Learning and Performing by Exploration: Label Quality Measured by Latent Semantic Analysis

Rodolfo Soto
Institute of Cognitive Science
University of Colorado
Boulder, CO 80309-0344, USA
+1 (303) 492-4574
Rodolfo.Soto@Colorado.edu

ABSTRACT
Models of learning and performing by exploration assume that the *semantic distance* between task descriptions and screen labels controls in part the users' search strategies. Nevertheless, none of the models has an objective way to compute semantic distance. In this study, participants performed twelve tasks by exploration and were tested for recall after a 1-week delay. Latent Semantic Analysis was used to compute the semantic similarity between the task descriptions and the labels in the application's menu system. When the labels were close in the semantic space to the task descriptions, subjects performed the tasks faster. LSA could be incorporated into any of the current models, and it could be used to automate the evaluation of computer applications for ease of learning and performing by exploration.

Keywords
Learning by exploration, label-following strategy, cognitive models, semantic similarity, latent semantic analysis

INTRODUCTION
In the interaction of humans and computers, words are the link between users' goals and the actions required to accomplish those goals. For command-based environments, such as UNIX, users must memorize sets of keywords that they type to interact with the system. Likewise, in display-based environments, such as Mac OS or Win 95, users must point at and click on display objects labeled by words (e.g., menu items, tool bars, or dialog boxes). The right choice of words can successfully lead users through novel or rarely used applications such as library databases [1], telephone menu systems [2], or graphics applications [3].

The study described in this paper provides empirical evidence supporting the hypothesis that users act on those interface labels that are semantically related to their task goal. Additionally, it shows that Latent Semantic Analysis (LSA) is a reliable technique to compute the semantic distance between the interface labels and the users' task goals, and that semantic distance predicts ease of discovery and recall of correct action sequences.

Learning and Performing by Exploration in Display-Based Computer Applications
Polson and Lewis [4] analyzed the exploratory behavior of novice users who have a goal in mind, and who have some experience with a particular application or operating system. In this situation, users engage in search through a problem space [5] composed of the multiple application states. Users employ some domain-independent method to guide their search without having to know much about the application. This kind of method is called a weak method, and means-ends analysis is probably its most used instance.

Two variants of means-ends analysis are frequently observed in novice exploration: hill climbing, and back chaining. In both cases, for each state, one action is chosen among the available alternatives using "perceptual similarity as a measure of distance" [4, p. 205]. Engelbeck [6] observed that during exploration novice users tend to explore those menu labels that share one or more words with the experimenter-supplied description of the tasks (or with the user's goal). Muncher [7] also found this behavior in novice users learning Lotus 1-2-3. This heuristic has been called the *label-following strategy* [4], and it can be classified as a hill-climbing technique that uses semantics to compute distance. Considerable evidence confirms that label following is an effective method for discovering the solution to novel computer tasks [3,4,8].

Users combine the label-following strategy with other searching techniques such as depth-first or breadth-first exploration. Rieman analyzed searching in menu systems and concluded that an effective search algorithm would be a combination of label-following and a hybrid between depth- and breadth-first search called depth-first iterative deepening (DFID). Rieman suggested that this combination of label following and DFID should be called "guided DFID", or gDFID. Rather than using brute force, as in pure DFID, gDFID "heuristically limits its search to items *semantically* [italics added] related to the current task" [9, p. 747].

Models Based on Cognitive Architectures
Several models have been proposed to simulate the exploration and recall of the action sequences needed to performed novel tasks using a display-based computer application. All these models emphasize the use of label

following as a means to acquire knowledge about the interface. Independently of the details in each model, they all use some version of semantic distance as an evaluation function to direct the user's search.

SOAR models

The Task-Action Learning (TAL) model [10] simulates users who are familiar with basic operations of the mouse and keyboard, but unfamiliar with a particular menu structure, object labels, and actions required to accomplish a task. TAL emphasizes the role of semantics, since it assumes that users analyze the instructions, the semantic features of tasks, and the labels of the screen, hoping to find a link between them. "Interpreting instructions involves matching the task description to the instruction using a rule base of *semantic* links [italics added] between features of the task and items on the display" [10, p. 313]. The semantic associations are implemented via a function that takes semantic features of the tasks (defined by the experimenter) and lexical items as parameters, and returns a Boolean expression indicating semantic matching.

