Approximate natural language understanding for an intelligent tutor

Peter Wiemer-Hastings, Katja Wiemer-Hastings, and **Arthur C. Graesser**
Department of Psychology
University of Memphis
Memphis TN 38152

Abstract

Intelligent tutoring systems (ITS's) have a rich history of helping students in certain scientific domains, like geometry, chemistry, and programming. These domains are ideal for ITS's, because they can be easily represented and because the type of interaction between the student and the tutor can be limited to entering a few simple numbers, symbols, or keywords. Students need help in other areas, but without the ability to robustly understand a student's input, ITS's in these areas are inherently limited. Recently a technique called Latent Semantic Analysis has offered a corpus-based approach to understanding textual input which is not sensitive to errors in spelling or grammar – in fact, it pays no attention to word order at all. We are using this technique as part of an ITS system which promotes learning using natural human-like dialogue between the human and the student. This paper describes the tutoring system and Latent Semantic Analysis, and how they operate together. Then it describes our evaluation of LSA's performance by comparing its judgments with those of human raters.

Introduction[1]

In the past, intelligent tutoring systems (ITS's) have been very successful in certain types of domains. The LISP tutor helped students stay on the right track while writing programs (Corbett & Anderson 1992). Algebra and geometry tutors have been the basis for two years' high school math curriculum (Anderson *et al.* 1995). College students have learned physics from an ITS (VanLehn *et al.* 1998). Another system helped military personnel learn to troubleshoot faults in electronic equipment (Lesgold *et al.* 1992). While these systems have performed very well, the types of interaction that they support have been largely limited to pressing buttons or entering a few numbers or symbols. Even systems which accept free-form inputs from students have attempted to limit them as much as possible to single-word entries (Freedman *et al.* 1998). This limitation in the types of interaction supported by ITS's unfortunately leads to a limitation in the range of domains which they can address, or the depth of knowledge within those domains. Without significant advances in the application of natural language understanding techniques, the use of an ITS in a domain like history, for example, would be very difficult, and for philosophy would be out of the question.

Metasurveys of human tutoring research have shown that even untrained tutors, who account for the vast majority of tutors, enhance learning by 0.4 to 2.3 standard deviation units over classroom teaching (Cohen, Kulik, & Kulik 1982; Bloom 1984). Our group's studies have shed light on the types of dialogue that untrained human tutors use in tutoring situations. They have also shown that human tutors do not develop an in-depth model of their students' knowledge (Person *et al.* 1994; Graesser & Person 1994). Rather they have an approximate understanding of whether a student's answer is anywhere close to the answer they are expecting. We are currently developing an ITS called AutoTutor which serves as a testbed for research into ways of supporting smooth dialogue interaction for tutoring in different types of domains. The key element for our ability to accomplish this is the evaluation of the student's responses using a corpus-based natural language processing mechanism called Latent Semantic Analysis (LSA). This mechanism uses a statistical technique as a basis for its judgments of the meanings of the words and sentences in a student's responses. Because it does not rely on syntax, AutoTutor is not affected by ungrammatical or colloquial texts. And it exhibits graceful degradation, focussing on the parts of a text that it does know.

This paper describes at a high level how LSA makes its judgments of text similarity. Then we present the overall structure of AutoTutor. In the fourth section, we give details of AutoTutor's use of LSA and how the student contributions are assessed. The fifth section describes our tests of LSA in comparison to human ratings.

LSA

LSA was originally developed as a method of information retrieval: selecting from a large database of texts the subset which matches a query (Deerwester *et al.* 1990). The simplest approaches to this task match

