The role of positive feedback in Intelligent Tutoring Systems

Davide Fossati
Department of Computer Science
University of Illinois at Chicago
Chicago, IL, USA
dfossa1@uic.edu

Abstract

The focus of this study is positive feedback in one-on-one tutoring, its computational modeling, and its application to the design of more effective Intelligent Tutoring Systems. A data collection of tutoring sessions in the domain of basic Computer Science data structures has been carried out. A methodology based on multiple regression is proposed, and some preliminary results are presented. A prototype Intelligent Tutoring System on linked lists has been developed and deployed in a college-level Computer Science class.

1 Introduction

One-on-one tutoring has been shown to be a very effective form of instruction (Bloom, 1984). The research community is working on discovering the characteristics of tutoring. One of the goals is to understand the strategies tutors use, in order to design effective learning environments and tools to support learning. Among the tools, particular attention is given to Intelligent Tutoring Systems (ITSs), which are sophisticated software systems that can provide personalized instruction to students, in some respect similar to one-on-one tutoring (Beck et al., 1996). Many of these systems have been shown to be very effective (Evens and Michael, 2006; Van Lehn et al., 2005; Di Eugenio et al., 2005; Mitrović et al., 2004; Person et al., 2001). In many experiments, ITSs induced learning gains higher than those measured in a classroom environment, but lower than those obtained with one-on-one interactions with human tutors. The belief of the research community is that knowing more about human tutoring would help improve the design of ITSs. In particular, the effective use of natural language might be a key element. In most of the studies mentioned above, systems with more sophisticated language interfaces performed better than other experimental conditions.

An important form of student-tutor interaction is *feedback*. *Negative feedback* can be provided by the tutor in response to students' mistakes. An effective use of negative feedback can help the student correct a mistake and prevent him/her from repeating the same or a similar mistake again, effectively providing a learning opportunity to the student. *Positive feedback* is usually provided in response to some correct input from the student. Positive feedback can help students reinforce the correct knowledge they already have, or successfully integrate new knowledge, if the correct input provided by the student was originated by a random or tentative step.

The goal of this study is to assess the relevance of positive feedback in tutoring, and build a computational model of positive feedback that can be implemented in ITSs. Even though some form of positive feedback is present in many successful ITSs, the predominant type of feedback generated by those systems is negative feedback, as those systems are designed to react to students mistakes. To date, there is no systematic study of the role of positive feedback in ITSs in the literature. However, there is an increasing amount of evidence that suggests that positive feedback may be very important in enhancing students' learning. In a detailed study in a controlled environment and domain, the letter pattern extrapolation task, Corrigan-Halpern (2006) found that subjects given positive feedback performed better in an assessment task than subjects receiving negative feedback. In another study on the same domain, Lu (2007) found that the ratio of the positive over negative messages in her corpus of expert tutoring dialogues is about 4 to 1, and the ratio is even higher in the messages presented by her successful ITS modeled after an expert tutor, being about 10 to 1. In the dataset subject of this study, which is on a completely different domain —Computer Science data structures— such a high ratio of positive over negative feedback messages still holds, in the order of about 8 to 1. In a recent study, Barrow et al. (2008) showed that a version of their SQL-Tutor enriched with positive feedback generation helped students learn faster than another version of the same system delivering negative feedback only.

What might be the educational value of positive feedback in ITSs? First of all, positive feedback may be an effective motivational technique (Lepper et al., 1997). Positive feedback can also have cognitive value. In a problem solving setting, the student can make a tentative (maybe random) step towards the correct solution. At this point, positive feedback from the tutor may be important in helping the student consolidate this step and learn from it. Some researchers outlined the importance of self-explanation in learning (Chi, 1996; Renkl, 2002). Positive feedback has the potential to improve self-explanation, in terms of quantity and effectiveness. Another issue is how students perceive and accept feedback (Weaver, 2006), and, in the case of automated tutoring systems, whether students read feedback messages at all (Heift, 2001). Positive feedback might also make students more willing to accept help and advice from the tutor.

2 A study of human tutoring

The domain of this study is Computer Science data structures, specifically *linked lists*, *stacks*, and *binary search trees*. A corpus of 54 one-on-one tutoring sessions has been collected. Each individual student participated in only one tutoring session, with a tutor randomly assigned from a pool of two tutors. One of the tutors is an experienced Computer Science professor, with more than 30 years of teaching experience. The other tutor is a senior undergraduate student in Computer Science, with only one semester of previous tutoring experience. The tutoring sessions have been videotaped and transcribed. Student took a pre-test right before the tutoring session, and a post-test immediately after. An additional group of 53 students (control group) took the pre and post tests, but they did not participate in a tutoring session, and attended a lecture about a totally unrelated topic instead.