The IDXL model [9] simulates learning by exploration. It implements gDFID searching and assumes that the user's attention mechanism focuses on one object at a time. The model is supplied with a task description in working memory and it has knowledge about Macintosh conventions and about the correct and legal actions that can be taken in the menu system. Scanning is the main operator used during exploration. It allows the visual focus to shift right, left, up, down, and to jump from place to place. Another operator comprehends the items that have been under attention and "may note that the scanned item is a label that *matches* [italics added] some key word in the task" [9, p. 758]. The model considers that a direct match costs less than an indirect match (e.g., a synonym). Thus, it tries those items that have a direct match with the experimenter-supplied task description before trying anything else. All the knowledge about what words are synonyms has to be explicitly given to the model.

A Comprehension-Based Model of Exploration

The LICAI+ model [11,12] simulates the user's comprehension of task instructions and hints, the generation of goals, and the use of these goals to discover correct actions by exploration. This model is based on the CI architecture [13], that was originally developed as a model for text comprehension and extended to action planning by Mannes and Kintsch [14]. LICAI+ predicts that successful exploration and recall requires semantic matching between the goal representation and the labels on the display objects.

The CI architecture combines propositional knowledge with connectionist spreading activation mechanisms. CI assumes that two propositions are related if they share one or more arguments. For LICAI+, this means that a menu label and the description of a task (or a hint, or a piece of instruction, or the user's goal) are related if there is concept overlap between them. The semantic distance notion of these models is very crude. If the labels and the task descriptions do not share words, additional knowledge can be provided by long-term memory to establish a link.

Other Models

Some models that are not based on cognitive architectures have been developed to explain learning by exploration. One of the most interesting is the Ayn model [15] that simulates a user learning how to use Microsoft's Word menu system through exploration. Ayn makes decisions about one menu at a time using four types of knowledge: (1) semantic knowledge to avoid irrelevant menu labels; (2) failure detection to avoid dead ends; (3) recognition to avoid exploring the same path more than once; and (4) task-control knowledge to remember the outcome of previous trials. In the Ayn model, semantic knowledge is explicitly given to the model as lists of words that may be related to particular tasks.

In summary, all the available models of learning by exploration share the same intuition about the role of semantic distance: users tend to act on objects with labels that "seem" to be semantically close to their task goals. Additionally, some of the models explain how the label-following strategy is frequently combined with other exploratory mechanisms. Although all the simulations confirm the reliability of the label-following strategy, they do not include an objective measure of semantic distance. This paper proposes that a mathematical model of semantics, such as Latent Semantic Analysis, is a good candidate for computing semantic distance.

Latent Semantic Analysis (LSA) As A Model For Semantics

LSA is both a model and a technique to extract semantic information from large bodies of text. LSA was originally conceived as an information retrieval technique [16] that makes use of statistical procedures to capture the similarity of words and documents in a high-dimensional space [17,18]. Once LSA is trained on a corpus of data (consisting of several thousands of documents), it is able to compute similarity estimates that go beyond simple co-occurrence or contiguity frequencies. Although LSA does not have any knowledge about grammar, morphology, or syntax, it mimics humans' use of language in several areas ranging from the rates of vocabulary acquisition by schoolchildren, to word and passage priming effects, to evaluating the performance of students on essay tests.

The collection of documents used to train LSA is arranged in a matrix where the columns correspond to the documents, and the rows correspond to unique word types. An entry in the matrix indicates the number of times the word appears in the document. Using a linear algebra technique called singular value decomposition it is possible to represent each document and each word as a vector of high dimensionality (e.g., 400) that captures the underlying relations of the words and their contexts. To determine how similar two words are, LSA computes the cosine between the vectors that represent the words. A cosine, like a correlation coefficient, ranges between –1 and 1, where 1 represents a perfect match (i.e., the same word), and 0 represents no relationship between the words.

The available evidence suggests that LSA is a plausible theory of learning, memory, and knowledge [18]. All the tests that LSA has performed successfully have been solved using semantic similarity as the main predictor of fitness. For instance, LSA does well on the synonym portion of the Test of English as a Foreign Language (Educational Testing Service) [19,20]. It computes the semantic distance between the stem word in each item and each of the four alternatives, choosing the one with the highest cosine. Using this method, LSA performed virtually identically to the average of a large sample of non-English speaking students. Since the HCI literature stresses the role of semantics as a measure of distance during hill-climbing-like strategies, LSA should be able to account for the "good-labels effect" observed during exploration. In other words, LSA could be extended to action planning.

