Exploring Gaze Data for Determining User Learning with an Interactive Simulation

Samad Kardan[1] and Cristina Conati[1]

[1] Department of Computer Science, University of British Columbia
2366 Main Mall, Vancouver, BC, V6T1Z4, Canada
{skardan,conati}@cs.ubc.ca

Abstract. This paper explores the value of eye-tracking data to assess user learning with interactive simulations (IS). Our long-term goal is to use this data in user models that can generate adaptive support for students who do not learn well with these types of unstructured learning environments. We collected gaze data from users interacting with the CSP applet, an IS for constraint satisfaction problems. Two classifiers built upon this data achieved good accuracy in discriminating between students who learn well from the CSP applet and students who do not, providing evidence that gaze data can be a valuable source of information for building user modes for IS.

Keywords: Eye-tacking, Eye Movement Data, Interactive Simulation Environments, User Classification, User Modeling

1 Introduction

In recent years, there has been increasing interest in using interactive simulations (IS) for education and training. The idea underlying these environments is to foster experiential learning by giving students the opportunity to proactively experiment with concrete examples of concepts and processes they have learned in theory. One possible drawback of IS is that not all students learn well from this rather unstructured and open-ended form of interaction (e.g., [1]). These students may benefit from having additional guidance when they interact with an IS. The long-term goal of our research is to devise mechanisms to provide this guidance in real-time during interaction, personalized to the needs of each individual student. Detecting these needs, however, is challenging because there is still limited knowledge of which behaviours are indicative of effective vs. non-effective interactions with an IS. In previous work [2], we showed that it is possible to build user models that can classify successful vs. unsuccessful learners in a IS using logs of user interface actions. In this paper, we investigate student gaze data as an additional source of information to give to a user model for assessing how well a user learns with an IS. Initial results on the value of eye-tracking data in user-modeling for IS were presented in [3] and [4]. They looked at gaze information related to the occurrence of a simple gaze pattern defined a priori as being relevant for learning with an IS for mathematical functions. We extend this

adfa, p. 1, 2011.

work by looking at a much broader range of general eye-tracking features, in the context of a different IS. This is an important contribution to research in user modeling for IS, because pre-defining gaze patterns that indicate learning (as was done in [3, 4]) may not always be easy or possible, due to the often unstructured and open-ended nature of the interaction that IS support. Furthermore, such pre-defined patterns are task specific, and may not directly transfer to a different IS. In contrast, our approach is more general and can be applied to a variety of IS. It relies on giving to a classifier user model a broad range of standard eye-gaze features that are either task independent or based solely on identifying the main components of the target IS interface. Then, it is left to the classifier to identify patterns that are indicative of users' learning with that IS. An additional difference of our work from [3, 4] is that, in [3,4], gaze data was integrated with information on action logs, whereas we look at gaze data only, to directly evaluate its value in assessing learning in IS. We discuss the performance of two gaze-based classifiers for modeling users who interact with the CSP applet, an IS that demonstrates the workings of an algorithm for constraint satisfaction problems (CSP). We show that these classifiers achieve good accuracy in discriminating between students who learn well from the CSP applet and students who do not, thus providing further evidence of the value of gaze data for user modeling in IS.

In the rest of the paper, we first discuss related work. Next, we describe the CSP applet, and the study we ran to collect the necessary eye-tracking data. After discussing data pre-processing, we illustrate the performance of two different classifiers built on this data. We conclude with a discussion of the future work.

2 Related work

Using eye tracking to understand cognitive constructs such as intentions, plans or behaviour has received a lot of attention in psychology (e.g., [5, 6]). Researchers in human computer interaction and intelligent interfaces also started looking at gaze data as a source of information to model relevant cognitive processes of users during specific interaction tasks. For instance, gaze data has been investigated to capture users' decision making processes during information search tasks (e.g., [7, 8]), for activity recognition during working with a user interface (e.g., [9]), to predict word relevance in a reading task [10], to predict how well users process a given information visualization (e.g., [11, 12]), and to estimate mental workload in relation to evaluating users' interruptibility (e.g., [13]). Muldner et al. [14] looked at pupil dilation to detect relevant user affective states and meta-cognitive processes during the interaction with a learning environment that supports analogical problem solving. Knoepfle et al. [15] used eye-tracking data for comparing existing theories of how users learn to play strategies in normal-form games. The theories were compared in terms of how they could predict users' moves and attention to relevant information during interaction with a computer card game, with all theories showing limited predictive power.