Topic	Tutor	Avg	Stdev	t	*df*	P
List	Novice	.09	.22	-2.00	23	.057
	Expert	.18	.26	-3.85	29	$< .01$
	Both	.14	.25	-4.24	53	$< .01$
	None	.01	.15	-0.56	52	ns
	iList	.09	.17	-3.04	32	$< .01$
Stack	Novice	.35	.25	-6.90	23	$< .01$
	Expert	.27	.22	-6.15	23	$< .01$
	Both	.31	.24	-9.20	47	$< .01$
	No	.05	.17	-2.15	52	$< .05$
Tree	Novice	.33	.26	-6.13	23	$< .01$
	Expert	.29	.23	-6.84	29	$< .01$
	Both	.30	.24	-9.23	53	$< .01$
	No	.04	.16	-1.78	52	ns

Table 1: Learning gains and t-test statistics

Paired samples t-tests revealed that post-test scores are *significantly higher* than pre-test scores in the two tutored conditions for all the topics, except for linked lists with the less experienced tutor, where the difference is only marginally significant. If the two tutored groups are aggregated, there is significant difference for all the topics. Students in the control group did *not* show significant learning for linked lists and binary search trees, and only marginally significant learning for stacks. Means, standard deviations, and t-test statistic values are reported in Table 1.

There is *no significant difference* between the two tutored conditions in terms of learning gain, expressed as the difference between post-score and pre-score. This is revealed by ANOVA between the two groups of students in the tutored condition. For lists, $F(1,53) = 1.82$, $P = ns$. For stacks, $F(1,47) = 1.35$, $P = ns$. For trees, $F(1,53) = 0.32$, $P = ns$.

The learning gain of students that received tutoring is *significantly higher* than the learning gain of the students in the control group, for all the topics.

This is showed by ANOVA between the group of tutored students (with both tutors) and the control group. For lists, $F(1, 106) = 11.0$, $P < 0.01$. For stacks, $F(1, 100) = 41.4$, $P < 0.01$. For trees, $F(1, 106) = 43.9$, $P < 0.01$. Means and standard deviations are reported in Table 1.

3 Regression-based analysis

The distribution of scores across sessions shows a lot of variability (Table 1). In all the conditions, there are sessions with very high learning gains, and sessions with very low ones. This observation and the previous results suggest a new direction for subsequent analysis: instead of looking at the characteristics of a particular *tutor*, it is better to look at the features that discriminate the most successful *sessions* from the least successful ones. As advocated in (Ohlsson et al., 2007), a sensible way to do that is to adopt an approach based on multiple regression of learning outcomes per tutoring session onto the frequencies of the different features. The following analysis has been done adopting a hierarchical, linear regression model.

Prior knowledge First of all, we want to factor out the effect of *prior knowledge*, measured by the pre-test score. A linear regression model reveals strong effect of pre-test scores on learning gain (Table 2). However, the R^2 values show that there is a lot of variance left to be explained, especially for lists and stacks, although not so much for trees. Notice that the β weights are negative. That means students with higher pre-test scores learn *less* then students with lower pre-test scores. A possible explanation is that students with more previous knowledge have less *learning opportunity* than students with less previous knowledge.

Time on task Another variable that is recognized as important by the educational research community is *time on task*, and we can approximate it with the length of the tutoring session. In the hierarchical regression model, session length follows pre-test score. Surprisingly, session length has a significant effect only on linked lists (Table 2).

Student activity Another hypothesis is that the degree of *student activity*, in the sense of the amount of student's participation in the discussion, might relate to learning (Lepper et al., 1997; Chi et al., 2001). To test this hypothesis, the following definition of student activity has been adopted:

$$\text{student activity} = \frac{\text{\# of turns} - \text{\# of short turns}}{\text{session length}}$$

Turns are the sequences of uninterrupted speech of the student. *Short turns* are the student turns shorter than three words. The regression analysis revealed *no significant effect* of this measure of students' activity on learning gain.

