The Effect of Using Online Tutors on the Self-Efficacy of Learners

Amruth N. Kumar
Ramapo College of New Jersey
Mahwah, NJ 07430, USA
amruth@ramapo.edu

Abstract—We conducted a study to evaluate the effect of using software tutors on the self-efficacy of students—in particular, whether the type of activity covered by the software tutor correlated with any improvement in self-efficacy after using the tutor along the levels of Bloom’s taxonomy; and whether any differential effects could be observed in the improvement of self-efficacy among the sexes and racial groups. The study was conducted over four semesters using two different software tutors. The collected data was analyzed using paired sample t-test and 2 X 2 ANOVA for three different sets of students—those who used the tutor, those who needed to use the tutor and those who learned one or more concepts by using the tutor. We found that if a significant difference was found among taxonomic levels of self-efficacy after using a software tutor, the improvement was statistically significantly greater on the taxonomic level directly relevant to the topic/activity of the tutor than any other level on Bloom’s taxonomy except comprehension. Such a difference was found at least among those who actually learned one or more concepts using the tutor, if not everyone who used the tutor. In most cases, the improvement in self-efficacy resulting from the use of the software tutor was indistinguishable across sexes and racial groups. (Abstract)

Keywords—Online tutors; self-efficacy; evaluation

I. INTRODUCTION

Self-efficacy is one’s belief in one’s abilities [1]. Abilities, in particular as related to computer programming, can be categorized according to the various levels of Bloom’s taxonomy [2], as follows:

- Comprehension – ability to identify and recall data types, data forms, control statements, etc.
- Application – ability to predict the behavior/output of a given program
- Analysis – ability to debug a given program
- Synthesis – ability to write a program for a given problem statement
- Evaluation – ability to critique, refactor and rewrite a given program to meet one or more objectives such as readability, efficiency, etc.

How does using software tutors on programming affect the self-efficacy of students along these taxonomic levels? How does the activity covered by the software tutor correlate with their post-tutor assessment of self-efficacy along the levels of Bloom’s taxonomy? Is there any difference in post-tutor assessment of self-efficacy among demographic groups—sex (male versus female), or race (traditionally represented versus under-represented groups) [3]?

In order to answer these questions, we conducted a study over multiple semesters using problem-solving software tutors for computer programming, called problets (problets.org). After using each problet, students were asked to fill out a Likert-scale survey on self-efficacy. They were also asked to optionally fill out their demographic information.

The results of this study will be discussed in this paper. The results will be of interest to developers of software tutors as well as their users. Since self-efficacy affects retention in the major, the results will also inform the ongoing discussion on broadening participation in Computer Science.

II. PROTOCOL

Problets (problets.org) are problem-solving software tutors delivered over the web. Students typically use them after class as assignments. Problets cover all the topics typically covered in an introductory programming course, including expressions, selection, loops, functions, arrays and classes. They are available for C/C++, Java and C#, and have been continually used by third-party Computer Science educators since fall 2004.

When a student uses a problet on a given topic, the student is presented problems on specific concepts in the topic, the student is asked to solve the problem and submit an answer, and the tutor grades the student’s solution and provides feedback designed to help the student learn about the concept.

For this study, two problets/tutors were used, both dealing with the concept of functions:

- A tutor on code tracing – students were asked to identify the output of programs involving one or more functions. The tutor covered 10 concepts, including parameter passing, calling functions in expressions, multiple calls to a function, and variables with the same name in multiple functions (See Fig. 1).
A tutor on debugging – students were asked to identify bugs in programs involving one or more functions. The tutor covered 8-9 concepts (depending on the language – C++/Java/C#), including mismatch in the number and type of actual and formal parameters, missing return statement in a function, incorrectly calling a function that returns void versus non-void, and re-declaring a formal parameter as a local variable in a function (See Fig. 2).

Both these tutors are on advanced concepts that students in introductory programming courses find to be difficult.

Each tutor was configured to administer the following protocol:

- **Pre-test** – During the pre-test, the tutor presented one problem per concept. If a student solved a problem correctly, the student was given credit for the corresponding concept. No feedback was provided to the student, and no more problems on the concept were presented to the student. On the other hand, if the student solved a problem incorrectly, feedback was presented to the student immediately after the student had submitted his/her solution to the problem. Additional problems were presented on the concept during the subsequent stages described below.

