Natural Language Engineering **12** (2): 195–208. © 2006 Cambridge University Press
doi:10.1017/S1351324906004153 Printed in the United Kingdom

Some useful tactics to modify, map and mine data from intelligent tutors

JACK MOSTOW and JOSEPH BECK
Project LISTEN, School of Computer Science, Carnegie Mellon University, RI-NSH 4213, 5000 Forbes Avenue, Pittsburgh, PA 15213-3890, USA
e-mail: mostow@cs.cmu.edu,joseph.beck@gmail.com

(*Received 1 May 2005; revised 17 November 2005*)

Abstract

Mining data logged by intelligent tutoring systems has the potential to discover information of value to students, teachers, authors, developers, researchers, and the tutors themselves – information that could make education dramatically more efficient, effective, and responsive to individual needs. We factor this discovery process into tactics to *modify* tutors, *map* heterogeneous event streams into tabular data sets, and *mine* them. This model and the tactics identified mark out a roadmap for the emerging area of tutorial data mining, and may provide a useful vocabulary and framework for characterizing past, current, and future work in this area. We illustrate this framework using experiments that tested interventions by an automated reading tutor to help children decode words and comprehend stories.

1 Introduction

The ability of intelligent tutoring systems to log and pool detailed, longitudinal interactions with large numbers of students could create a gold mine of educational data (Beck 2004). For example, during the 2003–2004 school year, some 200 computers running Project LISTEN's Reading Tutor (Mostow and Aist 2001) at nine elementary schools logged 54,138 sessions, 162,031 story readings, 1,634,660 sentences, 3,555,487 utterances, and 10,575,571 words of children's oral reading.

What information is worth extracting from this gold mine? Faculty, students, administrators, technical support staff, content authors, software developers, researchers, and the tutor itself differ in what content and form of information they are interested in and can understand. Faculty may need simple summaries of student usage and progress; administrators need evidence of tutor effectiveness; technical support staff need problem alerts; content authors need usability indicators; developers need accurate bug reports; researchers need detailed examples and informative analyses; and the tutor needs parameters it can use, rules it can interpret, or knowledge it can exploit. Such informational goals are typically not clear at the outset, instead emerging from and in turn guiding successively refined instrumentation and analyses.

An important goal of educational data mining is to discover what helps – specifically, which tutor and student actions help which students learn which skills in which contexts. This goal poses a credit assignment challenge complicated by student variability, the duration, richness, uniqueness, and irreproducibility of tutorial interaction, and the fact that changes in students' knowledge are not directly observable and can be estimated only indirectly and imperfectly from their observed behavior.

Controlled studies of students' pre- to post-test gains can evaluate a tutor compared to various alternatives, including independent practice (Mostow, Aist, Bey, Burkhead, Cuneo, Junker, Rossbach, Tobin, Valeri and Wilson 2002b; Poulson 2004), classroom instruction (Mostow, Aist, Huang, Junker, Kennedy, Lan, Latimer, O'Connor, Tassone, Tobin and Wierman, in press), and human tutors (Mostow, Aist, Burkhead, Corbett, Cuneo, Eitelman, Huang, Junker, Sklar and Tobin 2003). However, such comparisons tell only how well the tutor works overall. To understand and improve the tutor's effectiveness requires finer-grained analyses. How, then, can we use data from intelligent tutors to pursue this goal?

We outline an emerging approach based on experience in analyzing data from various tutors. This approach consists of *modifying* the tutor to obtain useful data, *mapping* heterogeneous streams of logged events into tabular data, and *mining* that data. Sections 2–4 describe these three phases. Section 5 illustrates them, and section 6 concludes the paper.

2 Modify tutor instrumentation

The types of tutor data available to analyze depend on which activities are instrumented in machine-analyzable form. But what makes such data mineable? How can the tutor be modified to capture usefully mineable data?

2.1 Log tutorial events

Learning by doing is essential in education, and takes multiple forms and names, such as homework, practice, problems, exercises, labs, simulations, and explorations. The products of such student work can be useful to analyze. But the *process* of task performance can be even more useful to instrument. A tutor can record whatever student activities it involves, such as reading, writing, taking tests, performing various tasks in real or virtual environments, even communicating with peers. Logged input may include mouse clicks, typing, and speech. Logged output may include text, graphical actions, and audio. Logged tutorial decisions may include not only actions chosen but alternatives rejected. For example, logging not only tutorial prompts but also decisions to skip them made it possible to analyze their effect on student behavior (Mostow and Aist 2001).

The level at which events are logged constrains their analysis. For example, logging a mouse click by its *x* and *y* coordinates may help analyze the student's motor skills. In contrast, logging the menu item clicked on may allow scoring of the student's reply to a multiple choice question. Likewise, logging tutor audio output 'kuh ae tuh' may allow a human to infer what the tutor did, but the higher-level description 'SoundOut *cat*' allows machine analysis.

The granularity of logged events also affects analysis. For example, measuring the usage of a reading tutor requires data about the duration and frequency of student sessions. In contrast, data at the level of individual read words is useful for much finer-grained analyses. Logging data at multiple grain sizes supports viewing and analyzing tutor data at different levels of detail (Mostow, Beck, Chalasani, Cuneo and Jia 2002c).

