Using a student model to improve a computer tutor's speech recognition

Joseph E. Beck[1], Kai-min Chang[2], Jack Mostow[3], and Albert Corbett[4]
[1]Center for Automated Learning and Discovery
[2]Language Technologies Institute
[3]Robotics Institute
[4]Human Computer Interaction Institute
School of Computer Science
Carnegie Mellon University
joseph.beck@cmu.edu

Abstract. Intelligent computer tutors can derive much of their power from having a student model that describes the learner's competencies. However, constructing a student model is challenging for computer tutors that use automated speech recognition (ASR) as input. This paper reports using ASR output from a computer tutor for reading to compare two models of how students learn to read words: a model that assumes students learn words as whole-unit chunks, and a model that assumes students learn the individual letter→sound mappings that make up words. We use the data collected by the ASR to show that a model of letter→sound mappings better describes student performance. We then compare using the student model and the ASR, both alone and in combination, to predict which words the student will read correctly, as scored by a human transcriber. Surprisingly, majority class has a higher classification accuracy than the ASR. However, we demonstrate that the ASR output still has useful information, and that classification accuracy is not a good metric for this task, and the Area Under Curve (AUC) of ROC curves is a superior scoring method. The AUC of the student model is statistically reliably better (0.670 vs. 0.550) than that of the ASR, which in turn is reliably better than majority class. These results show that ASR can be used to compare theories of how students learn to read words, and modeling individual learner's proficiencies may enable improved speech recognition.

1 Motivation and Introduction

Intelligent Tutoring Systems (ITS) derive much of their power from having a student model [1] that describes the learner's proficiencies at various aspects of the domain to be learned. For example, the student model can be used to determine what feedback to give [2] or to have the students practice a particular skill until it is mastered [3] Unfortunately, language tutors have difficulty in developing strong models of the student. Much of the difficulty comes from the inaccuracies inherent in automated speech recognition (ASR). Providing explicit feedback based only on student performance on one attempt at reading a word is not viable since the accuracy at distinguishing correct from incorrect reading is not high enough [4]. Due to such problems, student modeling has not received as much attention in computer assisted language learning systems as in classic ITS [5], although there are exceptions such as [6].

A common approach to developing cognitive models for use in an ITS is to use think-aloud protocols [7, 8]. In a think-aloud study [7], participants verbalize their thinking while solving a problem. Such verbalizations are then used to construct a cognitive model of how the participants were solving the task. This approach has also been used to develop cognitive models for ITS [8]. Unfortunately, due to the speed of the reading process, think-aloud methodology is not well suited to modeling reading.

There have been efforts to develop cognitive models that describe the reading process. For example, [9] developed a parallel distributed processing model that was able to simulate many aspects of human performance. A major drawback of this approach is the models are designed for individual word reading and not for reading connected text. Furthermore, rather than observing the reader's behavior with each word to model this particular reader, these studies use simulated input to try to mimic known human behavioral characteristics.

The goal of this paper is to first quickly compare two models of how children learn to read, and then to use the better model to improve the ability of the ASR to listen accurately to children.

We first describe our approach to collecting and representing our data, and describe two candidate models of children's reading. We then compare which model better fits student performance as scored by the ASR. Finally to determine whether the student model can improve listening accuracy, we compare the effects of combining the student model and the ASR to better predict how a human transcriber judges words as read correctly or incorrectly.

2 Approach to Constructing the Student Model

In this Section we discuss the data used for experiments, our statistical framework for modeling, and the two models of reading we are investigating.

2.1 Data collected and representation

We collected data from 541 students working with a computer tutor that helps children learn how to read. Over the course of the school year, these students read approximately 4.1 million words (as heard by the ASR). The tutor presented one sentence (or fragment) at a time, and asked the student to read it aloud. The student's speech was segmented into utterances that ended when the student stopped speaking. Each utterance was processed by the ASR and aligned against the sentence. This alignment scores each word of the sentence as either being accepted (heard by the ASR as read correctly), rejected (the ASR heard and aligned some other word), or skipped (not read by the student). We use the terms "accepted" and "rejected" rather than "correct" and "incorrect" due to inaccuracies in the ASR. The ASR only notices about 25% of student misreadings, and scores as incorrectly read about 4% of words that were read correctly. Therefore "accept" and "reject" are more accurate terms.