In our work, we are interested in investigating whether a user's gaze patterns during interaction with an IS can be used to assess if the student is learning. We were inspired by existing research showing that it is possible to identify distinctive patterns in the gaze data of successful vs. unsuccessful users during simple problem solving and question answering tasks (e.g., [16–19]). In this body of work, the attention patterns analyzed related mainly to processing the problem description [16] or supporting visual material [17–19]. The main finding was that successful problem solvers pay more attention to information relevant to answer correctly, while unsuccessful problem solvers show more scattered attention patterns. Eivazi and Bednarik [20] went a step further showing that it is possible to build a classifier that relies solely on gaze data to predict users' performance during an interactive 8-tile puzzle game. Conati and Merten [3] and Amershi and Conati [4] present results that are even more relevant for our work, since they also looked at gaze-data to model student reasoning and learning during interaction with an IS. As explained earlier, the student models in [3,4] combine simple gaze-pattern information with information on the user's interface actions, whereas in this paper we focus on gaze data only, in a broader and more generalizable manner, to better isolate its potential as a source of information for user modeling in IS.

3 The AISpace CSP applet

The Constraint Satisfaction Problem (CSP) Applet is one of a collection of interactive tools for learning Artificial Intelligence algorithms, called AIspace [21]. Algorithm dynamics are demonstrated via interactive visualizations on graphs by the use of color and highlighting, and graphical state changes are reinforced through textual messages.

CSP consists of a set of variables, variable domains and a set of constraints on legal variable-value assignments. Solving a CSP requires finding an assignment that satisfies all constraints. The CSP applet illustrates the Arc Consistency 3 (AC-3) algorithm for solving CSPs represented as networks of variable nodes and constraint arcs. AC-3 iteratively makes individual arcs consistent by removing variable domain values inconsistent with a given constraint, until all arcs have been considered and the network is consistent. Then, if there remains a variable with more than one domain value, a procedure called domain splitting can be applied to that variable to split the CSP into disjoint cases so that AC-3 can recursively solve each case.

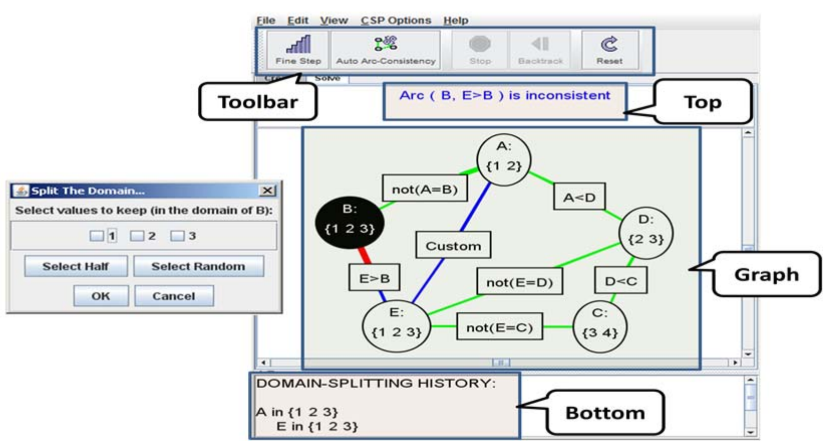

Fig. 1. CSP applet with example CSP problem

The CSP applet provides several mechanisms for the interactive execution of the AC-3 algorithm on a set of available CSP problems. These mechanisms are accessible through the toolbar shown at the top of Fig. 1 or through direct manipulation of graph elements. The user can, for instance: (i) use the *Fine Step* button to see how AC-3 goes through its three basic steps (selecting an arc, testing it for consistency, removing domain values to make the arc consistent); (ii) automatically fine step through the completion of the problem (*Auto Arc Consistency* button); (iii) pause auto arc consistency (*Stop* button); (iv) select a variable to split on, and specify a subset of its values for further application of AC-3 (see popup box in the left side of Fig. 1). Alternative sub-networks can be recovered by clicking on the *Backtrack* button on the toolbar. As a student steps through a problem, the message panel above the graph panel reports a description of each step. Another message panel situated below the graph panel reports the history of domain spitting decisions made by the user, i.e., which value-variable assignment has been selected at each domain splitting point.