2.2 Time events

Timestamping when each event starts and ends is simple but useful. For example, timing responses to multiple-choice questions can detect hasty responses even when they are correct (Mostow, Beck, Bey, Cuneo, Sison, Tobin and Valeri 2004). Timing how long students spend in different activities can shed light on their motivation and help predict their gains (Mostow, Aist, Beck, Chalasani, Cuneo, Jia and Kadaru 2002a). To help analyze students' reading, a tutor can log which text students see, at whose initiative (student or tutor), when, and how many times. It can log how long they take to read each document, page, sentence (by making them click for each new sentence), or even word (by using speech recognition to listen to them read aloud). Speed can be a useful measure of student performance, and speedup can be a useful measure of learning. For example, the time to read words aloud can be used to estimate the student's oral reading proficiency (Beck, Jia and Mostow 2004a) and to analyze fluency improvement over time (Mostow and Aist 1997; Mostow and Beck 2005).

2.3 Reify operations to log them analyzably

To be useful, logged data must be *machine-understandable*. Even a complete record of the student's brain activity would be useless without some way to extract useful information from it. Many tasks are important to student learning but hard for machines to observe or analyze, such as solving a math or physics problem on paper. Reification renders such processes machine-analyzable by reorganizing them to use operations that are easier for computers to instrument. For example, reification may replace handwritten input with typed input, freehand drawing with a limited palette of graphical objects and operations (Koedinger and Anderson 1993), and free-form responses with menu selections (Self 1988) or at least canned 'sentence openers' (Soller 2001; Goodman, Hitzeman, Linton and Ross 2003).

2.4 Randomize tutorial decisions

Allocating credit and blame over the long sequence of tutor and student decisions that lead to a given educational outcome is hard. One approach to making this problem somewhat more tractable is to embed randomized experiments in the tutor. For example, one such experiment (Aist 2001) randomly chose whether or not to explain a new word in a story, such as *replied*. If so, the Reading Tutor presented a short, automatically generated explanation of the word, such as *reply is a kind of tell*. The next day the student used the Reading Tutor, it tested the vocabulary words whether or not it had explained them. It tested each word using an automatically generated multiple choice question, such as *What word is MOST like replied? soft-pedal; observe; play down; answer*. Analysis of over 3,000 such randomized trials clarified when it helped to explain new words, compared to just letting students encounter them in context. Explaining common words like *apple* did not help, but explaining rare, single-sense words like *astronaut* boosted students' performance significantly on the test questions.

2.5 Probe student knowledge

Although many tutors include explicit probes as part of their normal interactions, adding probes to a tutor may provide more information than can be inferred from its normal interactions. For example, to test understanding of both the explained and unexplained words in the experiment above, Aist (2001) modified the Reading Tutor to generate and administer multiple choice questions about them the next day the student used the tutor. Likewise, a later modification probed students' text comprehension by automatically inserting multiple-choice cloze questions. The cloze method turns a sentence into a comprehension question by deleting a word for the student to fill in based on comprehending the rest of the text. These questions served not only as test items to assess students' comprehension, but also as the outcomes of randomized trials to test tutorial interventions (Mostow *et al.* 2004). Another example of adding a probe is the Piagetian development test that enabled the AnimalWatch tutor to tailor its actions to students' cognitive development (Arroyo *et al.* 2000).

2.6 Import student data

Analysis of tutor interactions can benefit from student data that the tutor might not be able to observe, such as gender (Arroyo, Beck, Woolf, Beal and Schultz 2000), age, IQ (Shute, Gawlick-Grendell, Young and Burnham 1996), prior declarative knowledge (Corbett, McLaughlin and Scarpinatto 2000), or pretest scores. Such data can be input to the tutor by the student or teacher, or obtained separately for analysis purposes (Mostow, Beck, Chalasani, Cuneo and Jia 2002c). Even when the extra data would not be feasible for a production version of the tutor to collect, it can still be valuable in analyzing research versions of the tutor.

2.7 Analyze natural language

Reading and writing are vital to tutoring. A tutor cannot directly observe students' reading comprehension, but can record which text students see, at whose initiative (student or tutor), when, and how many times. Data on students' writing may include not only their typed input itself, but also response time, duration, and even keystroke-level information. Evaluating matches of student responses to expected answers is more tractable than understanding natural language in general, with solutions ranging in depth from spell-checking, to using simple keyword analysis to grade short-answer questions, to using latent semantic analysis to grade students' essay questions (Calfee, Kukich, Landauer, Laham, Foltz, Hirschmann, Breck, Burger and Ferro 2000), to using parsing and domain-specific knowledge to analyze and critique students' self-explanations of proof steps (Aleven, Popescu and Koedinger 2002). Communication with peers is important in some tutors, but can be difficult to instrument. One possibility is to monitor computer-mediated channels such as email, newsgroups, and on-line chat, or channels built into the tutor itself (Goodman, Hitzeman, Linton and Ross 2003). Analysis of computer-mediated peer communication could range from tracking topic content using information retrieval methods, to quantifying communication frequency, volume, and patterns of who communicates with whom (Vassileva, Greer, McCalla, Deters, Zapata, Mudgal and Grant 1999), to tracing finer-grained effects of peer communication on learning (Soller 2004a, 2004b).