One problem is determining how to score each word in the sentence text. As an example, suppose the student is trying to read the sentence "They are formed over millions of years and once depleted will take millions of years to replenish," and misreads "depleted," and stops reading after "will." Clearly the word "depleted" was read incorrectly, but what about the words "take" through "replenish?" It is odd to score these words as incorrect, since the student did not try to read them. However, the student stopped reading the sentence for some reason. Since his true reason for stopping is unknown, we assume the student had difficulty with the next word in the sentence where he stopped reading. So in the above example, the student would be considered to have misread "take."

Our heuristic for scoring the sentence words was:

1. For each utterance
    a. Start = position of first accepted word
    b. End = 1+position of last accepted word
    c. Use the ASR's accept/reject decision to score all words from Start through End as correctly or incorrectly read.
    d. Even if the ASR accepted a word, if the student hesitated more than 300ms, score that word as incorrect.
2. For each sentence word *w*
    a. Find the first utterance where *w's* position is between Start and End
    b. Use the ASR's score for *w* from that utterance. If nothing is aligned against *w*, score it as incorrectly read.

c. If a student requested help on *w* before it was accepted by the ASR, mark it as incorrectly read.
d. If *w* is not contained within any utterance, then it is not scored since the student did not attempt to read the word.

To continue the above example, if the student's second attempt at reading the sentence consisted of "take millions of years to replenish," then all of the sentence words would be accepted as read correctly except for "depleted" (since it was misread) and "take" (since in the first utterance that contains this sentence word nothing was aligned against the sentence word).

After using this methodology to combine utterances, and removing students who were not part of the official study, we were left with 360 students and 1.95 million sentence words that students attempted to read. On average, students used the tutor for 8.5 hours. Most students were between six and eight years old, and had reading skills appropriate for their age.

2.2 Knowledge tracing

Now that we have determined how to score student attempts at reading a word as correct or incorrect, we must map those overt actions to some internal representation of the student's knowledge. Prior work in this area [10] has shown that knowledge tracing [3] is an effective approach for using ASR output to model students.

The goal of knowledge tracing is to map observable student actions while performing a skill (whether the student's response is correct or incorrect) to internal knowledge states (whether the student knows the skill or not). As illustrated in Figure 1, knowledge tracing maintains four constant parameters for each skill. Two parameters, L0 and t, are called learning parameters and refer to the students initial knowledge and to the probability of learning a skill given an opportunity to apply it, respectively. Two parameters, slip and guess, are called performance parameters and used to account for student performance not being a perfect reflection of underlying knowledge. The guess parameter is the probability that a student who has not mastered the skill can generate a correct response. For example, on a multiple-choice test with four response choices, a student with no knowledge still has a 25% chance of getting the question correct. The slip parameter is used to account for even knowledgeable students making an occasional mistake. For example, a student who when asked to multiply 4 and 3, could accidentally hit the keys in the wrong order and type "21."

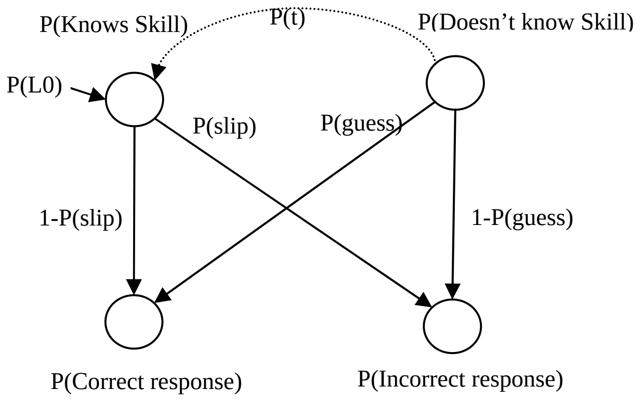

Figure 1. Overview of knowledge tracing

For each student and for each skill, knowledge tracing is maintains the probability that the student knows the skill. Knowledge tracing updates its estimates of P(knows) based on student performance. The approach is that whenever a student has an opportunity to apply a skill, observe whether the student performed the skill correctly or incorrectly.

The probability of P(knows) can then be computed via Bayes's rule [3]. In addition, the transition probability accounts for the expected increase in student knowledge due to the opportunity to practice the skill.

Knowledge tracing distinguishes between a student knowing a skill and getting it correct. P(knows skill) refers to the model's estimate of the student's internal knowledge. P(correct response) is derived from P(knows skill) and the performance parameters: P(correct response) = P(knows skill) * 1 – P(slip) + 1 – P(knows skill) * P(guess). Prior work on applying knowledge tracing to ASR output [10] demonstrated that the slip and guess parameters, in addition to accounting for variability in student performance, can also account for variability in the ASR scoring of student responses. Therefore, knowledge tracing is an appropriate modeling framework for this task.