The CSP applet currently does not provide any explicit support to help students learn at best from the mechanisms described above. Research however, shows that students may benefit from this support, since unaided exploration of interactive simulations often fails to help students learn [1]. The purpose of the study described in the next section was to collect data to investigate whether a user's attention patterns can be indicators of effective vs. non-effective learning with the CSP applet, to be eventually used in a user model that can drive personalized support when needed.

4 User Study

Fifty computer science students participated in the study. The data for 5 users was not usable due to technical issues, reducing the dataset to 45 users. All participants were required to have taken a set of courses ensuring that they would have the prerequisites to study Constraint Satisfaction Problems as discussed below. Participants were run one at the time, and each experimental session was structured as follows. First, participants were asked to study a text book chapter on Constraint Satisfaction Problems and the AC3 algorithm. This part was allotted 45 minutes and all the participants reported finishing the material in the given time. Then, participants wrote a pre-test designed to evaluate their understanding of the CSP concepts covered in the chapter they had studied. Next, participants were shown a video that explained the functionalities of the CSP applet.

The main part of the experiment was run on a Pentium 4, 3.2GHz, with 2GB of RAM with a Tobii T120 eye-tracker as the main display. Tobii T120 is a remote eye-tracker embedded in a 17" display, providing unobtrusive eye-tracking (as opposed to what head-mounted devices do). In addition to the user's gaze data, Tobii also records video data of the user's face. After undergoing a calibration phase for the eye-tracker, the participants started working with the applet to solve two CSP problems: first an easier problem involving 3 variables, 3 constraints and at most 2 domain splitting actions to find its unique answer; next, a more difficult problem involving 5 variables, 7 constraints and a minimum of 5 domain splitting actions to find its two solutions. Participants were instructed to find both of these solutions. All relevant instructions for this phase were provided on a written instruction sheet. No time limit was given for this phase, which lasted on average 16.7 (SD = 9.0) minutes. The study ended with a post-test analogous to the pre-test.

5 Data Preparation and Preprocessing

Eye-tracking data can be rather noisy when collected with eye-trackers that, like the Tobii T120, do not constrain the user's head movements [22]. In this section, we briefly explain the process we used to deal with two sources of noise in our dataset. This validation process is crucial to ensure that the data reliably reflects the attention patterns that users generated while working with the CSP applet.

The first source of noise relates to the eye-tracker collecting invalid samples while the user is looking at the screen, due to issues with calibration, excessive user movements or other user-related matters (e.g., eyeball's shape). Thus, gaze data for each user needs to be evaluated to ascertain whether there are enough valid samples to retain this user for analysis. The second source of noise relates to users looking away from the screen either for task-related reasons (e.g., looking at the instruction sheet) or due to getting distracted. During the looking-away events, the eye-tracker reports invalid samples similar to when there is a tracking error on the user gaze, even if there was no gaze to track. Thus, sequences of invalid samples due to looking-away events must be removed before starting the validation process of actual user's gaze samples. Looking-away events were automatically detected when the user gaze moved out of the screen gradually, by calculating the trajectory of fixations heading outside the screen. Automatic detection, however, is not possible when the user's gaze moves away from the screen suddenly. These events were manually identified by an investigator using videos of the user recorded during the study.

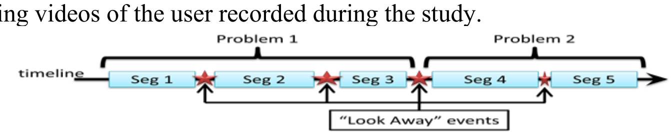

Fig. 2. A sample timeline showing segments and "look away" events

Detection of looking-away events resulted in the partitioning of the remaining gaze samples into sequences occurring between two such events (*segments* from now on, see Fig. 2). The next step was to analyze the validity of these gaze segments. In particular, we needed to set a threshold to define, for each user in our dataset: (i) whether there are enough valid samples in the user's complete interaction, represented by the aggregation of her eye-gaze segments; (ii) if so, whether there are sufficient valid samples in each segment. This second step is to avoid situations in which a large number of the invalid samples in an overall valid interaction are concentrated in few segments, making the gaze data in these segments unreliable.