2.8 Hand-label data

Although a tutor can capture too much data to inspect by hand, manual analysis of a strategic sample can be very helpful. For example, manual transcription of selected speech input is useful when automated speech recognition is not accurate enough. One such experiment tested the student's ability to read certain words in isolation. The tutor recorded these tests and sent them back to transcribe by hand, revealing significant differences in efficacy among alternative methods for previewing words before a story (Mostow in press).

3 Map events to variables

Instrumenting a tutor yields a rich but heterogeneous stream of events over time. A position or length along the flow of the stream corresponds to the temporal dimension of tutorial interaction. The breadth of the stream corresponds to the variety of skills or topics involved. How can we translate such data into variables to visualize and analyze, i.e., tabular data sets?

3.1 Segment an event stream into episodes by partitioning it at specified cut points

One way to simplify a complex stream of interaction is to parse it into shorter episodes that can be analyzed individually. Segmentation preserves the breadth of the interaction stream but cuts it into segments of shorter duration. For example, one study (Beck, Woolf and Beal 2000) segmented tutorial dialogue at each student response to a tutorial action. Outcome variables included the time it took the student to respond, and whether the response was correct. To characterize the context, 48 variables encoded various features of the student, the problem, and recent instruction, such as the amount of help provided. Segmenting the data yielded thousands of such episodes to train a model of student behavior.

3.2 Slice an event stream into strands for distinct skills

A powerful way to abstract a stream of interactions is inspired by 'program slicing' (Weiser 1984), a method to simplify analysis of a computer program by considering only the parts that affect particular variables. In mining a stream of tutorial interactions, the idea of slicing is to focus only on interactions relevant to a given target skill. Unlike segmentation, slicing preserves the length of the interaction stream but factors it into narrow strands, one for each target. The assumption that slices are independent of each other simplifies their analysis. For example, knowledge tracing (Corbett and Anderson 1995) 'slices' students' problem-solving into opportunities to apply different skills, with one slice for each skill. Plotting students' performance at successive opportunities to apply a given rule reveals a systematic learning curve for each skill, in contrast to the chaotic trajectory of performance on successive complete problems, which exercise multiple skills.

3.3 Reformulate an event stream as a set of trials, each with its own context, decision, and outcome

To analyze the complex effects on eventual educational outcomes of extended, rich interaction with a tutor, it helps to decompose that interaction into a series of experimental trials defined by local decisions. Each trial starts with a decision that occurs while (or before) a student uses the tutor, and affects the ensuing tutorial interaction. We have used this approach to extract dozens, hundreds, or even thousands of data points per student from their tutor use, enabling us to harness the power of 'big data' to resolve fine-grained effects of tutor behavior on student learning.

Some kinds of trials are more conducive than others to drawing well-supported causal inferences. A hardwired decision that always chooses the same option is hard to analyze because it provides no basis for comparison. Decisions made by the tutor are easier to analyze than decisions made by the student, because the effects of student-influenced decisions are hard to tease apart from other student effects, such as intrinsic characteristics of the student. For example, if students who pick harder stories make greater gains in reading, which is cause and which is effect? Randomized decisions are the easiest to analyze and provide the strongest evidence to support causal inferences. Ideally trials are independent, but complications can occur, as we shall shortly discuss.

Each trial occurs in a context, such as a particular student in a particular class encountering a particular word in a particular story on a particular computer at a particular time on a particular date. The characterization of the context as a set of features serves as a basis for aggregating or disaggregating trials to relate context to outcome.

Each trial culminates in an outcome. If the experimental outcome is formulated in advance, the tutor may be able to explicitly log each outcome as soon as it occurs. Logging each trial of an experiment as a single record in a table specific to that experiment, with fields encoding the experiment's context, decision, and outcome, simplifies analysis. For example, an experiment comparing different ways to preview new words before reading a story posttested each word after the story, logging the posttests as trial outcomes (Mostow *et al.* 2004).

Although logging entire trials at once simplifies analysis, it is not always appropriate to do. One reason is that the outcome may not be known until later. For example, a vocabulary experiment with a delayed posttest (Aist 2001) could not log the outcome of a trial until the next day the student used the tutor again.

Another reason is that explicit tests take time, can interrupt learning, and may not measure the target skill well. An alternative is to define the outcome of a trial as the student's performance at the next encounter of the target skill. This approach not only avoids the disadvantages of explicit tests, but also provides a more naturalistic measure of a tutorial intervention's effect on student learning. For example, one such experiment defined the outcome of tutorial help on a word in terms of the student's performance on that word in a later sentence. This approach defined over 180,000 trials in the data set for a single school year – enough to detect subtle but statistically significant differences in efficacy between alternative types of help.

A third reason not to log entire trials at once is that the outcome variable may not be defined until after the fact. The ability to design such 'post hoc experiments' is critical to support exploratory and iterative data analysis. The ease, power, and quality of such after-the-fact analyses benefited dramatically from replacing a sequential event log file representation with a carefully designed database (Mostow *et al.* 2002c). For example, back when we used log files to analyze the effects of Reading Tutor assistance on words, we wrote perl scripts to parse the log files into an analyzable data set (Aist and Mostow 1998). The perl scripts were sufficiently ad hoc, complex, and bug-prone to deter such analyses in the first place, or to cast doubt on their validity (Mostow and Aist 2001). In contrast, a database representation not only facilitated analyses of assistance on words (Heiner, Beck and Mostow 2004, 2005), but of other tutorial interventions as well, such as inserting *wh-* questions to stimulate comprehension (Beck, Mostow and Bey 2004b).