Since LSA learns about language exclusively from the training texts, it is very important to choose the right corpus for the specific situation to be modeled. Several corpora have been used to train LSA. One of the most versatile is the TASA (Touchstone Applied Science Associates, Inc.) corpus. This group of documents uses a variety of text sources, such as newspaper articles and novels that represent the kind of material students read during the school years. The TASA corpus is broken into grade levels, from third grade to college, using a readability score (DRP-Degrees of Reading Power Scale) that is assigned to each of its 37,651 documents. TASA is a good training corpus to model naïve or inexperienced users, especially because it is assumed that these kinds of users are forced to adapt their "everyday" knowledge about common words to the new context that is imposed by a computer application [10].

Predictions

Hypothesis 1

The main limitation of both the theoretical and the empirical work on the label-following strategy is the lack of a well-defined measure for semantic distance. In most cases, semantic relationships are established exclusively via some form of literal word overlap. Hence, the only well-defined distance metric is, in LSA terms, a cosine of 1 (i.e., an exact match). Likewise, "distance" is defined by the number of shared words or by the number of links between hand-coded "propositions" that describe the object labels, the task descriptions, and any other knowledge related to them. Unless informal intuitive estimates of semantic distance are included in the models, it is difficult to study situations where there are intermediate degrees of semantic matching.

For the present study, an *add-in* was written for Microsoft Excel to change the application's interface and to experimentally manipulate the semantic distance between the screen labels and the task descriptions. It is predicted that the probability of discovering and recalling a correct action sequence increases as the semantic distance between the labels on the menus and the task description decreases. LSA cosines are used to estimate the semantic distance.

Hypothesis 2

Previous studies have examined different configurations of matching and non-matching steps in an action sequence. For instance, it has been found that the probability of discovering a correct action sequence that contains two non-matching steps, one after the other, is very low. Whereas the probability of discovering an action sequence that contains two matching steps, one after the other, is very high. Finally, the probability of discovering a sequence with a non-matching step followed by a matching step is somewhere in the middle between the previous two probabilities [12].

Examination of commercial graphics packages reveals that it is not common to find tasks with a sequence of actions where a matching step is followed by a non-matching step. In the study described here, four possible combinations of matching and non-matching pairs of steps are manipulated. It is predicted that tasks where one matching step follows another matching step are the fastest to perform, whereas tasks where a non-matching step follows another non-matching step are the slowest. The other two cases (i.e., a non-matching step followed by a matching step, or vice versa) are in the middle, and there are no significant differences between them.

Hypothesis 3

In a pilot experiment for this study, subjects learned direct manipulation tasks (those that required action directly on the object, without any label involved) either by exploration or by explicit instruction. During exploration, subjects received hints if they could not figure out the solution of the task after 60 s. The surprising finding was that these tasks were recalled much better if they were explicitly instructed as opposed to learned by exploration. A possible explanation for this result is that the memories of unsuccessful trials interfere with the storage of the correct solution (i.e., with the hint given by the experimenter).

In the study described here, subjects learned one set of tasks by exploration, and another set by explicit instruction. Although none of the tasks involved direct manipulation, it is predicted that there will be, at most, no significant recall differences between the groups. If the tasks are recalled better when they are explicitly instructed, it could be argued that the unsuccessful exploration might be interfering with the storage of a hint. If not, it could be argued that this phenomenon, for some reason, only applies to tasks where no label is involved, and that unsuccessful exploration does not interfere with the storage of hints.

METHODS

Participants

Fifty-five undergraduate students participated in the experiment. Twenty-eight received class credit and twenty-seven received $10 for their participation. The data from seven participants, four from the group that received class credit and three from the group that received $10 were discarded: two of them were not able to follow the instructions correctly, and in the five other cases, technical errors invalidated the results. The remaining forty-eight

participants had at least four years of experience with either the Macintosh or the IBM-PC computer, or both. The group that received class credit had significantly more experience than the other group (on average 5.8 vs. 4.3 years, $F(1,47) = 5.21$, $p < .03$). However, the groups did not differ significantly in their years of experience with Microsoft Word, Microsoft Excel (without creating graphics), Mac Draw, and WWW Browsers. Likewise, they did not differ significantly in the number of graphs they had created by hand in their life. None of the participants had experience with graphics applications such as Cricket Graph or with the graphics capabilities of Microsoft Excel.