We determined the threshold by plotting the percentage of segments that get discarded for different threshold values. The threshold value of 85% was selected, because it is where the percentage of discarded segments starts to rise sharply (Fig. 3). Fig. 4 shows the percentage of samples left after discarding the invalid segments based on the 85% threshold. For all users except one, more than 90 percent of the samples were kept. The average duration of each user's interaction with the CSP applet only changed from 16.7 (SD = 9.0) to 16.3 (SD = 8.8) minutes. Next, we will explain the eye gaze features calculated for each user.

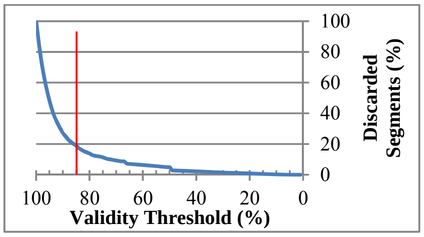

Fig. 3. Percentage of segments discarded for different threshold values

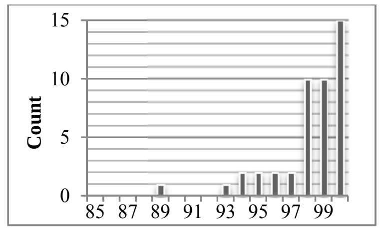

Fig. 4. Histogram of users with different percentage of segments left after removing the invalid segments

6 Eye gaze features

Table 1. Description of basic eye tracking measures

Measure	Description
Fixation rate	Rate of eye fixations per milliseconds
Number of Fixations	Number of eye fixations detected during an interval of interest
Fixation Duration	Time duration of an individual fixation
Saccade Length	Distance between the two fixations delimiting the saccade (d in Fig. 5)
Relative Saccade Angles	The angle between the two consecutive saccades (e.g., angle y in Fig. 5)
Absolute Saccade Angles	The angle between a saccade and the horizontal (e.g., angle x in Fig. 5)

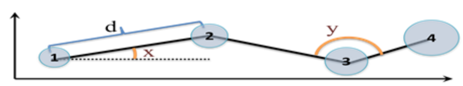

Fig. 5. Saccade based eye measures

An eye-tracker provides eye-gaze information in terms of *fixations* (i.e., maintaining eye-gaze at one point on the screen) and *saccades* (i.e., a quick movement of gaze from one fixation point to another), which are analyzed to derive a viewer's attention patterns. As mentioned in the related work section, previous research on using gaze information for assessing learning in IS relied on tracking one specific attention pattern, predefined a priori [3, 4]. In contrast, in our analysis we use a large set of basic eye-tracking features, described by [22] as the building blocks for comprehensive eye-data processing. These features are built by calculating a variety of statistics upon the basic eye-tracking measures described in Table 1. Of these measures, *Fixation rate*, *Number of Fixations* and *Fixation Duration* are widely used (e.g., [11, 16–18]); we also included *Saccade Length* (e.g., distance *d* in Fig. 5), *Relative Saccades Angle* (e.g., angle *y* in Fig. 5) and *Absolute Saccade Angle* (e.g., angle *x* in Fig. 5), as suggested in [22], because these measures are useful to summarize trends in user attention patterns within a specific interaction window (e.g., if the user's gaze seems to follow a planned sequence as opposed to being scattered). Statistics such as sum, average and standard deviation can be calculated over these measures with respect to: (i) the full CSP applet window, to get a sense of a user's overall attention; (ii) specific areas of interest (AOI from now on) identifying parts of the interface that are of specific relevance for understanding a user's attention processes.

We defined four AOIs for our analysis, corresponding to the areas that provide conceptually different functionalities in the CSP applet. Rectangles corresponding to these AOIs are shown in Fig. 1. One AOI covers the region of the applet toolbar that includes action buttons (*toolbar* AOI); one covers the main graph panel (*graph* AOI); one covers the part of the top panel where the description of every step of the algorithm is displayed (*top* AOI); the last covers the part of the bottom panel that displays domain splitting information (*bottom* AOI).