- **Adaptive practice** – During this stage, additional problems were presented to the student on only the concepts on which the student had made mistakes when solving problems during the pre-test. For each such concept, the student was presented multiple problems until the student mastered the concept, i.e., solved at least 60% of the problems correctly. On each problem, the student received feedback including step-by-step explanation of the correct answer, which has been shown to improve learning [4].

- **Post-test** - During this stage, the student was presented test problems on the concepts that the student had mastered during adaptive practice.

- **Self-Efficacy Survey** – Students were asked to respond on a 5-point Likert scale to eight statements related to their self-efficacy with the topic of the tutor.

- **Demographics** - Students were provided the option to identify their demographic information, including sex and race. Demographic information was solicited after the pre-test-practice-post-test protocol to avoid the effects of stereotype threat [5].

The entire protocol was administered online, back-to-back, with no break in between, all by the software tutor [6]. The pre-test-practice-post-test protocol was limited to 30 minutes.

During the self-efficacy survey, students were asked to respond to the following eight statements related to their self-efficacy with the topic of functions:

After using the tutor, I can do the following better:

1. Understand the grammar rules of defining and calling functions
2. Understand the meaning of function definitions and calls
3. Read function definitions and calls
4. Predict the output of programs with functions
5. Debug programs with functions
6. Write programs with functions
7. Design programs with functions
8. Critique programs with functions

The 5-point Likert scale used for the responses ranged from Strongly Agree (coded as 1) to Strongly Disagree (coded as 5).

Note that the eight statements on the survey correspond to levels of Bloom’s taxonomy as follows:

- Statements 1-3: Comprehension, i.e., constructing meaning.
- Statement 4: Application, i.e., using learned material in new situations.
- Statement 5: Analysis, i.e., breaking down material into its components to diagnose it.
- Statements 6 and 7: Synthesis, i.e., putting parts together to form a new whole.
- Statement 8: Evaluation, i.e., critiquing the value of material for a given purpose.

The statement directly relevant to the topic of code-tracing tutor was statement 4 on application. Similarly, the statement directly relevant to the topic of debugging tutor was statement 5 on analysis. Both the tutors could also be argued to improve comprehension, as stated on statements 1-3. Neither tutor was designed or expected to affect synthesis, covered by statements 6 and 7.

If using the tutors affected the self-efficacy of students, we could expect to see the following behavior:

- Students would respond most positively on statement 4 after using code-tracing tutor, and on statement 5 after using debugging tutor;
- Statistically significant difference could be observed between the student responses to statements 4 and 5 after both tutors (code-tracing tutor did not cover debugging and vice versa).
- Statistically significant difference could be observed between the student responses to the directly relevant statement (4 for code-tracing, 5 for debugging) and statements 6-7, since neither tutor was designed to help students write functions.

III. DATA COLLECTION AND ANALYSIS

Data was collected over four semesters: fall 2012, spring 2013, fall 2013 and spring 2014. In each of these semesters,
students from multiple institutions used the tutors over the web.

The tutors had been used for a controlled study of the effect of using expressive writing on the test performance of students [7]. For the current study, the control and test groups of this prior study were analyzed separately.

The pre-test contained 10 problems in code-tracing tutor and 9 problems in debugging tutor. On each tutor, only those students who had solved at least 6 pre-test problems were considered for this study – this eliminated tentative and incomplete uses of the tutors from the study. Similarly, students who did not respond to the self-efficacy survey were also eliminated from the study.

After combining data from all four semesters, the numbers of students who participated in the study using the two tutors and under the two conditions were as provided in Table 1.

<table>
<thead>
<tr>
<th>TABLE I. NUMBER OF STUDENTS IN THE STUDY</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
</tr>
<tr>
<td>Code-Tracing</td>
</tr>
<tr>
<td>Debugging</td>
</tr>
</tbody>
</table>

For each of the four cases in Table 1, we considered three sets of students:

1. **Set A**: All those who had used the tutor – the numbers are as shown in Table 1;

2. **Subset N**: All those who *needed* to use the tutor – these were the students whose normalized pre-test score per problem was less than 1.0. In other words, these students solved at least one pre-test problem incorrectly, and could benefit from the subsequent adaptive practice session. These were a proper subset of all those who used the tutor (Set A) – this group excluded the students who solved all the pre-test problems correctly and never solved any practice problems.