Although naturalistic outcomes offer some advantages, they can pose complications, especially when trials overlap in time. One such complication is 'masking,' in which one trial affects the outcome of another (Mostow and Aist 2001). For example, suppose the randomized decision is what type of help to give on a word, such as *cat*. The outcome of the trial is the student's performance the next time the student attempts the same word *cat* in some later sentence. An instance of masking arises if, just before that next attempt, the tutor gives help again on the word *cat*, thereby substantially affecting the student's performance and overestimating the impact of the earlier help.

Another complication involves confounding the experimental manipulation with other influences on trial outcomes. For example, the student's performance may depend on how soon the trial ends, which is influenced in turn by how often the word occurs in English. Some types of word help (e.g. rhyming hints such as *rhymes with mat*) tend to apply to high-frequency words such as *cat*, which the reader is therefore likely to encounter again soon. Other types of help (e.g. segmenting into syllables) tend to apply to low-frequency words such as *catgut*, which the reader is likely to wait longer before re-encountering. If rhyming hints lead more often to success at the next encounter than syllable segmentation does, is it because that type of hints is really more effective, or because it applies to easier words, or because the reader has less time to forget the word before the re-encounter that defines the trial outcome? That is, trial outcomes may confound help type with word frequency (Heiner *et al.* 2004). Such complications are not necessarily insurmountable, but require careful statistical treatment to control for them.

4 Mine tutor data set

Once tutor data is in tabular form, mining it extracts useful information, e.g. by assessing student skills, evaluating interventions to scaffold those skills, or measuring effects on student learning. What tactics does such mining use?

4.1 Browse example interactions to generate hypotheses

Inspecting specific examples of tutorial interaction can help identify interesting phenomena, check conjectures, and spot bugs. We were able to develop a generic tool to support such browsing, thanks to logging tutorial interactions to a database that meets certain conventions (Mostow, Beck, Cen, Cuneo, Gouvea and Heiner 2005a; Mostow, Beck, Cuneo, Gouvea and Heiner 2005b). The tool lets us input a query to describe tutorial events of interest, see the hierarchical context in which each such event occurred, and drill down to explore it in further detail. For example, we used the tool to browse example cases where students started to read stories but then 'bailed out' without finishing them. Figure 1 displays one such case.

4.2 Aggregate a data set with respect to some feature

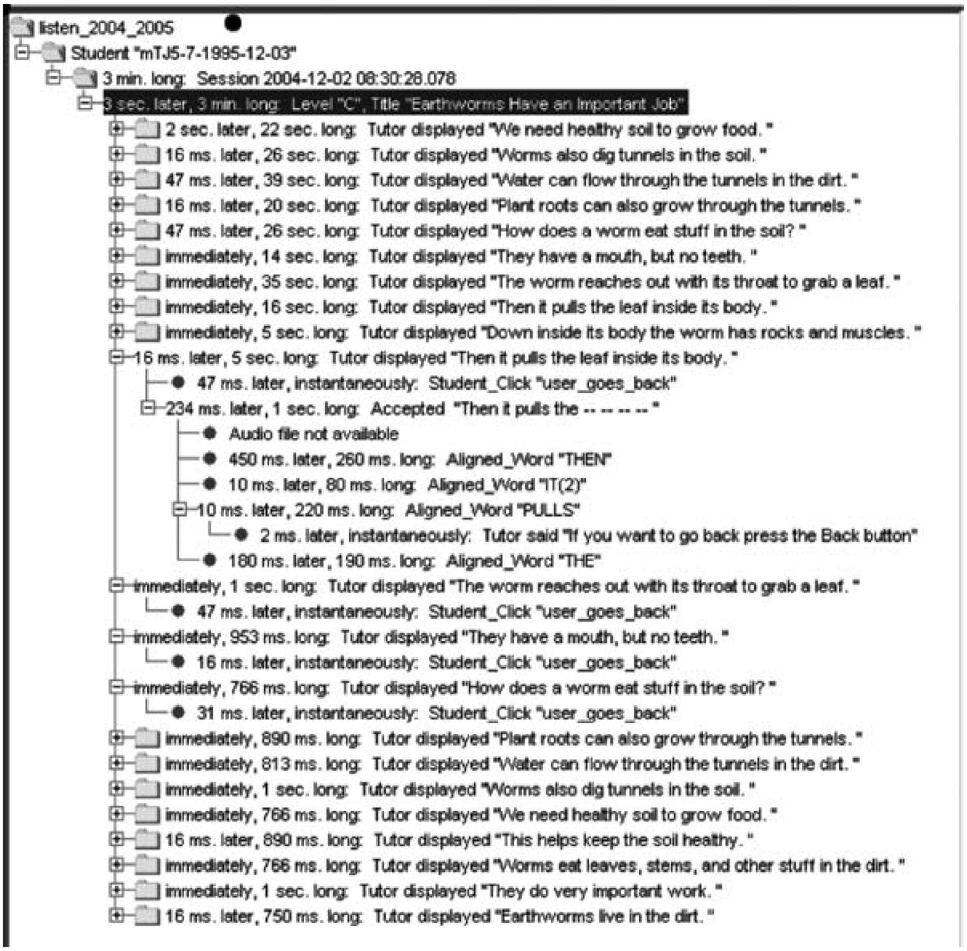
For example, one analysis (Mostow *et al.* 2002a) aggregated the total percentage of time that each student spent on different types of activities. The resulting time allocation correlated significantly with gains even after controlling for pretest scores. For instance, the percentage of time that students spent picking stories instead of reading them correlated negatively with their test score gains.