Table 2. Derived eye tracking features for the full CSP applet window

Fixation rate
Total Number of Fixations
Sum of Fixation Durations
Mean and Std. Dev. of Fixation Durations
Mean and Std. Dev. of Saccade Length
Mean and Std. Dev. of Relative Saccade Angles
Mean and Std. Dev. of Absolute Saccade Angle

Table 3. Derived eye tracking features for each of the four AOIs

Fixation rate
Total Number of Fixations
Proportion of Total Number of Fixations
Mean Fixation Durations
Proportion of Total of Fixation Durations
Highest Fixation Duration
Number of Transitions between pairs of AOIs
Proportion of Transitions between pairs of AOIs

Table 2 shows the set of gaze features calculated from the eye movement measures in Table 1 over the full CSP applet window. Table 3 shows the set of features calculated for each of the four AOIs. As the table shows, the two sets are different. For the AOIs, we added features that measure a user's relative attention to each AOI: *Proportion of Total Number of Fixations* and *Proportion of Total Fixation Duration* give the percentage of the overall number of fixations and fixation time, respectively, that were spent in each AOI. We also added features that quantify gaze transitions between different pairs of AOIs [22] (including from an AOI to itself), as a way to capture the dynamics of a user's attention patterns. Transitions are represented both in terms of total number (*Number of Transitions between pairs of AOIs* in *Table* 3), as well as a proportion of all transitions (*Proportion of Transitions between pairs of AOIs*). Adding the aforementioned AOI-specific features substantially increases the overall number of features considered. In order to keep this number manageable, for the AOIs we did not compute saccade-based features, which are less commonly used than fixation-based features in eye-tracking research. In total, we included 67 features, 11 for the full CSP window, and 56 for AOI.

7 Classifying learners based on gaze data

To ascertain whether a user's success in learning with the CSP applet can be identified using his/her eye movement data, we built two different classifiers using this data. The first classifier uses the eye-tracking features described in section 6, computed over the complete interaction of a student with the CSP applet (*Whole Interaction dataset* from now on). Thus, this classifier relies on features that describe a user's overall attention patterns during the study task. The second classifier uses features that reflect the *changes* in the user's attention patterns between solving the first and the second problem (*Interaction Evolution* dataset). Each classifier is built to discriminate between two classes of users, High Achievers (HA) and Low Achievers (LA), defined based on the median split of Proportional Learning Gain (PLG) from pre-test to post-test. PLG is calculated using equation 1.

$$PLG = \frac{PostestScore - PretestScore}{MaxPossibleScore - PretestScore} \times 100\,, \qquad (1)$$

The median PLG is 45.83, resulting 23 LA and 22 HA. The average PLG overall is 41.25 (SD = 35.31). It is 68.27 (SD = 12.39) for the HA and 15.40 (SD = 30.29) for the LA group. In the next two sections, we discuss each classifier and its performance results.

7.1 User classification based on the *Whole Interaction* dataset

This classifier aims to predict a user's class label (HA vs. LA) using the *Whole Interaction* dataset, i.e., the 67 features that describe a user's overall attention patterns during the study task. We tried 6 different classifiers from the different classifier types available in the Weka data mining toolkit (Decision Tree based, Support Vector Machine, Linear Ridge Regression, Binary Logistic Regression and Multilayer Perceptron), using feature-selection and leave-one-out cross-validation. The classifier with the highest accuracy is a Decision Tree based Classifier generated using the C4.5 algorithm (DTC from now on). The accuracy of the DTC for each class and overall is shown in Fig. 6 (we will discuss the RRC classifier shown in the picture in the next section). The figure also reports the accuracy of a baseline classifier that always selects the most likely class (LA in our case), thus failing in all cases of the other class. The DTC achieves 71.1% overall accuracy, which is significantly higher than baseline (χ^2 (1) = 16.01, $p < 0.001$). DTC does not have very high accuracy (63.3%) for the HA class, but achieves 78.3% accuracy for the LA class, showing that it can recognize those students who may need help to better learn with the CSP applet. These results clearly show the potential of using eye movement data as a source of information to classify learning performance.

The structure of the decision tree, shown in Fig. 7, indicates which features contribute to discriminate between high and low achievers with the CSP applet. In Fig. 7, each node represents a feature with a partitioning value that DTC uses to separate users into two groups, one with values higher than the partitioning value (right branch of the node) and one with values that are lower (left branch of that node). The numbers next to each branch specify how many HA and LA datapoints are found in the corresponding subgroup. The leaves of the tree assign a class label to all the users in the corresponding subgroups. For simplicity, we will only look at the top three nodes of the tree The partition of datapoints created by the root node (*prop_Total_fixations_Bottom* in Fig. 7) shows that LA tend to have a higher proportion of fixations in the Bottom AOI than HA. The Bottom panel is only used for displaying domain splitting information, which becomes relevant only when a CSP graph has been made arch consistent. Thus, showing a higher proportion of fixations in this panel may be an indication that LA are looking at irrelevant information due to confusion or not knowing which action to perform next. Interestingly, the partition created by the left child of the root node, (*Bottom_fixation_rate* in Fig. 7), shows that most HA in this branch have higher fixation rate in the Bottom AOI, suggesting that, although HA look at the bottom panel less often than LA, when they do look they seem to pay more attention.