2.3 Lexical and graphophonemic models

We considered two possible models for how students could learn to decode words. The first is a lexical model, which assumes that students learn words as a whole-unit, and there is no transfer between words. Although the assumed lack of transfer is somewhat naïve, it is likely that skilled readers recognize most words by sight. It is less clear, however, whether children learning to read have a similar representation as skilled readers.

The second model is a graphophonemic model, and assumes that rather than learning whole-words, students instead learn subword units. Specifically, it assumes that students learn the grapheme (letter) to phoneme (sound) mappings that make up words. For example, the word "chemist" contains the following grapheme→phoneme mappings: ch→/K/, e→/EH/, m→/M/, i→/IH/, s→/S/, and t→/T/. The grapheme→phoneme model is abbreviated as g→p model.

Given these two possible models, the next task is to determine which model is better described by our data under the knowledge tracing framework.

2.4 Evaluating the lexical and g→p models

To determine which model of student reading, lexical or g→p, better described student performance, we fit each of them to the student performance data as heard by the ASR (described above). First we split the students into two groups (to create a testing set to be used later). For students in the training set, we ordered each student's performance data chronologically. Then, for each model, we estimated the knowledge tracing parameters for each skill based on the student performance data.

For the lexical model, we simply treated words as skills. So each student attempt at reading a word was evidence for knowing the whole word or not. For the g→p model, we considered all of the g→p mappings in the word. If the word was accepted as correct, then all of the mappings were credited; if it was rejected as incorrect then all of the mappings were debited.

The lexical model had considerably more skills than the g→p model. There were 3210 lexical skill (i.e. words); in comparison there were only 295 g→p mappings encountered by students. As a result of this difference in number of skills, the g→p model had substantially more students encountering each skill on average (106 vs. 45). Table 1 describes the knowledge tracing parameter estimates for each of the models. These parameters are the average across each skill in the model, weighted by the number of times the skill occurred. This weighting is to avoid biasing the model by several skills that occur rarely (e.g. the word "arose" or "bts→/ts/" as in the word "debts").

Note that the performance parameters (guess and slip) are similar for both models, while the learning parameters (L0 and T) are different. These performance parameters are vastly different than in knowledge tracing done in other ITS (where typically "guess" is restricted to be less than 0.3). The reason for this difference is the uncertainty introduced by the ASR scoring. This uncertainty is the reason the performance parameters under both models are similar: the parameters are (mostly) modeling the speech recognition rather than the student. Thus, the agreement in parameter estimates between the two models is not surprising. The column labeled R^2 in the table refers to how well the knowledge tracing parameters fit each skill.

Table 1. Mean knowledge tracing parameters

	L0	T	Guess	Slip	R^2
Lexical	0.32	0.14	0.65	0.08	0.34
g→p	0.49	0.01	0.57	0.10	0.48

At least within the framework of knowledge tracing, student performance is better described by the g→p model (R^2 of 0.48) than by the lexical model (R^2 of 0.34). Thus, the g→p model appears to be a better description of how children at this age acquire reading skills.

3 Leveraging the Student Model to Improve Speech Recognition

Although the g→p model is a useful way of viewing student performance and it provides a reasonable *description* of how students learn how to read, we would like to use the g→p student model to make *predictions* about how students will behave in order to improve the speech recognition system. For example, if the student model believes that the student has mastered the g→p mappings of the word "cat," but the ASR believes the student misread the word, perhaps we should ignore the ASR output and instead credit the student with reading the word correctly.

To evaluate possible improvements to the ASR, we had a skilled human transcribe a sample of the student utterances throughout the year. We followed the same protocol for aligning the sentence text against the transcription as we did for the ASR output, and similarly computed regions of the sentence we thought the student was attempting to read and counted the student's first attempt at reading the word. There were two differences from the prior procedure.

First, we excluded cases where the student requested help on the sentence word before reading it. Since the goal of this experiment was to evaluate whether a student model could improve ASR performance, help was a confound since it could be detected by neither the ASR nor the transcriber. Therefore, we simply excluded such trials.

Second, we insisted that the ASR and transcriber agree about which word the student was trying to read. Sometimes the ASR would be confused by background noise and get off-track. We were not trying to improve performance in these cases, so simply excluded them from the data.

