Abstract
Current tutoring systems are often impersonal and static, failing to adapt to the needs and wants of the student. Because students may have vastly different learning styles and motivations, these hurdles are extremely difficult to overcome. In this paper, we describe the design of an intelligent tutoring system superior to current systems. We enlist the help of a humanoid robot acting as a social agent to facilitate the interaction. The physical presence of a social robot has been shown to improve cognitive abilities among children utilizing the system due to increased engagement and compliance. Furthermore, poor help-seeking behavior has been shown to limit the usefulness of a tutoring system. These mistakes often manifest themselves in one of two ways - students either ‘game the system’ and ask for hints until they reach the answer or are ‘help-averse’ and fail to ask for help even when needed. Accordingly, we implement an adaptive model that attempts to categorize students based on their interactions with the system, tailoring itself to reduce instances of the child’s poor help-seeking behavior. We also describe technologies that will help us improve this model in the future. This paper will proceed to describe the culmination of a year long project that seeks to create a new and improved intelligent tutoring system with real time adaptation.

1 Motivation
As technology advances, so does the prevalence of automated learning. Students seeking to learn can absorb information from a variety of teaching software that can be more cost efficient than hiring teachers and more helpful than learning in groups. Personalized, effective attention at a fraction of the cost of a human is the goal of any Intelligent Tutoring System (ITS).

It is difficult, however, to build a system that can communicate effectively with students. A computer has a hard time adapting to a student’s needs, failing to understand how that student learns or what that student’s motivations are. One popular general model is a system with a series of problems and hints, allowing students to work at their own pace while asking for help when needed.

This model, however, gives rise to a host of issues. These tutoring systems are often impersonal, resulting in a detachment from learning especially among children. Moreover, students learn differently, and these systems are usually designed in a one size fits all manner. The students themselves can have trouble utilizing the system and fully taking advantage of its material.

The latter often manifests itself in the form of poor help-seeking behavior. These poor behaviors are often categorized into two groups. The first takes the form of ‘gaming the system,’ where a student recklessly asks for all the hints possible to solve the problem as quickly as possible. In this case, the student may not actually absorb the material. The other group is a ‘help averse’ group of students who do not ask for many hints, even when they may be helped by the hints.

In this paper, we seek to build a system that effectively solves these problems. We use a humanoid robot to personalize the tutoring experience, and we apply heuristics to reduce instances of poor help-seeking behavior. We examine the construction of an adaptive tutoring system that will tailor itself to a child’s specific needs to improve the overall learning experience for a tutoring system.

2 Related Works
There has been substantial research regarding help-seeking behavior and more specifically, interactive learning environments (ILEs). The vast majority of these studies do not involve a robot helper or a real-time adaptive model that adjusts to the participant on the fly. To create an effective ILE, one must first
identify how people seek help, and in what way the system must provide help.

Studies have already shown that on-demand help in which a tutor provides a hint whenever thechild requests it, is effective [10, 2]. For example, the Geometry Cognitive Tutor, a program that offers no personalization and simply offers on-demand hints, was shown to be more effective than classroom instruction when used in conjunction with the curriculum of the Integrated Learning Environment (ILE) [10]. Additionally, there are studies that show help-seeking is a goal oriented behavior, and unsuccessful attempts at seeking help may be met with a reluctance to solve the problem [2]. The implications of the study suggest that children may require immediate gratification in the form of on-demand hints. It is also imperative, then, that hints requested be relevant and helpful, with dead end hints reversing progress.

There is much room for improvement over the on-demand model, however. Aleven et. al estimates that 72 percent of help requests are unproductive in on-demand tutoring, showing that help requests often come at the wrong time in these scenarios [1]. Again, these poor help-seeking behaviors usually take one of two forms, with the student either asking for too many hints in rapid succession or not asking for enough hints, needlessly struggling through the problem. Personalizing hints in the form of better timing, then, could facilitate better learning.