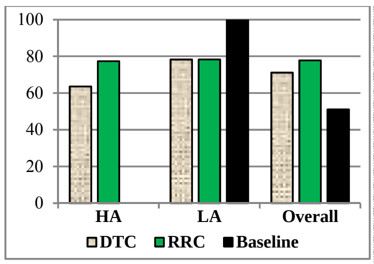

Fig. 6. The classifier performance in each class and overall

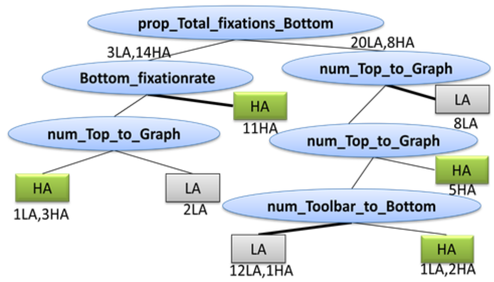

Fig. 7. The Decision Tree Classifier

Thus, it appears that HA know the value of the information displayed in the Bottom panel and only use it when it is relevant. These results are consistent with the findings in problem solving research indicating that successful problem solvers show selective attention to relevant information, while unsuccessful problem solvers tend to get distracted by irrelevant information [18,19]. The right child of the root node (*num_Top_to_Graph*) generates a partition based on the number of transitions between Top and Graph AOIs, and it appears twice in the right subtree. At the second level of the tree it identifies a subgroup of LA who show a high number of transitions, while at the third level it identifies a subgroup of HA who show this pattern. Since the Top panel is used for displaying the outcome of any action related to stepping through the AC-3 algorithm, for the HA sub-group the high number of transitions between the two AOIs could be a sign of focused attention to the workings of the algorithm, which helped them learn from the interaction. For LA, on the other hand, the high number of transitions from top to graph panel, may be another indication of confusion, for instance if they happened in a few clusters as opposed to regularly after every action. A more detailed gaze-data analysis at the level of user actions would be necessary to gain further insights on what is happening with this group of LA.

7.2 User classification based on the *Interaction Evolution* dataset.

For our second classifier, we wanted to explore whether *changes* in the user's attention patterns from the first to the second problem (P1 and P2 from now on) could be predictors of learning. We calculated the 67 features described in section 6 for each of the two periods during which the user was interacting with P1 and P2, respectively, and then we compared the values obtained to verify whether any difference actually existed. A battery of paired t-tests on the values for each feature in P1 and P2 resulted in 44 features that are significantly different, indicating that users' attention patterns do change to some extent when solving these two problems.

We used these 44 features, with values assigned to be the difference between their corresponding values for P1 and P2 (*Interaction Evolution* dataset), to train a second classifier of LA vs. HA. As with the previous dataset, we tried 6 different classifiers, with the classifier using Ridge Regression (RRC from now on), obtaining the highest accuracy. The RRC's performance of for each class and overall, is shown in Fig. 6. The RRC achieves 77.8% accuracy overall, which is significantly higher than baseline (χ^2 (1) = 29.17, p < 0.001). The overall accuracy of the RRC is also higher than DTC's, but the difference is not significant. It should be noted, however, the RRC achieves significantly higher accuracy than DTC on the HA class (χ^2 (1) = 8.408, p = 0.004), thus yielding a much better balance between the accuracy of the HA and LA classes (77.3% and 78.3% respectively). These results indicate that changes in a user's gaze patterns as the interaction with the CSP proceeds and the user attempts more difficult problems can be even more informative than overall attention patterns for predicting learning with this IS.