[1]

keywords from the query with words in the text and select those texts with the highest number of keywords. However these approaches are incapable of recognizing synonyms or alternate senses of keywords. LSA was developed as an attempt to circumvent these limitations. It uses a statistical technique called singular value decomposition (SVD) to reduce a *terms* × *documents* co-occurrence matrix to a set of three much smaller matrices of rank K which describe a K-dimensional semantic space. The first of these three resultant matrices specifies the K-dimensional vector in this space for each of the terms, the second gives the vectors for the documents, and the third gives the relative importance of each of the K dimensions. The sum of the vectors for the terms in a document equals (when normalized) the vector for the document. The product of the three smaller matrices is an approximation to the original co-occurrence matrix. Thus, SVD is a type of data compression mechanism. The functional relevance of this is that the reduced representation must generalize across similar instances of words occurring in similar contexts to best reproduce the original data. This results in a representation where similar terms and similar documents have similar vectors in the K-dimensional space. The distance between these vectors, viewed as the similarity between the terms or documents, is easily measured by taking the cosine between them. Several other researchers have shown that LSA can approach the performance of humans in domains such as the TOEFL synonym test (Landauer & Dumais 1997), grading college essays (Foltz, Kintsch, & Landauer 1998), and lexical acquisition (Landauer & Dumais 1997). We claim that LSA captures the latent semantic information in the co-occurrence of similar words in similar contexts in a corpus.

AutoTutor

Among ITS's, AutoTutor is rather strange. The primary goal of the project is not to demonstrate impressive learning gains in students (although we hope that it will). Instead it is meant as a research tool for exploring the types of knowledge and processing that are necessary to produce smooth dialogue interaction in a tutoring environment.

The domain that we chose for AutoTutor was computer literacy. There were several reasons for this choice. On the practical side, it is a required course at the University of Memphis, and several members of the Tutoring Research Group had experience teaching the class. More importantly, we chose this domain because it is unlike the more scientific domains described earlier. The computer literacy class does not focus on the details of computer design and programming. Rather it deals with the high-level issues involved in using computers and the role of computers in society. Within computer literacy, we chose a topics that would exhibit a range of specificity: computer hardware, software, and the internet.

The basic mode of operation for AutoTutor is very much like that between a human tutor and a human student. The participants interact through conversation. Thus, the primary interface consists of a "talking head" that produces the tutor's speech with facial expressions and gestures, and a window into which the student can type her response (We will explore speech input in a later phase of the project.) The flow of the tutoring session is simple: the tutor chooses a topic, optionally presents textual and/or graphical information, and asks a question – not a yes-or-no or single-word-answer question, but a deep reasoning question intended to ellicit explanatory reasoning from the student (Graesser, Baggett, & Williams 1996). Such "how", "what", or "why" questions prompt the student to construct the types of causal explanations or logical justifications that are required in less cut-and-dried domains. We use these types of questions in our curriculum script:

Question answer A question with an expected answer

Didactic content A question that is preceded by an information delivery item

Picture question answer Like question-answer, but includes a graphic

Problem solution A problem scenario for which the student should find a solution

An example of an easy question-answer item from the hardware portion of our computer literacy curriculum script is:

What does the CPU do in the computer?

A moderately difficult didactic content question from the software topic with its information delivery content is:

Most of the time the operating system works behind the scenes, taking care of business without the knowledge or intervention of the user. But sometimes it is necessary for the user to communicate directly with an operating system in order to help run a program. How is the operating system used when a person opens an application program?

This difficult picture question answer item from the software topic is accompanied by an animation that shows motion between computer components as the information delivery content is presented:

When the user turns on the computer, the operating system is booted into RAM. The operating system provides services to application programs that the user wants to run. For example, suppose that the user wanted to use a word processing program to create a document. How does the operating system interact with the word processing program when the user wants to create a document?

Finally, here is a moderate problem solution item from the internet area:

During the cold war, US military strategists were concerned that an enemy attack would cripple the US government's ability to communicate. The Department of Defense wanted a network that could function even if some connections were destroyed. Imagine that you are a computer engineer during that time and have been asked to help build a small experimental network. How will you design a network that will still function even if some connections are destroyed by an enemy?

After the student hears the information delivery (if it exists for this question), watches the animation (likewise), and hears AutoTutor ask the question, she types in her response and pushes the return key. Then the tutor evaluates the response with respect to how well it covers the expected answer to the question. Based on this evaluation, AutoTutor picks its next dialogue move from a set of moves observed in human tutoring sessions (Person *et al.* 1994; Hume *et al.* 1996). Example dialogue moves are pumps for more information, elaborations which trigger related information, hints, and summaries. The dialogue moves and the rules which produce them are described elsewhere (Graesser *et al.* 1998; Wiemer-Hastings *et al.* 1998). When the topic has been sufficiently covered, AutoTutor chooses a new question, and the procedure is repeated. In short, the tutoring session resembles a human-to-human tutoring session.