Feedback The dataset has been manually annotated for *episodes* where positive or negative feedback is delivered. All the protocols have been annotated by one coder, and some of them have been double-coded by a second one (intercoder agreement: kappa = 0.67). Examples of feedback episodes are reported in Figure 1.

The number of positive feedback episodes and the number of negative feedback episodes have been introduced in the regression model (Table 2). The model showed a significant effect of feedback for linked lists and stacks, but no significant effect on trees. Interestingly, the effect of positive feedback is *positive*, but the effect of negative feedback is *negative*, as can be seen by the sign of the β value.

4 A tutoring system for linked lists

A new ITS in the domain of linked lists, *iList*, is being developed (Figure 2).

The iList system is based on the *constraint-based* design paradigm. Originally developed from a cognitive theory of how people might learn from performance errors (Ohlsson, 1996), constraint-based modeling has grown into a methodology used to build full-fledged ITSs, and an alternative to the model tracing approach adopted by many ITSs. In a constraint-based system, domain knowledge is modeled with a set of *constraints*, logic units composed of a *relevance condition* and a *satisfaction condition*. A constraint is irrelevant when the relevance condition is not satisfied; it is satisfied when both relevance and satisfaction conditions are satisfied; it is violated when the relevance condition is satisfied but the satisfaction condition is not. In the context of tutoring, constraints are matched against student

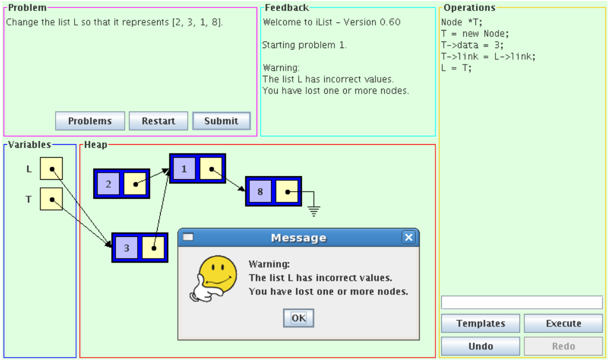
```
            T:   do you see a problem?
            T:   I have found the node a@l, see here I found the node b@l, and
                 then I put g@l in after it.
Begin +     T:   here I have found the node a@l and now the link I have to
                 change is +...
            S:   ++ you have to link e@l <over xxx.> [>]
End +       T:   [<] <yeah> I have to go back to this one.
            S:   *mmhm
            T:   so I *uh once I’m here, this key is here, I can’t go backwards.
```

```
Begin -     S:   <so you> [>] <you won’t get the same> [//] would you get the
                 same point out of writing t@l close to c@l at the top?
            T:   oh, t@l equals c@l.
            T:   no because you would have a type mismatch.
End -       T:   t@l <is a pointer> [//] is an address, and this is contents.
```

Figure 1: Positive and negative feedback (T = tutor, S = student)

Topic	Model	Predictor	β	R^2	P
List	1	Pre-test	-.45	.18	$< .05$
	2	Pre-test Session length	-.40 .35	.28	$< .05$ $< .05$
	3	Pre-test Session length + feedback - feedback	-.35 .33 .46 -.53	.36	$< .05$ .05 .05 $< .05$
Stack	1	Pre-test	-.53	.26	$< .01$
	2	Pre-test Session length	-.52 .05	.24	$< .01$ ns
	3	Pre-test Session length + feedback - feedback	-.58 .01 .61 -.55	.33	$< .01$ ns $< .05$ $< .05$
Tree	1	Pre-test	-.79	.61	$< .01$
	2	Pre-test Session length	-.78 .03	.60	$< .01$ ns
	3	Pre-test Session length + feedback - feedback	-.77 .04 .06 -.12	.59	$< .01$ ns ns ns
All	1	Pre-test	-.52	.26	$< .01$
	2	Pre-test Session length	-.54 .20	.29	$< .01$ $< .05$
	3	Pre-test Session length + feedback - feedback	-.57 .16 .30 -.23	.32	$< .01$ .06 $< .05$ .05

Table 2: Linear regression



Change the list L so that it represents [2, 3, 1, 8].

Welcome to iList - Version 0.60

Starting problem 1.

Warning:
The list L has incorrect values.
You have lost one or more nodes.

```
Node *T;
T = new Node;
T->data = 3;
T->link = L->link;
L = T;
```

Warning:
The list L has incorrect values.
You have lost one or more nodes.