Table 4. Regression features with non-zero coefficients

Feature	Change (P1 to P2)	Stand. Coef.	Feature	Change (P1 to P2)	Stand. Coef.
Bottom_num_fixations	Increase	1.3837	Total_num_fixations	Increase	-0.2487
num_Toolbar_to_Toolbar	Increase	0.6519	Top_longest_fixation	Decrease	-0.3498
prop_Graph_to_Graph	Increase	0.5857	num_Graph_to_Toolbar	Increase	-0.4110
num_Toolbar_to_Top	Decrease	0.3441	num_Graph_to_Top	Decrease	-0.5729
Top_fixationrate	Increase	0.3177	num_Graph_to_Bottom	Increase	-0.8279
			SD_absolute_saccade_angles	Decrease	-0.8783

As we did with the classifier described in the previous section, we now discuss some of the features that contribute to distinguish LA from HA in our second classifier. The complete set of features with non-zero coefficients in the regression model is shown in Table 4. The table also reports, for each feature, the direction of change between P1 and P2, as well as its standardized coefficient. Here we discuss some of the most intuitive features with high impact in the regression (as measured by the standardized coefficients). The strongest positive indicator of learning in Table 4 is an increase in the number of fixations on the Bottom AOI (*Bottom_num_fixations*) from P1 to P2. As discussed in the previous section, the Bottom panel shows domain splitting information. Domain splitting is required more often in P2 than in P1, so the trend found shows that HA change the amount of attention they devote to the bottom panel accordingly while LA fail to do so. Table 4 also shows that one of the highest *negative* predictors of learning is an increase in the number of transitions between Graph and Bottom panels from P1 to P2 (*num_Graph_to_Bottom*), i.e., the number of transitions from the Graph to the Bottom panel increases from P1 to P2 for LA. However, except for the times when domain splitting is performed, there is no new information presented in the bottom panel, so these results could be further evidence that LA tend to look at the bottom panel when it is not relevant, as indicated by the results discussed in the previous section. Another strong negative indicator of learning is an increase in the number of transitions from the Graph to the Toolbar AOI (*num_Graph_to_Toolbar*). As users gain more experience with the interface, it is expected that they would shift their attention less often between the Graph and

Toolbar. Thus, an increase in number of transitions can be interpreted as a sign that, during the interaction with the second more complex problem, LA were more often at loss about what action to perform next and looked frequently at the Toolbar for inspiration. In contrast, Table 4 shows that the number of gaze shifts staying in the Toolbar buttons area (*num_Toolbar_to_Toolbar*) is positively associated with learning. This feature shows the process of making decisions about which action to perform next. A likely reason for HA to go back and forth between the items on the toolbar more often during P2 than during P1 is that more actions are relevant at the same time for solving P2 (e.g., continuing to step through the solution of a sub-case resulting from domain splitting vs. deciding to backtrack to an alternative sub-case because the current one does not look promising) and HA are carefully considering the available options.

To summarize, the good classification performance on the Interaction Evolution dataset shows that taking into account temporal information on how attention patterns evolve during logical units of interactions (e.g., different problems in our case) can further improve the potential of eye-tracking data for user modeling for IS.

8 Conclusion and Future Work

We presented results on the value of eye-tracking data to assess user learning with an interactive simulation for constraint satisfaction problems (the CSP applet). We showed that a classifier using solely information on a user's overall attention patterns during a complete session with the CSP applet can already achieve good accuracy in distinguishing students who learned well from students who did not. Adding information on how students' attention patterns changed while solving two different problems of increasing difficulty further improved classification accuracy.

Our next step will be to leverage the results discussed here and the results obtained with a previous classifier that relied only on interaction logs to identify high vs. low learners [2] to build a classifier user model for the CSP applet that integrates both sources of information. We also plan to investigate techniques to further exploit the temporal nature of attention patterns, such as clustering of scanpaths (sequences of consecutive saccades). Finally, we are investigating how to design adaptive interventions for the CSP applet, to be provided to users when the user model detects that they are not learning well from the interaction.

References

1. Shute, V.J.: A comparison of learning environments: All that glitters. Computers as cognitive tools. pp. 47-73. Hillsdale, NJ, England: Lawrence Erlbaum Associates, Inc (1993).
2. Kardan, S., Conati, C.: A Framework for Capturing Distinguishing User Interaction Behaviours in Novel Interfaces. Proc. of the 4th Int. Conf. on Educational Data Mining. pp. 159-168. , Eindhoven, the Netherlands (2011).
3. Conati, C., Merten, C.: Eye-tracking for user modeling in exploratory learning environments: An empirical evaluation. Knowledge-Based Systems. 20, 557-574 (2007).
4. Amershi, S.A., Conati, C.: Combining Unsupervised and Supervised Machine Learning to Build User Models for Exploratory Learning Environments. J. of Educational Data Mining. 18-71 (2009).