3. **Subset L**: All those who *learned* one or more concepts by using the tutor – these were the students who solved pre-test, practice and post-test problems on at least one concept and demonstrated pre-post improvement in score on the concept [6]. These were a proper subset of all those who used the tutor (Set A) – this group excluded the students who solved all the pre-test problems correctly and never solved any practice problems.

For each of the above three subsets, in each of the four cases shown in Table 1, we conducted a five-way paired sample t-test of the responses on self-efficacy survey, with the pairings as shown in Table II.

<table>
<thead>
<tr>
<th>TABLE II. PAIRED SAMPLE T-TEST VARIABLES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variable 1</td>
</tr>
</tbody>
</table>

As mentioned earlier, we expected to observe significant differences in pairings C, D and E.

In addition, we conducted 2 X 2 ANOVA analysis, with sex and race as fixed factors, of the following three self-efficacy variables:

1. Average response on all 8 statements;
2. Response on statement 4 – the one most relevant to code-tracing tutor;
3. Response on statement 5 – the one most relevant to debugging tutor.

For this analysis, we considered two groups for race: traditionally represented, i.e., Caucasians and Asians [8], and the rest, i.e., Black/African American, Hispanic/Latino, Native American, Native Hawaiian/Pacific Islander and Other. In the study, we asked students to (optionally) identify their sex (biological notion of male/female) rather than their gender (social/cultural notion of man/woman) [9].

**A. Code Tracing Tutor – Control Group**

Table III lists the sample size N, mean and standard deviation of the responses to self-efficacy statements on comprehension, application, analysis and synthesis levels of Bloom’s taxonomy. The data is for all those who had used the tutor (Set A). The score on comprehension is the average of the responses to the first 3 statements and the score on synthesis is the average response to statements 6 and 7.

<table>
<thead>
<tr>
<th>TABLE III. CODE-TRACING TUTOR – CONTROL RESPONSES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group</td>
</tr>
<tr>
<td>Comprehension (1-3)</td>
</tr>
<tr>
<td>Application (4)</td>
</tr>
<tr>
<td>Analysis (5)</td>
</tr>
<tr>
<td>Synthesis (6-7)</td>
</tr>
</tbody>
</table>

Table IV lists the pairwise correlation (all correlations were significant at p < 0.001), the mean of the difference in pair-wise responses, the corresponding t-value and its 2-tailed significance for the 6 pair-wise comparisons.

<table>
<thead>
<tr>
<th>TABLE IV. CODE-TRACING TUTOR – CONTROL RESPONSES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pair</td>
</tr>
<tr>
<td>Compr. – Appln.</td>
</tr>
</tbody>
</table>
From Tables III and IV, it is clear that the student response was the most positive on statement 4 (application), which is the statement directly relevant to the topic of code-tracing tutor. Their response on this statement was statistically different from those on other levels of Bloom’s taxonomy, e.g., significance of the pairwise comparison of Application (4) versus Analysis (5) is given by \( t(125) = -4.423, p < 0.001 \). In this instance, the self-reported improvement in self-efficacy after using the software tutor was statistically significantly greater on the taxonomic level directly relevant to the topic of the tutor than any other level. The same results were also obtained for the students who needed the tutor (Subset N) and those who learned from using the tutor (Subset L).

Finally, the 2 X 2 ANOVA analysis of self-efficacy responses with sex and race as fixed factors did not yield any significant difference between the responses of the sexes or the racial groups.

C. Debugging Tutor – Control Group

Table VII lists the responses to self-efficacy statements for all those who had learned using the tutor (Subset L) in the control group of debugging tutor.

Table VIII lists results of the 6 pair-wise comparisons of the responses to the self-efficacy statements along Bloom’s taxonomic levels for all those who had learned using the tutor (Subset L).