4.3 Fit a model to a data set by finding parameter values that minimize its prediction errors

For example, given a cognitive model of a task in the form of a set of production rules, 'knowledge tracing' (Corbett and Anderson 1995) estimates the probability that the student has learned a given rule by analyzing a trace of the student's problem-solving steps. Here aggregation consists of Bayesian updating at each step where the rule was appropriate, based on whether the student correctly applied the rule. Once trained, such a model can help the tutor pick what to teach next.

4.4 Train a model on a data set

Various methods such as statistical regression and decision tree learning induce a model to approximate a given data set. For example, one such model (Beck *et al.*



```
listen_2004_2005
  Student "mTJ5-7-1995-12-03"
    3 min. long: Session 2004-12-02 08:30:28.078
      3 sec. later, 3 min. long: Level "C", Title "Earthworms Have an Important Job"
        2 sec. later, 22 sec. long: Tutor displayed "We need healthy soil to grow food."
        16 ms. later, 26 sec. long: Tutor displayed "Worms also dig tunnels in the soil."
        47 ms. later, 39 sec. long: Tutor displayed "Water can flow through the tunnels in the dirt."
        16 ms. later, 20 sec. long: Tutor displayed "Plant roots can also grow through the tunnels."
        47 ms. later, 26 sec. long: Tutor displayed "How does a worm eat stuff in the soil?"
        immediately, 14 sec. long: Tutor displayed "They have a mouth, but no teeth."
        immediately, 35 sec. long: Tutor displayed "The worm reaches out with its throat to grab a leaf."
        immediately, 16 sec. long: Tutor displayed "Then it pulls the leaf inside its body."
        immediately, 5 sec. long: Tutor displayed "Down inside its body the worm has rocks and muscles."
        16 ms. later, 5 sec. long: Tutor displayed "Then it pulls the leaf inside its body."
          ● 47 ms. later, instantaneously: Student_Click "user_goes_back"
          234 ms. later, 1 sec. long: Accepted "Then it pulls the -- -- -- --"
            ● Audio file not available
            ● 450 ms. later, 260 ms. long: Aligned_Word "THEN"
            ● 10 ms. later, 80 ms. long: Aligned_Word "IT(2)"
            10 ms. later, 220 ms. long: Aligned_Word "PULLS"
              ● 2 ms. later, instantaneously: Tutor said "If you want to go back press the Back button"
            ● 180 ms. later, 190 ms. long: Aligned_Word "THE"
        immediately, 1 sec. long: Tutor displayed "The worm reaches out with its throat to grab a leaf."
          ● 47 ms. later, instantaneously: Student_Click "user_goes_back"
        immediately, 953 ms. long: Tutor displayed "They have a mouth, but no teeth."
          ● 16 ms. later, instantaneously: Student_Click "user_goes_back"
        immediately, 766 ms. long: Tutor displayed "How does a worm eat stuff in the soil?"
          ● 31 ms. later, instantaneously: Student_Click "user_goes_back"
        immediately, 890 ms. long: Tutor displayed "Plant roots can also grow through the tunnels."
        immediately, 813 ms. long: Tutor displayed "Water can flow through the tunnels in the dirt."
        immediately, 1 sec. long: Tutor displayed "Worms also dig tunnels in the soil."
        immediately, 766 ms. long: Tutor displayed "We need healthy soil to grow food."
        16 ms. later, 890 ms. long: Tutor displayed "This helps keep the soil healthy."
        immediately, 766 ms. long: Tutor displayed "Worms eat leaves, stems, and other stuff in the dirt."
        immediately, 1 sec. long: Tutor displayed "They do very important work."
        16 ms. later, 750 ms. long: Tutor displayed "Earthworms live in the dirt."
```

Fig. 1. Hierarchical context and partially expanded details of a selected event, from (Mostow *et al.* 2005a).

2004a) used data from automated speech recognition of children's oral reading to predict their actual reading fluency with correlation 0.9. Another such model (Mostow *et al.* 2004) used students' performance on automatically generated cloze questions to predict their scores on a standard comprehension test with correlation 0.8. Besides guiding tutor decisions, such estimates may be useful to teachers and to students. A set of association rules induced from data logged by a web-based logic tutor enabled it to predict mistakes that students were likely to make, and warn them first (Merceron and Yacef 2005) – a nice example of a tutorial intervention enabled by mining tutor data.

5 Examples

To illustrate the 'modify, map, mine' strategy described above, we summarize how it was applied in two experiments involving an automated reading tutor.