AutoTutor's language analysis consists of segmenting the student's response into speech acts (sentences, initially), classifying them, and then evaluating them, primarily with LSA, but also with other text evaluation mechanisms. We focus on the LSA evaluations in this paper, and describe them in the next section. The basic knowledge representation structure for AutoTutor is a curriculum script (Putnam 1987), a declarative structure which holds the questions that the tutor can ask in the tutoring domain. As mentioned earlier, each question is a deep reasoning or problem-solving question, and an ideal answer to the question will cover many different aspects of the topic. For each aspect, there is a prototype good answer, and a set of associated dialogue moves. For each question, there is also a set of bad aspects, or common misconceptions, and associated dialogue moves. The next section describes how LSA evaluates the similarity between a student answer and these aspects. Based on these similarity judgments, AutoTutor makes its decision about what dialogue move to use next.

AutoTutor takes a Vygotskyan zone-of-proximal-development approach by focussing on the aspect of the current topic which has the highest LSA rating that is below an empirically determined threshold (more on this below). Thus, AutoTutor steers the conversation toward those aspects of the topic which don't have a high enough similarity to what the student has typed in. If a student answer has a high similarity to one of the bad aspects for the current question, AutoTutor may choose a splice, as human tutors do, substituting the correct information for the erroneous answer. When enough of the aspects of the current question have achieved an LSA rating over the threshold, AutoTutor chooses the next topic, using the difficulty ratings in the curriculum script, and its assessment of the overall ability of the student.

Evaluation of student contributions

We trained LSA on a corpus of 2 computer literacy textbooks and 10 articles or book chapters from each of our three topic areas. As the textual units for training LSA, we used the paragraphs of this corpus, because paragraphs tend to hold a semantically unified, well-encapsulated idea. We also included the items from our curriculum script in the corpus. The resulting corpus contained approximately 8100 documents, and 8400 terms that occurred in more than one document. We chose 200 dimensions as the target dimensionality of the SVD process, and a cosine threshold of 0.5. (Subsequent analyses will empirically determine the optimal number of dimensions and threshold to use.) When AutoTutor receives a student response to a question, LSA derives the vector of each content-related speech act by adding the vectors of the terms therein. Then it calculates the cosine between these vectors and the vectors of the documents corresponding to the set of good and bad aspects for the current question. If the cosine is above the threshold, that particular aspect of the question is counted as "covered". If not, it is a candidate for further discussion in the session.

This approach led to the definition of two measures of the quality of a student contribution:

Completeness The percentage of the good aspects of a question which achieved an above-threshold LSA cosine with a student speech act. In other words, this is the amount of the expected ideal answer that has been covered in this tutoring session.

Compatibility The percentage of the speech acts of a student contribution which achieved an above-threshold LSA cosine with one of the good aspects. This is the amount of the student's response which matches the expected answer.

In addition to allowing AutoTutor to determine what dialogue move to employ next, these measures are used to calculate an overall assessment of the student's ability, and they were used in the comparison with human ratings described in the next section.

Evaluation of LSA in AutoTutor

To evaluate the performance of LSA's assessment of student contributions, we presented eight of the questions from each topic of the curriculum script to the students in the computer literacy class. The questions were presented in a word processing document because the AutoTutor interface was not yet completed. To earn extra credit for the class, each student was asked to provide

complete answers for twelve of the questions which were randomly assigned. This data collection yielded approximately 30 answers per question. From this set, we randomly selected eight answers for each of the 24 questions for our test set. For example, here are three student answers for the internet question, "How is an operating system like a communications coordinator?"

1. It controls the use of the hardware.
2. It communicates with the peripherals.
3. It coordinates processing of programs.
4. Makes it possible to run programs simultaneously by switching back and forth between them.
5. It finds a program on disk or in memory and loads into main memory.