Materials

Twelve computer tasks were designed manipulating the semantic match between the labels of the menu system and the description of the tasks. Microsoft Excel was used to administer the tasks, running in a Macintosh Centris 650 with 16 MB in RAM, 500 MB in hard disk, and a page-size grayscale monitor. An Excel *add-in* was developed to reconfigure Excel's interface. This made it possible to have a fully-functional graphics application in which the tasks had the features required by the experiment, and which guaranteed that the application was novel for the participants.

An S-VHS camera and a clip-on microphone were used to record the computer screen and the participants' voice. Each participant received a package containing an informed consent form, a blue pen, and a notebook with the instructions and the description of tasks.

Tasks

There were four warm-up and eight experimental tasks. All consisted of editing a bar graph, using a graphics application (see [21], for a detailed description of the tasks). Participants received detailed descriptions of the tasks, but no information about how to perform them (Table 1 shows the task descriptions). The eight experimental tasks followed the same structure of five steps. (1) Choose a top-level menu item. (2) Choose a submenu item. (3) Choose a sub-submenu item. (4) Click on a radio button or check box (in a dialog box). (5) Click on a button labeled "Ok" to close the dialog and end the task.

Table 1. Description of the experimental tasks

Task 1	Change the graph type to column
Task 2	Apply the default format to the graph
Task 3	Hide the graph title
Task 4	Add a third dimension to the graph
Task 5	Change the graph font to bold
Task 6	Delete the values from the bar graph
Task 7	Change the graph background color to green
Task 8	Apply a logarithmic scale to the graph axes

Four levels of matching were used for the labels in the second and third steps (submenu items): good match (G), and three degrees of bad match (B_1, B_2, B_3). LSA cosines were computed between the description of the tasks and the menu labels using the TASA space. To choose the menu labels, the closest 1000 terms to the description of the tasks were computed. From this pool, words were selected and two-word phrases were created for each menu item. In all cases cosines between the two-word phrases and the task descriptions followed the relationship $G > B_1 > B_2 > B_3$. G exhibited the best semantic match, and B_3 the worst. On average, G labels had a cosine of .67 (SD = .17), B_1 labels a cosine of .25 (SD = .08), B_2 labels a cosine of .15 (SD = .02), and B_3 labels a cosine of -.05 (SD = .01) with the task descriptions. For all the experimental tasks, the fourth step had a label matching in the range of the G level, whereas the first step had a label matching in a range between B_1 and B_2.

Design

Each of the eight experimental tasks could be presented in one out of four configurations, depending on the semantic matching for the second and third steps: C_1 = (G,G); C_2 = (G,Bi); C_3 = (Bi,G); C_4 = (Bi, Bi), where Bi represents one of the three degrees of bad semantic matching, and G represents a good semantic match. For instance, a task with a configuration C_3 had a bad semantic match in the second step, and a good semantic match in the third step. The tasks were divided in two sets of four tasks: set A and set B. Each set was equivalent to the other in number of steps and in the degree of semantic match between the task descriptions and the labels of the menus. Half of the participants explored the tasks in set A, and they were explicitly instructed in the tasks in set B. The other half explored the tasks in set B, and they were explicitly instructed in the tasks in set A. The presentation of the tasks were counterbalanced using a greco-latin square design for the eight tasks and the four configurations.

Procedure

The experiment consisted of two 30-minute sessions: a training session followed by recall 7 days later. Participants were interviewed individually, their responses were recorded, and the computer screen was videotaped. In the training session, participants read and signed a consent form and received a written version of the instructions. During the first 3 minutes, a verbal protocol practice task was administered consisting of a "think aloud" description of the participant parent's house, as recommended by [22]. During both sessions, participants had to think aloud while performing the experiment. The experimenter reminded the participants that they had to think aloud if they remained silent for more than 15 s.

After signing the consent form, participants opened the notebook and read the instructions. At this point, the experimenter answered any question the participants had. Participants were informed that they would be explicitly instructed in 4 of the 12 tasks, and that they would have to pay close attention to what they did because they had to repeat it in one week. When a task was explicitly instructed, the experimenter gave step-by-step instructions on how to perform the task. When the task was not explicitly instructed, the participant could explore the interface to "figure out" how to perform the task. During this process, users could undo or cancel any incorrect action. If after 60 s the participant did not show progress, the experimenter gave a hint that consisted in revealing the corresponding step of the sequence. The hints were the same as the ones used in the explicitly instructed version. The experimenter gave as many hints as needed in order from the first step to the last step, and allowed 60 s for exploration at each step.