Our approach was to treat the problem as classification. We used the second, testing, half of our data, so these data are not the ones used to perform the knowledge tracing parameter estimates of L0, t, slip, or guess. For the students in the testing set, we ran their data for the year through the knowledge tracing equations to determine skill estimates for each student for each g→p mapping. While tracing through a student's performance for the year, if a particular word had been transcribed, we recorded: 1) the student's knowledge at that point in time (before updating the knowledge tracing estimates of the student's knowledge for this attempt), 2) the ASR's accept/reject decision, and 3) whether the transcriber thought the student said the word correctly. The transcriber's scoring was the outcome variable for the classifier. We considered several types of features for the classifier:

1. The relative difficulty of the word for the student. We pretested students at the beginning of year on a variety of tests, including the Woodcock Reading Mastery's [11] Word Identification subtest, which gives a student's proficiency at reading words in grade equivalent terms (e.g. 3.2 means second month of the 3rd grade). We also had a heuristic that estimates the difficulty of the word on the same grade equivalent scale. The difference between these scores is the relative difficulty of the word for this student.
2. The student's proficiencies at the g→p mappings in the word. Since words have a variable number of mappings, we needed some way to get a constant number of features per word. We settled on extracting the student's proficiency on the following g→p mappings in the word: the first, the last, the one with the lowest proficiency, and the one with the highest proficiency. We also computed the student's mean proficiency across all the g→p mappings in the word, and product of P(knows) for a all the mappings in the word. In addition to computing those six features for P(knows), we also computed them for P(correct) (according to the guess and slip parameter estimates for the skill).
3. The ASR's accept/reject decision for this word.

The testing set contained nearly 1 million sentence words heard by the ASR. However, only 8,818 of those words were transcribed. Furthermore, these data were highly imbalanced, with 369 (4.2%) instances of students misreading a word and 8449 (95.8%) instances of correct reading. Although it may seem unusual for students to read 95.8% of words correctly (not counting those words on which they requested help), this level of performance is appropriate for material to help children learn to read.

For input to the classifier, we used several combinations of the above three groups of features: relative word difficulty, knowledge tracing features, the ASR, ASR + relative word difficulty, and ASR + relative word difficulty + knowledge tracing features.

The relative word difficulty is in essence a simple student model: how hard is this word for a student of this general reading proficiency; the knowledge tracing model is a more nuanced view since it accounts for variations in the student's knowledge. Thus, we can compare the relative benefit of using different levels of knowledge about the student.

Table 2 shows the results of the classification procedure. All results were generated using Weka's [12] REPTree fast decision tree learner's default settings, with bagging (10 bags) and a 20-fold cross validation.

The five most salient items from Table 2 are:

1. No approach did noticeably better than baseline (maximum difference 1.85%).
2. Majority class outperformed the ASR.
3. The only classifiers that beat majority class used the knowledge tracing features as inputs.
4. The student model was not able to improve classification accuracy by much. In fact, the best performer only classified 6 more cases correctly than the majority classifier.
5. Having the ASR as a feature hurt performance.

However, perhaps classification accuracy is not the best metric to use. Even though majority class outperforms the ASR, would it really be a superior scoring system to always assume the student read the word correctly? As a thought experiment, pretend that we had simply scored all student reading as being correct rather than using the ASR at all. It would have been impossible to apply knowledge tracing (or other student modeling approaches). Therefore, we would never have been able to get the small improvement in classification accuracy over majority class that we obtained by adding the knowledge tracing estimates. Although the improvement is very slight, it does exist. Given that the student model was built from the ASR output, it must contain some signal that is being overlooked by a simple majority classifier.

Table 2. Classifier accuracy for predicting transcription

Features	Classifier accuracy
Knowledge tracing	95.88%
ASR + relative difficulty + knowledge tracing	95.84%
ASR + knowledge tracing	95.84%
Majority class	95.82%
Relative difficulty	95.79%
ASR + relative difficulty	95.78%
ASR (baseline)	94.03%

Furthermore, the primary goal of the ASR in a computer tutor is not to get high classification accuracy, it is to serve as a means to construct a student model to enable the tutor to select appropriate feedback and customize instruction for the student. It is unclear how assuming that the student is always correct can accomplish these modeling or teaching goals.