Further validating the adaptive help-seeking model, Roll et. al discuss a new Help Tutor used in conjunction with the Geometry Cognitive Tutor that coaches students on when to ask for hints, resulting in students making less help-seeking errors and seeking out less hints in general [10]. They identify the main problem as the overuse of the help-seeking system, providing suggestions that advise against requesting a hint too often, ultimately reducing the number of hints the children ask for on average and improving help-seeking behavior. Learning gains were substantial over the 3 weeks (6 sessions), though in some cases it was noted that there may have been an aversion to using the Help Tutor, since those that did not want to use the Geometry Tutor may also have spurned the second one. These studies provide a solid foundation with findings that support the possibility of an even more effective intelligent tutoring model that involves robots. This paper will go on to discuss these timing improvements within the adaptive model that while simple, improve the productivity of hint requests during a multi-session robot-child tutoring interaction.

Previous work in tutoring presented to the Human Robot Interaction (HRI) community has focused on demonstrating the efficacy of social robots in a tutoring setting. Kanda et. al shows the establishment of a stronger relationship between the child and a robot that exhibits social behavior, while the positive effects of having a physically present entity are cited as well [6]. Even more improvements to the adaptive model described previously, therefore, could be made by simply having a physical presence, perhaps strengthening the relationship between the child and the tutoring system and facilitating better help-seeking habits. To support this, Howley et. al describes the potential advantages of a robot tutor over a human tutor in some situations due to distinctions in social role and help seeking [5].

Mohseni-Kabir et. al reinforce the idea that a robot dialogue can promote learning gains within the context of a bidirectional coaching model between an adult and a robot [9]. The paper uses a similar workflow of participants taking an initial test, engaging in a learning activity either with a robot providing additional hints or without, then finishing with a post test. It shows that a robot providing suggestions at the right time can significantly increase the ability of a person to perform a task and retain knowledge, allowing the person to learn the task much more efficiently. While much of this work is related to the study conducted in this paper, no prior work in HRI aims to understand the effects of shaping productive help-seeking behaviors in a robot tutoring scenario.

Furthermore, in examining affective behavior within a tutoring interaction, Spathuling has shown that children show more emotion when engaging with a physical robot in a one on one interaction, as well as demonstrated the viability of using affect detection in this context [12]. There have been numerous other studies that have integrated the analysis of affective behavior into a tutoring interaction [3].

3 Previous Study

This paper builds on a paper by Aditi Ramachandran, Alex Litou, and Brian Scassellati that was accepted to the HRI conference. The previous paper describes a study of a rudimentary adaptive help seeking model for children in an intelligent robot tutoring system. This methods outlined in this paper will build upon its predecessor, identify points of improvement, and build a new adaptive model given a wealth of collected data.
3.1 Methodology

The study recruited 33 students in the fifth and sixth grade from local New Haven public schools, four of whom were excluded from the final analysis due to absences and non-compliance. An Android tablet application was written from scratch to tutor mathematics, and students were asked to complete four sessions of eight questions each. Each question had a total of five possible attempts (before moving on to the next question) and three possible hints (requested in succession). A humanoid NAO robot (see Figure 1) facilitated the interaction, adding comments and encouragement for the children.

To measure learning gains across the interaction, students were asked to complete a pretest to judge prior knowledge and similar posttest. They were also asked to complete a self efficacy survey, which results in a Relative Autonomy Index (RAI) [11]. The RAI is a general, widely accepted measure of how much responsibility a child takes for his/her own learning.

3.2 Experiment

The goal of the study was to attempt to reduce poor help-seeking behavior and thus facilitate learning gains. It also utilized a social robot as a proof of concept for the human robot tutoring system. Specifically, the two main groups of poor help-seeking behavior, those that ‘game the system’ and those that are ‘help-averse’ were targeted. The NAO robot, trusted by the children, provided the shaping suggestions targeted at this suboptimal behavior, differentiating this study from the static ‘Help Tutor’ described in the related works section.