Figure 2: The iList system

solutions. Satisfied constraints correspond to knowledge that students have acquired, whereas violated constraints correspond to gaps or incorrect knowledge. An important feature is that there is no need for an explicit model of students’ mistakes, as opposed to buggy rules in model tracing. The possible errors are implicitly specified as the possible ways in which constraints can be violated.

The architecture of iList includes a problem model, a constraint evaluator, a feedback manager, and a graphical user interface. Student model and pedagogical module, important components of a complete ITS (Beck et al., 1996), have not been implemented yet, and will be included in a future version. Currently, the system provides only simple negative feedback in response to students’ mistakes, as customary in constraint-based ITSs.

A first version of the system has been deployed into a Computer Science class of a partner institution. 33 students took a pre-test before using the system, and a post-test immediately afterwards. The students also filled in a questionnaire about their subjective impressions on the system. The interaction of the students with the system was logged.

T-test on test scores revealed that *students did learn* during the interaction with iList (Table 1). The learning gain is somewhere in between the one observed in the control condition and the one of the tutored condition. ANOVA revealed no significant difference between the control group and the iList group, nor between the iList group and the tutored group, whereas the difference between control and tutored groups is significant.

A preliminary analysis of the questionnaires revealed that students felt that iList helped them learn linked lists to a moderate degree (on a 1 to 5 scale: avg = 2.88, stdev = 1.18), but working with iList was interesting to them (avg = 4.0, stdev = 1.27). Students found the feedback provided by the system somewhat repetitive (avg = 3.88, stdev = 1.18), which is not surprising given the simple template-based generation mechanism. Also, the feedback was considered not very useful (avg = 2.31, 1.23), but at least not too misleading (avg = 2.22, stdev = 1.21). Interestingly, students declared that they read the feedback provided by the system (avg = 4.25, stdev = 1.05), but the logs of the system reveal just the opposite. In fact, on average, students read feedback messages for 3.56 seconds (stdev = 2.66 seconds), resulting in a reading speed of 532 words/minute (stdev = 224 words/minute). According to Carver's taxonomy (Carver, 1990), such speed indicates a quick skimming of the text, whereas reading for learning typically has a lower speed, in the order of 200 words/minute.

5 Future work

The main goal of this research is to build a computational model of positive feedback that can be used in ITSs. The study of empirical data and the system design and development will proceed in parallel, helping and informing each other as new results are obtained.

The conditions and the modalities of positive feedback delivery by tutors will be investigated from the human tutoring dataset. To do so, more coding categories will be defined, and the data will be annotated with these categories. The results of the statistical analysis over the first few coding categories will be used to guide the definition of more categories, that will be in turn used to annotate the data, and so on. An example of potential coding category is whether the student's action that triggered the feedback was prompted by the tutor or volunteered by the student. Another example is whether the feedback's content was a repetition of what the student just said or included additional explanation.

The first experiment with iList provided a comprehensive log of the students' interaction with the system. Additional analysis of this data will be important, especially because the nature of the interaction of a student with a computer system differs from the interaction with a human tutor. When working with a computer system, most of the interaction happens through a graphical interface, instead of natural language dialogue. Also, the interaction with a computer system is mostly student-driven, whereas our human protocols show a clear predominance of the tutor in the conversation. In the CS protocols, on average, 94% of the words belong to the tutor, and most of the tutors' discourse is some form of direct instruction. On the other hand, the interaction with the system will mostly consist of actions that students make to solve the problems that they will be asked to solve, with few interventions from the system. An interesting analysis that could be done on the logs is the discovery of sequential patterns using data mining algorithms, such as MS-GSP (Liu, 2006). Such patterns could then be regressed against learning outcomes, in order to assess their correlation with learning.

After the relevant features are discovered, a computational model of positive feedback will be built and integrated into iList. The model will encode knowledge extracted with machine learning approaches, and such knowledge will inform a discourse planner, responsible of organizing and generating appropriate positive feedback. The choiche of the specific machine learning and discourse planning methods will require extensive empirical investigation. Specifically, among the different machine learning methods, some are able to provide some sort of human-readable symbolic model, which can be inspected to gain some insights on how the model works. Decision trees and association rules belong to this category. Other methods provide a less readable, black-box type of models, but they may be very useful and effective as well. Examples of such methods include Neural Networks and Markov Models. The ultimate goal of this research is to get both an effective model and to gain insights on tutoring. Thus, both classes of machine learning methods will be tried, with the goal of finding a balance between model effectiveness and model readability.