Four human raters evaluated these answers. Two raters were subject area experts: a graduate student and a postdoc in computer science. The other two raters were a graduate student and a professor in psychology. They had intermediate level knowledge of the domain; they had read all of the related information from the computer literacy course and participated in the development of AutoTutor. These raters were asked to break down each of the student answers and each of the curriculum script's ideal answers for the questions into propositions, i.e. information that could stand alone in a sentence. Then they were asked to compute the compatibility measure by calculating the percentage of the propositions in a student answer that matched one of the propositions in the ideal answer. They calculated the completeness score by taking the percentage of ideal answer propositions that were matched by a student answer proposition.

Note that the raters did not use the good aspects of the ideal answer mentioned above. At the time of this rating exercise, the good aspects had not been added to the curriculum script. They were added before the LSA ratings were calculated, and this mechanism used the number of good aspects as the denominator of its completeness measure. Because there was very low correlation between the number of propositions our raters found in the ideal answers and the number of good aspects, there was a corresponding low correlation between the completeness measures of LSA and the human raters. The compatibility scores were based on the same measure of LSA "matching" a student answer, and so we report the correlations between the compatibility scores here.

The correlation between average intermediate rater score and the average expert rater score on the compatibility measure was $r = 0.76$. The correlation between the two expert raters was $r = 0.78$. The correlation between the two intermediate raters was $r = 0.52$.

To demonstrate how LSA computes the compatibility score, we continue with the previous example. The three good aspects for the question presented above are:

1. Some of the most complex tasks performed by a computer involve communicating with screens, printers, disk drives, and other peripheral devices
2. A computer's operating system includes programs that take care of the details of communication with peripherals.
3. Communications with peripherals require memory resources, therefore, to continuously run programs, respond to the user's commands, and operate peripherals all at the same time, the operating system must coordinate all operations so that they all run efficiently.

The cosines between the previously presented student answer sentences and these good aspects are shown in table 1. Because only one of the cosine values exceeded the threshold of 0.5, the compatibility score for this student answer was 0.2. This rating seems intuitively reasonable; although the student answer covered a range of aspects of the question, it was rather vague. Note that this set of maximum cosines would lead AutoTutor to focus on aspect 2, because it has the highest sub-threshold value. AutoTutor would chose the corresponding elaboration, hint, or prompt to attempt to get the student to say more about this aspect.

Table 1: Cosines between speech acts and ideal answer aspects

Sentence	Aspect1	Aspect2	Aspect3	Max
1	0.04	-0.01	-0.01	0.04
2	0.15	0.46	0.51	0.51
3	0.05	0.11	0.14	0.14
4	0.03	0.14	0.29	0.29
5	0.13	0.05	0.13	0.13

The correlation between the LSA compatibility scores and the average human compatibility scores was $r = 0.47$, virtually equivalent to the correlation between the human raters. For us, this is very good news. Most human tutors are not domain experts. They are classroom peers, who know the material only slightly better than their tutees (Cohen, Kulik, & Kulik 1982). Yet they still produce significant learning gains. The first version of AutoTutor simulates a normal, untrained tutor. In the last phase of the project, we will focus on implementing sophisticated tutoring strategies like modeling, scaffolding, fading, and a socratic tutor.

Conclusions

ITS's have a proven record of supporting learning in certain scientific or engineering domains. They have not been extended to more theoretical domains at least in part because of the difficult of understanding student responses to questions. We believe that LSA is a mechanism that can robustly, and with relatively low training costs, provide the needed assessments. With

this capability, AutoTutor can carry on an extended conversation, fostering student construction of knowledge and enabling richer evaluation of that knowledge. We believe that this is a critical step in the evolution of intelligent tutoring, a step that is necessary for increasing the range of domains that ITS's can handle.

In this work we have chosen computer literacy as the tutoring domain on practical and theoretical grounds. Our approach is not limited to this domain, however, and future work on the project will stress the development of tools for porting AutoTutor to other areas in which "how", "what", and "why" questions are central.