3.3 Adaptive Model

The children were split up into two groups, an adaptive model and a control model. The control group simply completed the questions with the robot providing hints on demand. The adaptive group, however, had the robot apply two simple heuristics. When requesting three consecutive hints, the robot would tell the student to make an attempt before providing him/her with the last hint. Additionally, when a student had made two incorrect attempts in a row, the robot would automatically provide him/her with a hint without a prompt from the student.

The first of the four sessions were the same for both groups. The last three sessions for the student in the adaptive group saw these heuristics applied. In the end, the group that received the adaptive behavior from the robot showed statistically significant learning gains from the pretest to the posttest, while the control group did not.

4 Goals

With knowledge of the success of this prior study, we move forward to improve upon the adaptive model. We seek to analyze the data obtained in the study, extract key features, and build a new model that reflect this knowledge. More specifically, we seek to cluster students with similar learning patterns together, then find features that represent these groups of children to better understand them. We attempt to replace knowledge about a child (like RAI or a pretest score) with patterns seen in a child’s interactions with the tutoring system.

We then look into using a child’s affective behavior to further tailor the system to the child’s needs. We look into numerous such systems to extract features from the middle of an interaction.

To do this, we used multiple python scripts for data extraction and manipulation. We primarily used Matlab and Excel for data visualization, and we used csv files to hold much of the intermediate and final data. We examined a variety of affect detection packages, which we will go into more detail later.

5 Key Features

5.1 Relative Autonomy Index

The RAI, mentioned briefly above and detailed in a 1989 Ryan and Connell study, is a good indicator of
a child’s reason for learning [11]. It measures, primarily, whether a child, in the context of performing school related activities, is intrinsically or externally motivated. It is scored from -4 to 4, with the low end representing a student who is almost entirely externally motivated (by parents, grades, etc.) and a 4 representing a student who is almost entirely intrinsically motivated (by a desire to learn, etc.). It is used in multiple studies, usually on a binary scale (positive RAI score versus negative RAI score) or as a continuous value [4, 8].

This feature is important because children on either end of the spectrum can be expected to learn very differently. If we can find indicators in the child’s tutoring interaction that can represent points in the RAI, then we can better adapt our model to the child.

5.2 Pre/Post Test Scores

Benchmarking children can lead to different ways to approach a tutoring interaction as well. Clearly, a student who already knows most of the material would not require the same help as a student who is completely lost and knows nothing. We need to either obtain this information in the form of a pretest or extract this information during the tutoring interaction so that the robot can adapt to the specific needs of the child. Pretest scores and posttest scores are collected for each of the participants for further analysis.

5.3 Give Up Attempts

Next, we define a ‘give up’ attempt to target children who disengage with the system. We define such an attempt to be two attempts made in rapid succession, and we initially set this time limit to twenty seconds. Note that the robot talks between every attempt, so a break of only twenty seconds means that the child makes another attempt without really reconsidering the problem at hand. The ‘give up’ attempt proves to be very useful later on and was extracted through close analysis of video logs from the interactions.

5.4 Feature Collection

Other important features involve number of hints denied by the robot or number of hints automatically prompted by the robot. They span timestamp manipulation and distribution of hint requests and attempts. We write a python script to collect all of this from the raw data, which looks like a sequence of events (such as robot speech, question starts, hints, and attempts) and their associated timestamps. These features are extracted and placed into a csv file for further manipulation.

6 Preliminary Analysis

As a first step, we attempt to categorize the children into groups based on help-seeking behavior. We collect two features from the first session (which was the same for all the children): number of potential automatic hints and number of potential denied hints. These auto and denied hints result from the same heuristics as described above.

We run k-means on the two values, finding the best clustering based on mean silhouette values. We obtain a mean silhouette value of .7457 with three clusters, as shown in Figure 2. We observe the three clusters as students who often have hints prompted in the upper left corner (help-averse), students who commit few help-seeking mistakes in the bottom left, and finally students who often have hints denied in the bottom right corner (gaming the system). This clear distinction lends our analysis strength in determining how exactly these children interact and learn.