One experiment (Heiner *et al.* 2004) compared the efficacy of different types of tutorial assistance in decoding words. The key modification to the tutor, made years earlier (Mostow and Aist 1999, 2001), was to randomize the decision of what kind of assistance to give on a word. The tutor chose at random among plausible alternatives, such as pronouncing the word, giving a hint of the form *Rhymes with …* (if feasible), or sounding out its individual phonemes (but not if the word was more than four phonemes long, lest this type of assistance overload the student's short-term memory).

The key mapping step, made after the fact, was to formulate the resulting data in terms of trials. The context for each trial arose whenever the student was reading a story and either the student clicked on a word for help, or the tutor decided on its own initiative to give preemptive or corrective assistance on a word. For example, the student is reading the sentence *Laron likes to draw*. The student clicks on the word *draw*, and the tutor randomly chooses the hint *Rhymes with saw*. The outcome of the trial is a function of the ensuing tutorial dialog. The purpose of the study was to analyze how tutorial decisions affect student learning, so a natural outcome was defined by the student's subsequent encounter of the word *draw*, for example in '*I want to draw something, not color in something somebody else drew,' Jon whispered to himself*. One definition of success is whether the speech recognizer accepts this later encounter of *draw* as read fluently without tutorial assistance.

The mining step in this example consisted of aggregating 189,039 trials to compare the success rates of different forms of tutorial assistance on words – that is, the percentage of trials ending in success as defined above. Rhyming hints turned out to have the highest success rate. Slicing each student's recorded tutorial interaction into separate trials for each word exploits the assumption that assistance on a word affects the student's performance on subsequent encounters of the same word much more than it transfers to performance on other words.

A different experiment (Beck *et al.* 2004b) tested the effect of a general intervention to scaffold students' comprehension. Detecting learning effects would be problematic using a within-subject experiment design as in the word assistance experiment, because a general comprehension strategy should transfer to other text. However, such a design can still be used to detect scaffolding effects during the period where students can use the strategy when prompted to do so, but have not yet learned to apply it habitually on their own.

The comprehension scaffolding experiment used two types of modifications: adding randomized interventions, and probes to measure their effects. The intervention consisted of randomly inserting a generic multiple-choice *wh-* question designed to scaffold comprehension, based on previous findings (Roshenshine, Meister and Chapman 1996) that training students to ask *wh-* questions improved their comprehension. For example, after the story sentence (from one of Aesop's fables) '*Why not come and chat with me,' said the Grasshopper,'instead of toiling and moiling in that way,?*' the tutor might insert either the question *What has happened so far?*, or a different generic *wh-* question, or no question. The choices for each question were also generic. In this example, the choices were: *facts were given; a problem is being solved; a problem has been solved; a problem; an introduction; a* *mistake; a meeting; nothing yet; I don't know*. After answering the question (if any), the student resumed reading the story.

The probe tested comprehension by randomly choosing sentences in the story to make into multiple-choice cloze questions. The tutor deleted a word and prompted the student to choose among four story words to fill in the blank. Previous work (Mostow *et al.* 2004) showed that children's performance on such questions correlated strongly with measures of their reading comprehension. One such probe occurred after the sentence '*I am helping to lay up food for the winter,' said the Ant, 'and recommend you to do the same.*' The cloze prompt was '*Why bother about ______*'? The choices were *food; winter; dying; past*. After answering the question, the student resumed reading the story, starting with the unmodified sentence, in this case '*Why bother about winter*'? (which is dialog spoken by a character in the story as part of the text, not a question inserted by the tutor).

The resulting data stream can be mapped to trials by treating each randomized decision of whether to insert a *wh-* question as starting a trial, and the student's performance on the next cloze question as the outcome of that trial. However, multiple decisions likely affected the same trial, and the same decision likely affected multiple outcomes. So instead, it makes sense to analyze the overall impact on each cloze question of *all* the preceding *wh-* questions.

Accordingly, an appropriate way to mine this data was to construct a logistic regression model to predict performance on the 15,187 cloze questions. To control for individual differences between students and account for statistical dependence among each student's performance on different questions (Menard 1995), the model included student identity as a factor. The model included the number of preceding *wh-* and cloze questions as variables to determine their effects on comprehension. Initial analysis revealed that performance was lower on cloze questions administered too soon after the preceding question, presumably because they annoyed students, who then answered randomly. The model therefore also included variables for the number of 'recent' *wh-* and cloze questions, i.e., asked in the preceding two minutes, as well as a variable for the amount of time since the last question of any kind.

The logistic regression showed that the total number of *wh-* questions was a significant positive predictor of cloze performance ($p = .023$), and the number of 'recent' *wh-* questions was a negative predictor ($p = 0.074$), as was the recency of the last question ($p = 0.036$), but the number of cloze questions (total or recent) was not a significant predictor. These results indicate that the *wh-* questions were effective in scaffolding children's comprehension.

6 Contributions and future work

We have decomposed the goal of automated educational discovery into *modifying* the data that tutors log, *mapping* that data from event streams into tabular form, and *mining* the data to extract information useful for assessing student skills and evaluating tutorial actions. We have identified tactics for each of these phases and illustrated their use. This approach transforms year-long, fine-grained streams of daily, mixed-initiative tutorial interactions with several hundred students into hundreds of thousands of within-subject experimental trials. The resulting 'big data' offers the statistical power needed to discover which tutorial actions help which students in which cases. Future work is needed to refine the model, identify additional tactics, and automate them. An intelligent tutor that explores alternative strategies fully autonomously in order to discover what works best may lie indefinitely far off. But intelligent tutors can already accrue – and pool – more one-on-one interaction with students than any human tutor can accumulate in a lifetime. The intelligent tutor with a million hours of experience may not be far off at all. Our challenge is to mine useful educational discoveries from such experience.