For the recall session, participants had to perform the same training tasks and in the same order. During the recall session, none of the tasks was explicitly instructed, but hints were given if necessary following the same procedure used in the training session. At the end of the recall session, a survey was administered to obtain information about the participants' computer experience. After the questionnaire, the experimenter turned off the computer screen and handed them a piece of paper with the description of the tasks used during the experiment. Participants were asked to write down as many labels as they could recall from the menus and other screen objects that had to be manipulated to perform each of the tasks.

Scoring and Data Measurement

During the explored part of the training session and during the whole recall session two measures were recorded for each task step: elapsed time, and number of hints. The experimenter recorded the number of hints whereas the VCR's counter was used to measure the time per step from the videotapes of the sessions.

RESULTS

ANOVA tests were conducted for both dependent variables (time and number of hints) to determine the effect of the "design" factors. On average, no significant differences in performance were found between the group that received $10 for the experiment and the group that did not receive any payment. Likewise, no overall difference was found between the group that explored set A during training compared to the group that explored set B. Additionally, there was no effect of task and configuration order. None of these factors was included in further analyses.

Task Data

The total elapsed time for each task was computed as the sum of the elapsed time for each of the 5 steps. Likewise, the number of hints was computed as the number of steps that could not be performed in less than 60 s and, therefore, required a hint. On average, each task was performed in 87.36 s during training (SD = 15.1), and in 68.1 s during recall (SD = 20.1). This difference was significant, $F(1, 47) = 51.35$, $p < .0001$. Similarily, 0.96 hints per task were given on average during training (SD = .35), and 0.67 during recall (SD = .22). This difference was also significant, $F(1,47) = 29.12$, $p < .0001$.

Time and number of hints were collapsed over the individual tasks and over training and recall to analyze the effect of the configuration of the task ((good, good), (good, bad), (bad, good), and (bad, bad)). As predicted, the configuration (good, good) was performed much faster and required fewer hints than the other three task configurations ($F(1, 47) = 195.3$, $p < .0001$, $F(1,47) = 189.8$, $p < .0001$, for time and for hints, respectively). Likewise, the (bad, bad) configuration was performed much more slowly and required more hints than the other three task configurations ($F(1, 47) = 183.9$, $p < .0001$, $F(1,47) = 156.1$, $p < .0001$, for time and for hints, respectively). Finally, there was no significant difference between the (good, bad), and the (bad, good) configurations ($F(1,47) = 1.71$, $p = .19$, $F(1, 47) = 2.19$, $p = .14$, for time and for hints, respectively). Figure 1 shows the effect of task configuration and the effect of degree of badness on task performance time. The interaction between task configuration and degree of badness was not significant.

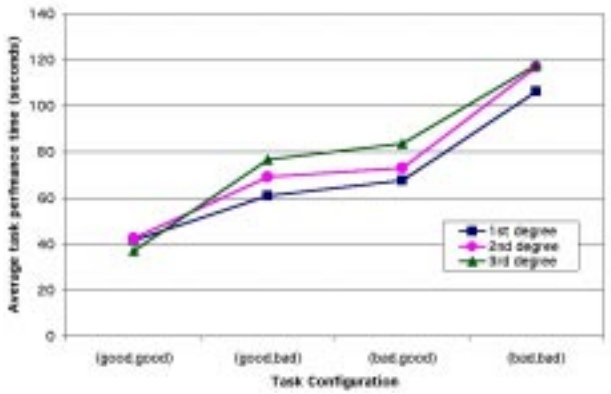

Figure 1. Effects of task configuration and degree of badness on **task** performance time. (3rd degree of badness represents the worst semantic match)

Step Data

Since the semantic distance of only the second and third steps in each task was manipulated, only the data from those two steps were used to analyze the effects of practice, semantic distance, degree of badness, and type of training. Collapsing over task configuration, sessions, and step number (second and third), the closer in the semantic space the label of the step was to the task description, the faster it was performed and the fewer hints were needed. Good steps (average LSA cosine of .67) were performed, on average, in

7.97 s (SD = 4.56) and required .05 hints (SD = .06). On average, bad steps (average LSA cosine of .11) were performed in 20.83 s (SD = 7.71) and required .17 hints (SD = .1). Broken down by degree of badness, bad steps (first degree of badness, LSA cosine of .25) were performed, on average, in 18 s (SD = 7.75) and required .13 hints (SD = .1). Bad steps (second degree of badness, LSA cosine of .15) were performed, on average, in 20.67 s (SD = 8.13) and required .17 hints (SD = .09). Bad steps (third degree of badness, LSA cosine of -.05) were performed, on average, in 23.83 s (SD = 6.47) and required .22 hints (SD = .09).