5. Keith, R.: Eye movements and cognitive processes in reading, visual search, and scene perception. Eye Movement Research Mechanisms, Processes, and Applications. pp. 3-22. North-Holland (1995).
6. Rayner, K.: Eye movements in reading and information processing: 20 years of research. Psychological Bulletin;Psychological Bulletin. 124, 372-422 (1998).
7. Rong-Fuh, D.: Examining the validity of the Needleman–Wunsch algorithm in identifying decision strategy with eye-movement data. Decision Support Systems. 49, 396-403 (2010).
8. Simola, J., Salojärvi, J., Kojo, I.: Using hidden Markov model to uncover processing states from eye movements in information search tasks. Cognitive Systems Research. 9, 237–251 (2008).
9. Courtemanche, F., Aïmeur, E., Dufresne, A., Najjar, M., Mpondo, F.: Activity recognition using eye-gaze movements and traditional interactions. Interacting with Computers. 23, 202-213 (2011).
10. Loboda, T.D., Brusilovsky, P., Brunstein, J.: Inferring word relevance from eye-movements of readers. Proc. of the 16th int. conf. on Intelligent user interfaces. pp. 175–184. ACM, New York, NY, USA (2011).
11. Loboda, T.D., Brusilovsky, P.: User-adaptive explanatory program visualization: evaluation and insights from eye movements. User Modeling and User-Adapted Interaction. 20, 191–226 (2010).
12. Conati, C., Carenini, G., Harati, M., Tocker, D., Fitzgerald, N., Flagg, A.: User-Adaptive Visualizations: Can Gaze Data Tell Us When a User Needs Them? Workshops at the 25th AAAI Conf. on Artificial Intelligence (2011).
13. Iqbal, S.T., Adamczyk, P.D., Zheng, X.S., Bailey, B.P.: Towards an index of opportunity: understanding changes in mental workload during task execution. Proc. of the SIGCHI conf. on Human factors in computing systems. pp. 311–320. ACM, New York, NY, USA (2005).
14. Muldner, K., Christopherson, R., Atkinson, R., Burleson, W.: Investigating the Utility of Eye-Tracking Information on Affect and Reasoning for User Modeling. Proc. of the 17th Int. Conf. on User Modeling, Adaptation, and Personalization. pp. 138–149. Springer-Verlag, Trento, Italy (2009).
15. Knoepfle, D.T., Wang, J.T., Camerer, C.F.: Studying Learning in Games Using Eye-tracking. J. of the European Economic Association. 7, 388-398 (2009).
16. Hegarty, M., Mayer, R.E., Monk, C.A.: Comprehension of Arithmetic Word Problems: A Comparison of Successful and Unsuccessful Problem Solvers. J. of Educational Psychology. 87, 18-32 (1995).
17. Canham, M., Hegarty, M.: Effects of knowledge and display design on comprehension of complex graphics. Learning and Instruction. 20, 155-166 (2010).
18. Jarodzka, H., Scheiter, K., Gerjets, P., van Gog, T.: In the eyes of the beholder: How experts and novices interpret dynamic stimuli. Learning and Instruction. 20, 146-154 (2010).
19. Tsai, M.-J., Hou, H.-T., Lai, M.-L., Liu, W.-Y., Yang, F.-Y.: Visual attention for solving multiple-choice science problem: An eye-tracking analysis. Computers & Education. 58, 375-385 (2012).
20. Eivazi, S., Bednarik, R.: Predicting Problem-Solving Behavior and Expertise Levels from Visual Attention Data. The 2nd workshop on the Eye Gaze in Intelligent Human Machine Interaction. pp. 9-16. , Palo Alto, California, USA (2011).
21. Amershi, S., Carenini, G., Conati, C., Mackworth, A.K., Poole, D.: Pedagogy and usability in interactive algorithm visualizations: Designing and evaluating CIspace. Interacting with Computers. 20, 64-96 (2008).
22. Goldberg, J.H., Helfman, J.I.: Comparing Information Graphics: A Critical Look at Eye Tracking. Presented at the BELIV'10 , Atlanta, GA, USA (2010).