Acknowledgments

This project is supported by grant number SBR 9720314 from the National Science Foundation's Learning and Intelligent Systems Unit. The research reported here was completed with the assistance of the other members of the Tutoring Research Group at the University of Memphis: Ashraf Anwar, Myles Bogner, Patrick Chipman, Scotty Craig, Rachel DiPaolo, Stan Franklin, Max Garzon, Barry Gholson, Doug Hacker, Xiangen Hu, Derek Harter, Jim Hoeffner, Jeff Janover, Bianca Klettke, Roger Kreuz, Kristen Link, Johanna Marineau, Bill Marks, Lee McCaulley, Michael Muellenmeister, Fergus Nolan, Brent Olde, Natalie Person, Victoria Pomeroy, Melissa Ring, Yang Yang, Holly Yetman, and Zhaohua Zhang.

References

Anderson, J. R.; Corbett, A. T.; Koedinger, K. R.; and Pelletier, R. 1995. Cognitive tutors: Lessons learned. *The Journal of the Learning Sciences* 4:167–207.

Bloom, B. S. 1984. The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. *Educational Researcher* 13:4–16.

Cohen, P. A.; Kulik, J. A.; and Kulik, C. C. 1982. Educational outcomes of tutoring: A meta-analysis of findings. *American Educational Research Journal* 19:237–248.

Corbett, A. T., and Anderson, J. R. 1992. LISP intelligent tutoring system: Research in skill acquisition. In Larkin, J. H., and Chabay, R. W., eds., *Computer-assisted instruction and intelligent tutoring systems.* Hillsdale, NJ: Erlbaum. 73–110.

Deerwester, S.; Dumais, S. T.; Furnas, G. W.; Landauer, T. K.; and R., H. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science* 41:391–407.

Foltz, P. W.; Kintsch, W.; and Landauer, T. K. 1998. The measurement of textual coherence with latent semantic analysis. *Discourse Processes* 25:285 -308.

Freedman, R.; Zhou, Y.; Glass, M.; Kim, J.; and Evens, M. 1998. Using rule induction to assist in rule construction for a natural-language based intelligent tutoring system. In *Proceedings of the 20th Annual Conference of the Cognitive Science Society*, 362–367. Hillsdale, NJ: Erlbaum.

Graesser, A. C., and Person, N. K. 1994. Question asking during tutoring. *American Educational Research Journal* 31:104–137.

Graesser, A.; Baggett, W.; and Williams, K. 1996. Question-driven explanatory reasoning. *Applied Cognitive Psychology* 10:S17–S32.

Graesser, A. C.; Franklin, S. P.; Wiemer-Hastings, P.; and the Tutoring Research Group. 1998. Simulating smooth tutorial dialogue with pedagogical value. In *Proceedings of the 11th International Florida Artificial Intelligence Research Symposium Conference*, 163–167. AAAI Press.

Hume, G. D.; Michael, J.; Rovick, A.; and Evens, M. W. 1996. Hinting as a tactic in one-on-one tutoring. *The Journal of the Learning Sciences* 5:23 47.

Landauer, T., and Dumais, S. 1997. A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review* 104:211–240.

Lesgold, A.; Lajoie, S.; Bunzo, M.; and Eggan, G. 1992. Sherlock: A coached practice environment for an electronics troubleshooting job. In Larkin, J. H., and Chabay, R. W., eds., *Computer-assisted instruction and intelligent tutoring systems.* Hillsdale, NJ: Erlbaum. 201–238.

Person, N. K.; Graesser, A. C.; Magliano, J. P.; and Kreuz, R. J. 1994. Inferring what the student knows in one-to-one tutoring: The role of student questions and answers. *Learning and Individual Differences* 6:205–229.

Putnam, R. T. 1987. Structuring and adjusting content for students: A study of live and simulated tutoring of addition. *American Educational Research Journal* 24:13–48.

VanLehn, K.; Niu, Z.; Siler, S.; and Gertner, A. 1998. Student modeling from conventional test data: A bayesian approach without priors. In Goettl, B.; Halff, H.; Redfield, C.; and Shute, V., eds., *Intelligent Tutoring Systems, Proceedings of the 4th international conference*, 434–443. Berlin: Springer.

Wiemer-Hastings, P.; Graesser, A.; Harter, D.; and the Tutoring Research Group. 1998. The foundations and architecture of AutoTutor. In Goettl, B.; Halff, H.; Redfield, C.; and Shute, V., eds., *Intelligent Tutoring Systems, Proceedings of the 4th international conference*, 334–343. Berlin: Springer.