7 Scripts

We utilize scripts written over the course of the semester to form a basis for future analysis. These scripts allow for quick clustering of data and speedy formation of graphs to compare variables, setting the stage for later work. We use python to aggregate raw
data into csv files, which are then graphed using Excel or fed into Matlab scripts to run k-means. We utilize the plot and scatter3 functions to visualize data in Matlab, as well as rely on silhouette values, mean silhouette values, and average distances from clusters as a sanity check for the generated clusterings.

7.1 K-Means

We run k-means with a distance function of squared Euclidean distance. Accordingly, when we compare unlike variables, we need to normalize them in some way to balance the distance function [13]. We write a pre-processing script that takes csv columns and normalizes them into values between 0 and 1, with the formula

\[
\frac{(value - minvalue)}{(maxvalue - minvalue)}
\]

where value is the entry, minvalue is the minimum value in the column, and maxvalue is the maximum value in the column. This allows k-means to run without being sensitive to the very different magnitudes of values that we have in the features [13]. While this approach is not perfect because of the somewhat arbitrary selection of minimum and maximum values on continuous values that have true minimums and maximums, it is a well researched approach that we take.

8 Grouping

As we continue our analysis, it helps to implement rudimentary grouping schema to delineate differences between groups. Some of the more obvious metrics of where a student lies are their RAI and pretest scores.

8.1 RAI Groups

As we mentioned above, RAI is often grouped by low RAI and high RAI. After running k-means with two clusters on the RAI scores of the students, we get two cluster centers at 1.02 and -1.99. K-means with a cluster value of two gave a mean silhouette value of 0.71, while more clusters gave lower values which validates the decision. With these clusters, we then compare mean values across clusters, as shown in Figure 3.

It is apparent from the graph that the average number of denied hints, automatic hints, and hints requested until answering a problem correctly is higher amongst those students with low RAI scores, while the average number of problems answered correctly on the first try is lower. These features, then, could be indicative of a student’s RAI scores.

8.2 Pretest Groups

For pretest scores, we run k-means on three clusters and obtain a mean silhouette value of 0.79. The cluster centers are 0.36, 3.09, and 5.91, representing low, medium, and high pretest scores. This is shown in Figure 4. As in the RAI grouping, it is apparent that pretest score is indicative of these features. Lower pretest scores correspond to more denied and auto hints as well as hints until an answer is correct, while higher pretest scores correspond to fewer instances of bad help-seeking behavior. Unsurprisingly, the number of questions answered correctly on the first try corresponded to pretest score as well.

8.3 Combined Groups

We then combine our two groupings to create six new groups, each indicating an RAI grouping and a pretest grouping. Group 1 corresponds to a low
pretest score and a low RAI, group 2 corresponds to a low pretest score and a high RAI, group 3 corresponds to a medium pretest score and a low RAI, group 4 corresponds to a medium pretest score and a high RAI, group 5 corresponds to a high pretest score and a low RAI, and group 6 corresponds to a high pretest score and a high RAI. Figures 5 and 6 show a sample of the data, which again point to the same trends.

9 Multiple Features

We find other important clusterings of data when we graph 3 dimensional plots. We utilize the same approach as before, normalizing values before running k-means. These clusterings can show the interaction of new variables with known clusters, such as a 3 dimensional plot involving automatically prompted hints, denied hints, and a third feature as shown in Figure 7. In this figure, the green asterisks are the cluster centers. Furthermore, interactions between seemingly unrelated variables that we attempt to separate and understand can be leveraged to create new heuristics in the new adaptive model. Figure 8 provides an example of this, although it is difficult to view these three dimensional plots in two dimensions.