Figure 2 shows a linear trend in the effect of semantic match on performance time. As expected, good steps were performed faster than the average bad step, $F(1,47) = 127.47$, $p < .0001$. Likewise, there was a reliable linear effect of degree of badness on the bad steps performance time, $F(1,47) = 4.86$, $p < .05$. When moving from one degree of badness to another (i.e., moving farther away in the semantic space), the performance time increases, on average, by 2.9 s. Good steps required fewer hints than the average bad step, $F(1,47) = 60.69$, $p < .0001$. Additionally, for the bad steps, there was a reliable linear effect of degree of badness, $F(1,47) = 7.5$, $p < .05$. When moving from one degree of badness to another (i.e., moving farther away in the semantic space), on average, .04 more hints were required per step.

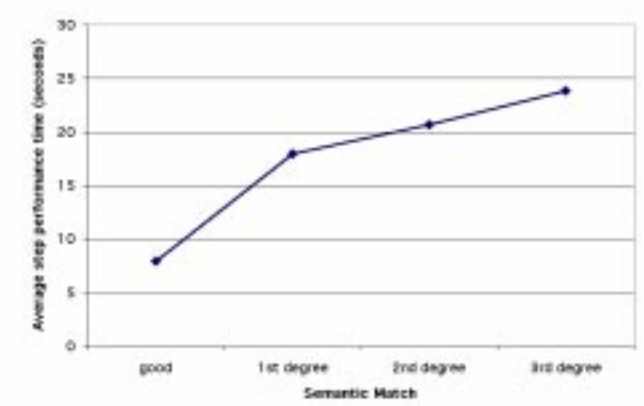

Figure 2. Effect of degree of badness on **step** performance time. (3rd degree of badness represents the worst semantic match)

Effect of Type of Training

Only the data for the recall session was analyzed to determine whether the type of training (exploration or explicit instruction) made a difference. As shown in Figure 3, explored tasks were performed faster than the instructed tasks. On average, explored tasks were performed during the recall session in 63.5 s (SD = 21.7) and required 0.58 hints (SD = .34), whereas instructed tasks where performed in 72.7 s (SD = 22.29) and required 0.76 hints (SD = .35). As expected, the time difference was not significant. However, the difference in the number of hints was significant, $F(1, 47) = 5.52$, $p < .05$. The interaction between type of training and degree of badness was not significant for task performance time, nor it was significant for the number of hints per task.

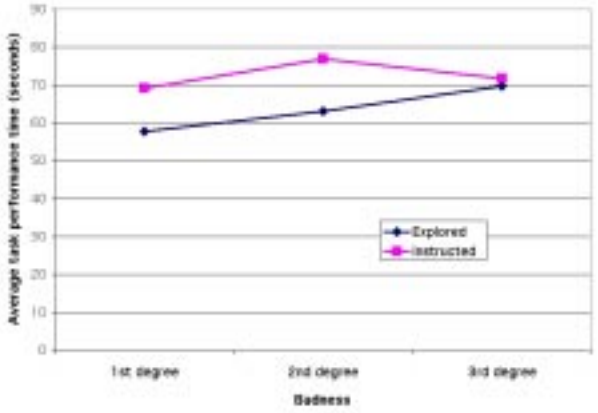

Figure 3. Effects of type of training and degree of badness on **task** performance time. (3rd degree of badness represents the worst semantic match)

The same analyses were performed at the step level, using the performance time and the number of hints required for the manipulated steps (the second and the third ones). Figure 4 shows that, on average, steps from explored tasks were performed faster than steps from instructed tasks. On average, steps belonging to explored tasks were performed during the recall session in 11.9 s (SD = 6.58) and required 0.06 hints (SD = .08), whereas instructed tasks where performed in 13.5 s (SD = 6.2) and required 0.1 hints (SD = .09). The difference in time was not significant, but the difference in the number of hints was significant, $F(1, 47) = 6.29$, $p < .05$. The interaction between type of training and degree of badness was not significant for step performance time, nor for the number of hints per step.