From Tables VII and VIII, it is clear that the student response was the most positive on statement 5 (analysis) and statements 1-3 (comprehension). Statement 5 (analysis) is the statement directly relevant to the topic of debugging tutor. Their response on this statement was statistically different from those on all other levels of Bloom’s taxonomy except
comprehension. So, the self-reported improvement in self-efficacy after using the software tutor was statistically significantly greater on the taxonomic level directly relevant to the topic of the tutor than all other levels except comprehension.

However, these results were obtained for students who learned from using the tutor (Subset L). Among all those who used the tutor (Set A) or those who needed to use the tutor (Subset N), no significant difference was observed between analysis (5) and application (4). So, we qualify the above results as applying at least to those who actually learned using the tutor (Subset L), which is a proper subset of those who used the tutor (Set A) and those who needed to use the tutor (Subset N).

The 2 X 2 ANOVA analysis of self-efficacy responses with sex and race as fixed factors did not yield any significant difference between the responses of the sexes or the racial groups for any of the sets (A, L or N).

D. Debugging Tutor – Test Group

Table IX lists the responses to self-efficacy statements for all those who learned using the tutor (Subset L) in the test group of debugging tutor.

<table>
<thead>
<tr>
<th>Group</th>
<th>N</th>
<th>Mean</th>
<th>Std. Dev</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comprehension (1-3)</td>
<td>97</td>
<td>2.179</td>
<td>.978</td>
</tr>
<tr>
<td>Application (4)</td>
<td>97</td>
<td>2.18</td>
<td>1.09</td>
</tr>
<tr>
<td>Analysis (5)</td>
<td>97</td>
<td>2.19</td>
<td>1.014</td>
</tr>
<tr>
<td>Synthesis (6-7)</td>
<td>97</td>
<td>2.273</td>
<td>1.00</td>
</tr>
</tbody>
</table>

Table X lists results of the 6 pair-wise comparisons of the responses to the self-efficacy statements along Bloom’s taxonomic levels for all those who had learned using the tutor (Subset L).

<table>
<thead>
<tr>
<th>Pair</th>
<th>Corr.</th>
<th>Mean</th>
<th>t-value</th>
<th>Sig.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compr. – Appln.</td>
<td>.948</td>
<td>.004</td>
<td>.107</td>
<td>.915</td>
</tr>
<tr>
<td>Compr. - Analysis</td>
<td>.898</td>
<td>-.007</td>
<td>-.148</td>
<td>.882</td>
</tr>
<tr>
<td>Appln. - Analysis</td>
<td>.892</td>
<td>-.01</td>
<td>-.217</td>
<td>.829</td>
</tr>
<tr>
<td>Appln. – Synthesis</td>
<td>.884</td>
<td>-.098</td>
<td>-.290</td>
<td>.048</td>
</tr>
<tr>
<td>Analysis - Synthesis</td>
<td>.884</td>
<td>-.088</td>
<td>-1.782</td>
<td>.078</td>
</tr>
<tr>
<td>Comp. - Synthesis</td>
<td>.908</td>
<td>-.094</td>
<td>-2.195</td>
<td>.031</td>
</tr>
</tbody>
</table>

No significant difference was found among comprehension, application and analysis statements in Tables IX and X, although analysis (5) was the taxonomic level directly relevant to the topic of the tutor. Even the difference observed between synthesis and the other levels in Table X for those who learned using the tutor (Subset L) was not observed for those who used the tutor (Set A) or those who needed to use the tutor (Subset N). In light of these results, we qualify the earlier results as observable provided a significant difference is found among taxonomic levels of self-efficacy.

On the 2 X 2 ANOVA analysis of the responses for statement 5 for the students who learned using the tutor (Subset L), we found:

- Significant main effect for race [F(1,81) = 4.931, p = .029]: Caucasians and Asians agreed with the statement (2.00 ± .278, N=58) more than students from underrepresented groups (2.559 ± .416, N=24)
- Marginally significant main effect for sex [F(1,81) = 3.073, p = .084]: male students agreed with the statement (2.059 ± .266, N=60) more than female students (2.50 ± .425, N=22)
- Marginally significant interaction between sex and race [F(1,81) = 3.073, p = .084]: whereas Caucasians and Asians agreed with the statement the same whether male (2.00 ± .283, N=43) or female (2.00 ± .479, N=15), underrepresented male students agreed with it (2.118 ± .45, N=17) more than underrepresented female students (3.00 ± .701, N=7).