10 Preliminary Conclusions

10.1 Intrinsic Motivation

We begin to analyze the RAI a little bit more, and we separate out the pieces of RAI into intrinsic and external motivations. To recap, the intrinsic motivation score corresponds to a person’s inner desire to do something for the sake of the action itself. We decided that perhaps those with a higher desire to learn from the robot for the sake of learning would teach a lesson or two.

We find that intrinsic motivation is correlated
strongly to two features that we have extracted, the number of give up attempts, and the time it takes for a student to first answer the question. We run linear regressions on each of these features with the intrinsic motivation score, and obtained p-values of less than 0.5. We have attached the graphs of these correlations in Figures 9 and 10.

Our previous analysis showed that RAI is highly correlated with pretest score, thus showing that high performing students also have higher RAI scores. One may expect that in correlating the number of ‘give up’ attempts and time until first attempt that higher performing students would have a strong effect on the results. We found, however that a higher intrinsic motivation score did not correlate at all with a higher pretest score (which indicates a high performing student) as shown in Figure 11.

10.2 Analysis

With these correlations, we begin to paint a picture of a student. With higher intrinsic motivation scores, students are more likely not to submit ‘give up’ attempts, and also take less time to submit their first answer. Thus, in our new model, we prompt a student who makes a ‘give up’ attempt with a hint, re-engaging the student with the problem because we understand that the student may not be self invested in engagement. We also decide to prompt a student who takes too long on a question with a hint, because a student who takes longer to submit an answer is likely to be disengaged with the robot, allowing the hint to attract the attention of the student. These two new heuristics should serve to strengthen the tutoring interaction and improve student engagement, especially among those with low intrinsic motivation scores.

These improvements could be applied in two ways. We can realize that student with low intrinsic motivation scores (taken before the session) need to be prompted with hints in these scenarios to improve engagement. The other method is to mine features continually during an interaction and classify the student into having a low intrinsic motivation score. We can then apply these heuristics in the same way.

11 Affective Behavior

We then examine integrating the mining of affective behavior during the tutoring interaction to improve the adaptive model. We considered a few different systems, each with its own integration and implementation difficulties.

11.1 Affdex

We started off by analyzing Affdex Affectiva software, specifically the Android SDK. Of the systems considered, Affdex offered the most number of features as well as emotions - in addition to features like eyebrow furrowing and attention, they also provided
analysis on whether the child was happy, surprised, angry, etc. The Android SDK was very responsive and looked good.

However, the field of vision of the application did not lend itself well to the scope of the project. The participant had to be looking straight onto the camera for it to register emotions, and the child simply would not be doing that in an interaction. Furthermore, directing a cell phone towards the face of a child is perhaps a little more difficult than a web cam of some sort.

11.2 Attention Tracker

We then examined an attention tracker described by Lemaignan et. al. in a paper detailing their attention tracking feature, ‘with-me-ness’ [7]. The paper detailed the mining of certain points on the participant’s face then constructing projections to determine where the participant’s attention was directed. This worked pretty well, except for a poorly maintained and documented repository that would have been very difficult to work with. Additionally, like with Affdex, the field of view was severely limited - if the participant had turned his or her head too far, the tracking would no longer be effective.

11.3 CLM Tracker

The final tool analyzed, the CLM Tracker, or Cambridge Face Tracker, had a larger field of vision and a well documented code base. We are currently working to integrate the CLM tracker into the next iteration of the tutoring project.

12 Concluding Remarks

With all the background data analyzed, as well as the specific heuristics mentioned, we seek to finish constructing a new, more-effective adaptive model for social robot tutoring. We plan on fully integrating real-time attention tracking to learn more about where the student’s attention lies, as well as learn more about how to engage students. We plan on extracting ever more features from a different type of interaction, learning more and more heuristics that we can apply to children in particular.

In this paper, we have detailed the year-long senior project that outlined feature extraction for an adaptive social robot tutoring interaction, found some interesting data points and results, and sought to explore technologies that will culminate in an improved tutoring system.

References