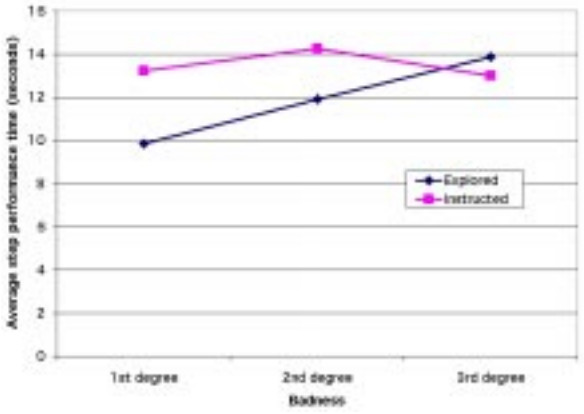

Figure 4. Effects of type of training and degree of badness on **step** performance time. (3rd degree of badness represents the worst semantic match)

Free Recall Data

Replicating the literature [23], the correct action sequences necessary to perform the tasks were very poorly recalled when participants did not have access to the application

interface. None of the subjects was able to recall a complete sequence of steps, and although 46 out of the 48 participants were able to recall at least one label, on average, only .11 labels were correctly recalled for each task. This number is significantly different from zero, $F(1,47) = 127.55$, $p < .0001$.

DISCUSSION

Semantic distance between task descriptions and menu labels reliably predicted the ease of discovering and recalling the experimental tasks. The semantic distance, computed as a LSA cosine, predicted users' performance not only at the task level, but also at the individual step level. Additionally, there was no significant difference between training methods (exploration vs. explicit instruction), and it was shown that subjects had very poor recall of the correct action sequences when they were away from the application interface.

Models of learning by exploration have been reviewed. All of them describe an attention mechanism that is driven by semantics. Regardless of the details of the processes assumed by each model, there is consensus that users select actions based on the semantic distance between the goal and the labels of the objects on the screen. It is assumed that the display can be represented as a collection of objects and labels, and that other information about the objects (e.g., what action can be carried out on them) is stored in long-term memory. Therefore, deciding what object to act on is a matter of matching object labels, task descriptions, and long-term memory knowledge. LSA can then be applied to any of these models to estimate semantic distance.

The instruction-less approach

No differences were found between the tasks that were learned by exploration and the tasks learned by explicit instruction. This manipulation was included to explore whether the memories for unsuccessful trails interfered with the storage of the experimenter's hints. In this experiment, every subject required at least one hint in both the training and the recall sessions. As Figure 3 shows, for the 3rd degree of badness (the worst semantic match), the difference in recall between the explored and the instructed tasks is virtually zero. During training, each 3rd degree task received, on average .84 hints (.70 for 1st degree tasks), which means that almost every subject, after 60 s of exploration, received a hint. These results show that memories from unsuccessful exploration do not interfere with the recall of the hint. In practical terms, it is worthwhile to explore the application for a while, and then ask for help from a peer, the manual, or the help system. This is just as good as receiving step-by-step instruction. Hopefully, new techniques can be developed with the help of LSA to minimize the number of tasks that cannot be discovered by exploration, so that users will be able to learn new applications without needing explicit instruction.

It can be speculated that memories from successful trials, especially those from well-labeled tasks, do not need to be stored in users' memory at all. Every time users face the task, they can reconstruct the whole action sequence by following the best matching labels. In other words, the external memory supplied by the application interface provides the necessary cues to re-discover the action sequence. Related to this, previous research [3] has found that subjects do not improve performance in well-labeled tasks from the training to the recall session. The putative lack of memory required for well-labeled tasks may be the reason why subjects have poor recall of the action sequences when they are away from the display.

Other applications of Latent Semantic Analysis to HCI

Perhaps it is impossible to design an application in which, for each task, the best matching labels can be always included in the action sequence. However, a good design should guarantee that the correct label is always the best match among the available labels. In order to evaluate the differences in semantic distance between labels and task descriptions, an objective method, such as LSA, could be desirable. So far, theorist and designer have used very informal estimates of semantic distance. This study suggests that this may not be necessary.