Finally, the system with enhanced feedback capabilities will be deployed and evaluated.

Acknowledgments

This work is supported by award N00014-07-1-0040 from the Office of Naval Research, and additionally by awards ALT-0536968 and IIS-0133123 from the National Science Foundation.

References

Devon Barrow, Antonija Mitrović, Stellan Ohlsson, and Michael Grimley. 2008. Assessing the impact of positive feedback in constraint-based tutors. In *ITS 2008, The 9th International Conference on Intelligent Tutoring Systems*, Montreal, Canada.

Joseph Beck, Mia Stern, and Erik Haugsjaa. 1996. Applications of AI in education. *ACM crossroads*. http://www.acm.org/crossroads/xrds3-1/aied.html.

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

Ronald P. Carver. 1990. *Reading Rate: A Review of Research and Theory*. Academic Press, San Diego, CA.

Michelene T.H. Chi, Stephanie A. Siler, Heisawn Jeong, Takashi Yamauchi, and Robert G. Hausmann. 2001. Learning from human tutoring. *Cognitive Science*, 25:471–533.

Michelene T.H. Chi. 1996. Constructing self-explanations and scaffolded explanations in tutoring. *Applied Cognitive Psychology*, 10:33–49.

Andrew Corrigan-Halpern. 2006. *Feedback in Complex Learning: Considering the Relationship Between Utility and Processing Demands*. Ph.D. thesis, University of Illinois at Chicago.

Barbara Di Eugenio, Davide Fossati, Dan Yu, Susan Haller, and Michael Glass. 2005. Aggregation improves learning: Experiments in natural language generation for intelligent tutoring systems. In *ACL05, Proceedings of the 42nd Meeting of the Association for Computational Linguistics*, Ann Arbor, MI.

Martha Evens and Joel Michael. 2006. *One-on-one Tutoring by Humans and Machines*. Mahwah, NJ: Lawrence Erlbaum Associates.

Trude Heift. 2001. Error-specific and individualized feedback in a web-based language tutoring system: Do they read it? *ReCALL Journal*, 13(2):129–142.

M. R. Lepper, M. Drake, and T. M. O'Donnell-Johnson. 1997. Scaffolding techniques of expert human tutors. In K. Hogan and M. Pressley, editors, *Scaffolding student learning: Instructional approaches and issues*, pages 108–144. Brookline Books, New York.

Bing Liu. 2006. *Web Data Mining*. Springer, Berlin.

Xin Lu. 2007. *Expert Tutoring and Natural Language Feedback in Intelligent Tutoring Systems*. Ph.D. thesis, University of Illinois at Chicago.

Antonija Mitrović, Pramuditha Suraweera, Brent Martin, and A. Weerasinghe. 2004. DB-suite: Experiences with three intelligent, web-based database tutors. *Journal of Interactive Learning Research*, 15(4):409–432.

Stellan Ohlsson, Barbara Di Eugenio, Bettina Chow, Davide Fossati, Xin Lu, and Trina C. Kershaw. 2007. Beyond the code-and-count analysis of tutoring dialogues. In *AIED07, 13th International Conference on Artificial Intelligence in Education*.

Stellan Ohlsson. 1996. Learning from performance errors. *Psychological Review*, 103:241–262.

N. K. Person, A. C. Graesser, L. Bautista, E. C. Mathews, and the Tutoring Research Group. 2001. Evaluating student learning gains in two versions of AutoTutor. In J. D. Moore, C. L. Redfield, and W. L. Johnson, editors, *Artificial intelligence in education: AI-ED in the wired and wireless future*, pages 286–293. Amsterdam: IOS Press.

Alexander Renkl. 2002. Learning from worked-out examples: Instructional explanations supplement self-explanations. *Learning and Instruction*, 12:529–556.

Kurt Van Lehn, Collin Lynch, Kay Schulze, Joel A. Shapiro, Robert H. Shelby, Linwood Taylor, Don J. Treacy, Anders Weinstein, and Mary C. Wintersgill. 2005. The Andes physics tutoring system: Five years of evaluations. In G. I. McCalla and C. K. Looi, editors, *Artificial Intelligence in Education Conference*. Amsterdam: IOS Press.

Melanie R. Weaver. 2006. Do students value feedback? Student perceptions of tutors' written responses. *Assessment and Evaluation in Higher Education*, 31(3):379–394.