Perhaps lower classification accuracy is better if it enables the tutor to better model the student? One method of accomplishing this goal is to give different penalties to different types of classification mistakes. Unfortunately, it is difficult to specify *a priori* a good evaluation function that would lead to a good student model. For example, we could try different penalties for different types of classification mistakes, then compute how well we can model the student, and iterate. However, this approach is not computationally efficient. Furthermore, our ability to model the student depends on the domain being taught, known models for how students acquire the skills, etc. Therefore, it would be difficult to transfer research results tuned for one system to others.

One approach that sidesteps the problems of non-generalizable results and inventing penalties for various classification mistakes is to examine the Receiver Operator Characteristic (ROC) curves of the classifiers [13, p. 361]. Specifically, we investigate the Area Under Curve (AUC) of the ROC curves. AUC is a measure of the classifier sensitivity: how well does the classifier do at distinguishing instances of each target class.

Classifiers with a higher AUC are better than those with a lower AUC. A random (or majority) classifier will have an AUC of 0.5.

Table 3 shows the AUC for the classifiers shown in Table 2. The lower and upper bounds for the AUC were computed via SPSS's 95% confidence intervals making no parametric assumptions.

All of the AUCs are reliably superior to 0.5 (i.e. better than majority class). Therefore each set of features is able to distinguish the difference in likelihoods of the student making a mistake under different circumstances. Interestingly, all of the student models, even the simple model of relative word difficulty, were reliably superior to just using the ASR. The more complicated knowledge tracing model did not outperform the simpler model that just used word difficulty. However, there may be some slight gain from combining them with ASR (0.686 vs. 0.678). Therefore, it seems likely that both the simple and more complicated student model contain some independent information about the student's chances of reading a word correctly.

4 Contributions

This work extends prior work on testing models of reading in several ways. First, it applies the models to individual's data rather than to aggregate performance (as in [9]). Second, it examines students learning to read *in vivo* in the classroom rather than using simulated data (as in [9, 14]).

This work extends prior work on using ASR output to build student models (e.g. [10]). First, it considers using the student model to aid speech recognition. Also, rather than simply assuming a model of how children acquire reading skills, this paper examines the ability of the ASR to help select competing cognitive models reading (lexical and g→p models).

Table 3. ROC for various feature combinations

Features	AUC	Lower bound	Upper bound
ASR + knowledge tracing + difficulty	0.686	0.656	0.716
ASR + knowledge tracing	0.678	0.649	0.708
ASR + difficulty	0.678	0.648	0.707
Just difficulty	0.670	0.641	0.699
Just knowledge tracing	0.670	0.640	0.700
Just ASR	0.550	0.518	0.583

Compared to existing work on user modeling for (generally dialog) systems that use ASR (e.g. [15] and [16]) this work describes a richer model of the user. Two advantage of an ITS over many other systems that use ASR as input are that users work with the system for an extended length of time (8.5 hours in our study), and the system has a better idea of what the user is trying to do. Both of these features make for stronger user models.

5 Conclusions and Future Work

The ASR of a computer tutor for reading provides information about an individual student's reading development. The content of this information is sufficient to choose which of two possible models of reading development better describes the students using the tutor. Specifically, children learning to read are better modeled using subword properties (grapheme→phoneme mappings) than by treating words as atomic units.

The ASR is also powerful enough to construct a student model based on the student's past actions that can predict how the student will perform next—even when judged by a human transcriber.

Determining what constitutes good ASR performance in an ITS is complex, and classification accuracy can be misleading. Instead, AUC is a better metric for the actual task of the ASR: to provide a signal to customize instruction to the student. Using AUC as an outcome measure, the student model was able to improve the ability to hear the student's reading.

The method for constructing a student model from the ASR output is somewhat crude. Two areas of improvement are a better credit model and using cues other than acceptance/rejection of a word. Currently, all of the g→p mappings in a word are blamed or credited. However, if a student misreads a word it is probable that not all of the mappings are responsible. A Bayesian credit assignment approach (e.g. [2]) would overcome this weakness. Similarly, the student's pattern of hesitation before a word contains a useful signal for modeling the student [17]. One possible avenue is to use the amount of hesitation before reading a word as a clue to the strategy the student is using: a short pause suggests a lexical strategy while a longer pause suggests the student is using his knowledge of g→p mappings.

One open question is rather than using the ASR to compare two competing models of reading, is to instead ask whether the ASR be used to determine for *which words* and for *which students* a particular model is appropriate. For example, it is likely as students become more familiar with a word they will treat it as an atomic unit (as in the lexical model), and rely on their knowledge of grapheme→phoneme mappings for less familiar words. In the future, we would like to study the ASR's capacity to detect such transitions.