Acknowledgements

This work was supported in part by the National Science Foundation under ITR/IERI Grant No. REC-0326153. Any opinions, findings, conclusions, or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation or the official policies, either expressed or implied, of the sponsors or of the United States Government. We thank the educators and students who contributed to our data, and our Project LISTEN colleagues, past and present, who contributed to the work described here.

References

Aist, G. (2001) Towards automatic glossarization: Automatically constructing and administering vocabulary assistance factoids and multiple-choice assessment. *International Journal of Artificial Intelligence in Education* **12**: 212–231.

Aist, G. and Mostow, J. (1998) Estimating the effectiveness of conversational behaviors in a reading tutor that listens. *Working Notes of the AAAI Spring Symposium on Applying Machine Learning to Discourse Processing*, Stanford, CA.

Aleven, V., Popescu, O. and Koedinger, K. (2002) Pilot-testing a tutorial dialogue system that supports self-explanation. *6th International Conference on Intelligent Tutoring Systems*, pp. 344–354. Biarritz, France.

Arroyo, I., Beck, J. E., Woolf, B. P., Beal, C. R. and Schultz, K. (2000) Macroadapting AnimalWatch to gender and cognitive differences with respect to hint interactivity and symbolism. *5th International Conference on Intelligent Tutoring Systems (ITS2000)*, pp. 574–583. Montreal, Canada.

Beck, J. E. (ed.) (2004) *Proceedings of the ITS2004 Workshop on Analyzing Student-Tutor Interaction Logs to Improve Educational Outcomes*. Maceio, Brazil.

Beck, J. E., Jia, P. and Mostow, J. (2004a) Automatically assessing oral reading fluency in a computer tutor that listens. *Technology, Instruction, Cognition and Learning* **2**: 61–81.

Beck, J. E., Mostow, J. and Bey, J. (2004b) Can automated questions scaffold children's reading comprehension? *Proceedings of the 7th International Conference on Intelligent Tutoring Systems*, pp. 478–490. Maceio, Brazil.

Beck, J. E., Woolf, B. P. and Beal, C. R. (2000) ADVISOR: A machine learning architecture for intelligent tutor construction. *Proceedings of the Seventeenth National Conference on Artificial Intelligence*, pp. 552–557. Austin, Texas.

Calfee, R., Kukich, K., Landauer, T., Laham, D., Foltz, P., Hirschman, L., Breck, E., Burger, J. and Ferro, L. (2000) The debate on automated essay grading. *IEEE Intelligent Systems* **15**(5): 22–37.

Corbett, A. and Anderson, J. (1995) Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction* **4**: 253–278.

Corbett, A., McLaughlin, M. S. and Scarpinatto, K. C. (2000) Modeling student knowledge: Cognitive tutors in high school and college. *User modeling and user-adapted interaction* **10**: 81–108.

Goodman, B., Hitzeman, J., Linton, F. and Ross, H. (2003) Towards intelligent agents for collaborative learning: Recognizing the roles of dialogue participants. *9th International Conference on User Modeling*, pp. 363–367. Johnstown, PA.

Heiner, C., Beck, J. E. and Mostow, J. (2004) Improving the help selection policy in a Reading Tutor that listens. *Proceedings of the InSTIL/ICALL Symposium on NLP and Speech Technologies in Advanced Language Learning Systems*, pp. 195–198. Venice, Italy.

Heiner, C., Beck, J. E. and Mostow, J. (2005) When do students interrupt help? Effects of time, help type, and individual differences. *Proceedings of the 12th International Conference on Artificial Intelligence in Education (AIED 2005)*, pp. 819–826. Amsterdam, The Netherlands.

Koedinger, K. R. and Anderson, J. R. (1993) Reifying implicit planning in geometry: Guidelines for model-based intelligent tutoring system design. In: S. P. Lajoie and S. J. Derry (eds.), *Computers as Cognitive Tools*, pp. 15–45. Hillsdale, NJ: Erlbaum.

Menard, S. (1995) Applied Logistic Regression Analysis. *Quantitative Applications in the Social Sciences 106.*

Merceron, A. and Yacef, K. (2005) Educational data mining: a case study. In: C.-K. Looi, G. McCalla, B. Bredeweg and J. Breuker (eds.), *Artificial Intelligence in Education: Supporting Learning through Intelligent and Socially Informed Technology (Proceedings of the 12th International Conference on Artificial Intelligence)*, pp. 467–474. Amsterdam: IOS Press.

Mostow, J. (in press) Evaluation purposes, excuses, and methods: Experience from a Reading Tutor that listens. In: C. K. Kinzer and L. Verhoeven (eds.), *Interactive Literacy Education*, Mahway, NJ: Erlbaum.

Mostow, J. and Aist, G. (1997) The sounds of silence: Towards automated evaluation of student learning in a Reading Tutor that listens. *Proceedings of the Fourteenth National Conference on Artificial Intelligence* (AAAI-97), pp. 355–361. Providence, RI.