The above differences in self-efficacy were found even though no significant differences were found on:

- Prior preparation, i.e., average score per problem on the pre-test between the sexes [F(1,82) = 1.22, p = .273] or the racial groups [F(1,82) = .188, p = .666]
- Concepts learned, between the sexes [F(1,82) = .012, p = .911] or the racial groups [F(1,82) = .00, p = .999]

However, the differences between the sexes and racial groups on self-efficacy were not found for the other two sets – those who used the tutor (Set A) and those who needed the tutor (Subset N). They were not observed in any of the other conditions either: control group of debugging tutor, or control or test group of code-tracing tutor. So, while there may be differences in the improvement of self-efficacy of sexes and racial groups resulting from the use of software tutors, additional data collection and analysis is needed to confirm or refute it, and if confirmed, identify the conditions under which such differences manifest.

IV. Discussion

We found that the improvement in self-efficacy after using the software tutor was statistically significantly greater on the taxonomic level directly relevant to the topic of the tutor than all other levels of Bloom’s taxonomy except comprehension. That we found this result in three out of four cases adds credence to the result. However, we found that the correlation between the number of concepts learned and the response to the self-efficacy statement directly relevant to the taxonomic level of the software tutor was minimal in all four cases (e.g., .06). In other words, those who learned more concepts did not necessarily agree more with the statement about improved self-efficacy. This lack of correlation is to be expected, since, testimonials can mislead [10].
In three out of the four cases, the improvement in self-efficacy resulting from the use of the software tutor was indistinguishable across sexes and racial groups. Higher level of self-efficacy has been shown to result in higher performance accomplishments [11]. Given the significant underrepresentation of female and minority students in Computer Science, using software tutors to improve their self-efficacy has the potential to broaden their participation.

In the current study, students had to fill out the self-efficacy survey after having spent 30 minutes solving problems with the software tutor. Filling out the survey was optional, and contained eight statements. Cynicism would dictate that students would skip the self-efficacy survey altogether or fill it out perfunctorily, with the same score on all eight statements. The results of this study are all the more remarkable given these adverse conditions.

Whereas self-efficacy instruments have been created and validated on computing [12], programming [13], introductory courses [14], etc., the focus of this study was a small subset of programming, viz., functions. So, a self-efficacy survey was custom-created for the study, as recommended in literature [15]. Many of the principles for creation and administration of self-efficacy surveys listed in literature [15] were followed: survey items were phrased in terms of “can do” rather than “will do”; the survey was multi-faceted – the items were based on analysis of the programming domain along Bloom’s levels of taxonomy; students were instructed to judge their capabilities after using the tutor, and not their potential or future capabilities; and a nondescript title of “Feedback” was used for the survey rather than “Self-Efficacy”. However, the 5-point Likert scale used in this study is considered less sensitive and reliable – scales with 10 intervals or even 100 possible responses are recommended [16]. That statistically significant results were found even with the less sensitive 5-point scale makes the results all the more significant. However, the survey instrument has not been empirically validated, which is a confounding factor of this study.

We plan to repeat the study using data collected with some of the other tutors, including those on expression evaluation and advanced loop concepts, to see if we can replicate the results.

ACKNOWLEDGMENT

Partial support for this work was provided by the National Science Foundation under grant DUE-1432190.

REFERENCES

Figure 1: Snapshot of Code Tracing Tutor in Java: Problem shown in the left panel, instructions for solving the problem shown in the top right panel and student’s attempt is shown in the bottom right panel – correct steps are marked in green and incorrect steps are marked in red. Button to submit the answer is shown at the bottom right.

Figure 2: Snapshot of Debugging Tutor in C++: Problem shown in the left panel, instructions are shown in the top right panel and student’s attempt is shown in the bottom right panel – correct steps are marked in green and incorrect steps are marked in red. Button to submit the answer is shown at the bottom right.