, A.C., Jackson, G.T., Hu, X., Cai, Z. (2004). Combining computational models of short essay grading for conceptual physics problems. In Lester, J.C., Vicari, R.M., Paraguaçu, F. (Eds.) *Proceedings of the intelligent tutoring systems conference* (Vol. 3220, pp. 423–431). Berlin: Springer.

Vosniadou, S., & Brewer, W.F. (1987). Theories of knowledge restructuring in development. *Review of Educational Research*, *57*(1), 51–67.

Walker, M.A., Kamm, C.A., Litman., D.J. (2000). Towards developing general models of usability with PARADISE. *Natural Language Engineering*, *6*(3).

Ward, A., & Litman, D. (2006). Cohesion and learning in a tutorial spoken dialog system. In *Proceedings of 19th International FLAIRS (Florida Artificial Intelligence Research Society) Conference*.

Yeh, A. (2000). More accurate tests for the statistical significance of result differences. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING 2000)* (pp. 947–953). Association for Computational Linguistics.