Finally, this paper does not resolve the best method for combining the information contained in the student model (historical, averaged, data) and the ASR (current, noisy, data). For example, we demonstrated that for a new utterance, the ASR does not do as good a job at determining which words the student read correctly than the student model—even though the student model does not use any information from the current attempt! An obvious conclusion is to use the student model to second guess the ASR for the current interaction. Less obvious is how the student model should be updated. Should the student's estimates be decreased (according to the ASR's scoring) or increased (according to the student model's scoring)? If the latter option is chosen, there is a positive feedback mechanism built into the student model which could lead to instability: once the student begins to demonstrate knowledge (or lack of knowledge), his scores will have a built-in tendency to further increase (decrease). Intuitively, this mechanism does not sound like a good one. Perhaps it is necessary to decouple the scoring of the student's responses with one set of rules for determining feedback (student model + ASR), but just using the ASR to update the student model?

Acknowledgements
This work was supported by the National Science Foundation, under ITR/IERI Grant No. REC-0326153. Any opinions, findings, and 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. The author also thanks John Helman for transcribing many, many, student utterances.

References

1. Woolf, B.P., *AI in Education*, in *Encyclopedia of Artificial Intelligence*. 1992, John Wiley &Sons: New York. p. 434-444.
2. Conati, C., A. Gertner, and K. VanLehn, *Using Bayesian Networks to Manage Uncertainty in Student Modeling.* User Modeling and User-Adapted Interaction, 2002. **12**(4): p. 371-417.
3. Corbett, A.T. and J.R. Anderson, *Knowledge tracing: Modeling the acquisition of procedural knowledge.* User Modeling and User-Adapted Interaction, 1995. **4**: p. 253-278.
4. Williams, S.M., D. Nix, and P. Fairweather. *Using Speech Recognition Technology to Enhance Literacy Instruction for Emerging Readers*. in *Fourth International Conference of the Learning Sciences*. 2000. p. 115-120: Erlbaum.
5. Heift, T. and M. Schulze, *Student Modeling and ab initio Language Learning.* System, the International Journal of Educational Technology and Language Learning Systems, 2003. **31**(4): p. 519-535.
6. Michaud, L.N., K.F. McCoy, and L.A. Stark. *Modeling the Acquisition of English: an Intelligent CALL Approach".* in *Eighth International Conference on User Modeling*. 2001. p.: Springer-Verlag.

7. Newell, A. and H. Simon, *Human Problem Solving*. 1972, Englewood Cliffs, N.J.: Prentice-Hall.
8. Anderson, J.R., *Rules of the Mind*. 1993: Lawrence Erlbaum Assoc.
9. Harm, M.W., B.D. McCandliss, and M.S. Seidenberg, *Modeling the successes and failures of interventions for disabled readers.* Scientific Studies of Reading, 2003. **7**(2): p. 155-182.
10. Beck, J.E. and J. Sison. *Using knowledge tracing to measure student reading proficiencies*. in *Proceedings of International Conference on Intelligent Tutoring Systems*. 2004. p. 624-634.
11. Woodcock, R.W., *Woodcock Reading Mastery Tests - Revised (WRMT-R/NU)*. 1998, Circle Pines, Minnesota: American Guidance Service.
12. Witten, I.H. and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. 2000: Morgan Kaufmann.
13. Hand, D., H. Mannila, and P. Smyth, *Principles of Data Mining*. 2001, Cambridge, Massachusetts: MIT Press.
14. Seidenberg, M.S. and J.L. McClelland, *A Distributed, Developmental Model of Word Recognition and Naming.* Psychological Review, 1989. **96**: p. 523-568.
15. Horvitz, E. and T. Paek. *Harnessing Models of Users' Goals to Mediate Clarification Dialog in Spoken Language Systems*. in *Eighth Conference on User Modeling, Sonthofen*. 2001. p. Sonthofen, Germany.
16. Singh, S., D. Litman, M. Kearns, and M. Walker, *Optimizing Dialogue Managment with Reinforcement Learning: Experiments with the NJFun System.* Journal of Artificial Intelligence Research, 2002. **16**(5): p. 105-133.
17. Mostow, J. and G. Aist. *The Sounds of Silence: Towards Automated Evaluation of Student Learning in a Reading Tutor that Listens*. in *Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI-97)*. 1997. p. 355-361 Providence, RI: American Association for Artificial Intelligence.