Mostow, J. and Aist, G. (1999) Giving help and praise in a reading tutor with imperfect listening – because automated speech recognition means never being able to say you're certain. *CALICO Journal* **16**(3): 407–424.

Mostow, J. and Aist, G. (2001) Evaluating tutors that listen: An overview of Project LISTEN. In: K. Forbus and P. Feltovich (eds.), *Smart Machines in Education*, pp. 169–234. Menlo Park, CA: MIT/AAAI Press.

Mostow, J., Aist, G., Beck, J., Chalasani, R., Cuneo, A., Jia, P. and Kadaru, K. (2002a) A la recherche du temps perdu, or as time goes by: Where does the time go in a Reading Tutor that listens? *Proceedings of the Sixth International Conference on Intelligent Tutoring Systems* (ITS2002), pp. 320–329. Biarritz, Franc.

Mostow, J., Aist, G., Bey, J., Burkhead, P., Cuneo, A., Junker, B., Rossbach, S., Tobin, B., Valeri, J. and Wilson, S. (2002b) Independent practice versus computer-guided oral reading: Equal-time comparison of sustained silent reading to an automated reading tutor that listens. *Ninth Annual Meeting of the Society for the Scientific Study of Reading*, Chicago, IL.

Mostow, J., Beck, J., Chalasani, R., Cuneo, A. and Jia, P. (2002c) Viewing and analyzing multimodal human-computer tutorial dialogue: a database approach. *Proceedings of the Fourth IEEE International Conference on Multimodal Interfaces (ICMI 2002)*, Pittsburgh, PA, pp. 129–134.

Mostow, J., Aist, G., Burkhead, P., Corbett, A., Cuneo, A., Eitelman, S., Huang, C., Junker, B., Sklar, M. B. and Tobin, B. (2003) Evaluation of an automated Reading Tutor that listens: Comparison to human tutoring and classroom instruction. *Journal of Educational Computing Research* **29**(1): 61–117.

Mostow, J., Beck, J., Bey, J., Cuneo, A., Sison, J., Tobin, B. and Valeri, J. (2004) Using automated questions to assess reading comprehension, vocabulary, and effects of tutorial interventions. *Technology, Instruction, Cognition and Learning* **2**: 97–134.

Mostow, J. and Beck, J. (2005) Micro-analysis of fluency gains in a Reading Tutor that listens: Wide vs. repeated guided oral reading, *Twelfth Annual Meeting of the Society for the Scientific Study of Reading*. Toronto.

Mostow, J., Beck, J., Cen, H., Cuneo, A., Gouvea, E. and Heiner, C. (2005a) An educational data mining tool to browse tutor-student interactions: Time will tell! *Proceedings of the Workshop on Educational Data Mining, National Conference on Artificial Intelligence*, pp. 15–22. Pittsburgh, PA.

Mostow, J., Beck, J., Cuneo, A., Gouvea, E. and Heiner, C. (2005b) A generic tool to browse tutor-student interactions: Time will tell! *Proceedings of the 12th International Conference on Artificial Intelligence in Education (AIED 2005)*, pp. 884–886. Amsterdam, The Netherlands.

Mostow, J., Aist, G., Huang, C., Junker, B., Kennedy, R., Lan, H., Latimer, D., O'Connor, R., Tassone, R., Tobin, B. and Wierman, A. (in press) 4-Month evaluation of a learner-controlled Reading Tutor that listens. In: V. M. Holland and F. N. Fisher (eds.), *Speech Technology for Language Learning*, Lisse, The Netherlands: Swets & Zeitlinger.

Poulsen, R. (2004) *Tutoring Bilingual Students With an Automated Reading Tutor That Listens: Results of a Two-Month Pilot Study*. Unpublished Masters Thesis, DePaul University, Chicago, IL.

Roshenshine, B., Meister, C. and Chapman, S. (1996) Teaching students to generate questions: A review of the intervention studies. *Review of Educational Research* **66**(2): 181–221.

Self, J. (1988) Bypassing the intractable problem of student modelling. *Intelligent Tutoring Systems*, pp. 18–24.

Shute, V., Gawlick-Grendell, L. A., Young, R. K. and Burnham, C. A. (1996) An experiential system for learning probability: Stat Lady description and evaluation. *Instructional Science* **24**(1): 25–46.

Soller, A. L. (2001) Supporting social interaction in an intelligent collaborative learning system. *International Journal of Artificial Intelligence in Education* **12**(1): 40–62.

Soller, A. (2004a) Computational modeling and analysis of knowledge sharing in collaborative distance learning. *User Modeling and User-Adapted Interaction* **14**(1): 351–381.

Soller, A. (2004b) Understanding knowledge-sharing breakdowns: a meeting of the quantitative and qualitative minds. *Journal of Computer Assisted Learning* **20**(3): 213–223.

Vassileva, J., Greer, J., McCalla, G., Deters, R., Zapata, D., Mudgal, C. and Grant, S. (1999) A multi-agent approach to the design of peer-help environments. *International Conference on Artificial Intelligence in Education*, pp. 38–45. Le Mans, France.

Weiser, M. (1984) Program slicing. *IEEE Transactions on Software Engineering* **10**(4): 352–357.