LSA could be used as an "automated" cognitive walkthrough [2]. This is a method for assessing the usability of a system, focusing on ease of learning. It involves hand simulation of the cognitive processes of how users, with no formal instruction, learn an application by exploration. The method takes into account users' elaboration of goals and users' interpretation of the application's feedback. The cognitive walkthrough is very labor intensive, and for this reason it is impractical for large modern applications. However, with LSA it would be possible to construct an automated system to evaluate large applications. Given a set of task descriptions and the labels of the objects that have to be acted on to perform these tasks, it is possible to evaluate the learnability of the system.

As stated above, LSA can be trained in any written language and with different corpora of texts. This makes it possible to model users with different backgrounds and skill levels. In the present study, a corpus of very broad and general knowledge was used to train LSA because the participants were mostly college freshmen, and there was no reason to believe they had any advanced technical knowledge. During the construction of the stimuli, it was found that one of the closest words to the phrase "hide the legend" (referring to a graph legend) was "dragons", with a cosine value of .41. This result is due to the fact that all the knowledge that the TASA space has about the word "legend" comes form epic novels, rather than from computer manuals ("map" and "heroes" are among the top 5

closest terms to "legend"). LSA has no way of knowing that “legend” also refers to part of a graph. The word "legend" was not used in the present experiment because "legend" does not seem to be a good way to describe this object to a novice user. Using computer manuals to train LSA could bring more information about how more advanced users exercise the label-following strategy.

Conclusions

This study showed that users rely on semantic similarity to discover the correct action sequence necessary to perform tasks using a novel application. The degree of closeness in a semantic space between the labels of the objects to be acted on and the description of the tasks determines how easy is for users to discover the correct action sequences and later to recall them. Latent semantic analysis proved to be a reliable way to measure and explain the users’ action-planning processes. The cognitive phenomena involved in the discovery and recall by exploration of computer tasks could be described as hill climbing driven by semantics or, in other words, as a process led by the label-following strategy.

LSA can be also applied to any of the cognitive models that has been developed to explain users’ performance of rarely used applications. Regardless of the particular mechanisms that the models propose to explain users’ behavior, they all rely on semantic issues that can be modeled with LSA. Eventually, LSA could be used in conjunction with other already available techniques (e.g., the cognitive walkthrough method) to automatically test the usability of computer applications. Additionally, it was shown that users have very poor recall of correct action sequences when they are away from the display. It was also shown that the type of training (exploration vs. explicit instruction) has no effect on recall.

ACKNOWLEDGMENTS

Partial support was provided by NASA Grant NCC 2-904. This paper is based on the author’s master thesis [21]. The author thanks his thesis committee members, Professor Peter G. Polson (chair), Professor Tomas K. Landauer, and Professor Walter Kintsch, for their help and support in developing this project. Dr. Eileen Kintsch provided very useful comments on an earlier version of this manuscript.

REFERENCES

1. Rieman, J., *et al.* (1991). An automated walkthrough. *Proceedings of CHI'91 Conference on Human Factors in Computer Systems*, pp. 427-428. New York, NY: ACM Press.
2. Polson, P.G., *et al.* (1992). Cognitive walkthroughs: A method for theory-based evaluation of user interfaces. *International Journal of Man-Machine Studies, 36*(5), 741-773.
3. Franzke, M. (1995). Turning research into practice: Characteristics of display-based interaction. *Proceedings of CHI'95 Conference on Human Factors in Computing Systems*, pp. 421-428. New York, NY: ACM Press.
4. Polson, P.G. and Lewis, C.H. (1990). Theory-based design for easily learned interfaces. *Human-Computer Interaction, 5*(2-3), 191-220.
5. Newell, A. and Simon, H.A. (1972). *Human Problem Solving*. Englewoods Cliffs, NJ: Prentice-Hall.
6. Engelbeck, G.E. (1986). *Exceptions to generalizations: implications for formal models of human-computer interaction.* Unpublished masters thesis, University of Colorado, Boulder, CO.
7. Muncher, E. (1989). *The acquisition of spreadsheet skills*. Unpublished masters thesis, University of Colorado, Boulder, CO.
8. Kitajima, M. and Polson, P.G. (1997). LICAI+: A Comprehension-Based Model of Learning for Display-Based Human–Computer Interaction. *Proceedings of CHI'97 Conference on Human Factors in Computing Systems*, pp. 333-334. New York, NY: ACM Press.
9. Rieman, J., Young, R.M., and Howes, A. (1996). A dual-space model of iteratively deepening exploratory learning. *International Journal of Human-Computer Studies, 44*(6), 743-775.
10. Howes, A. and Young, R.M. (1996). Learning consistent, interactive and meaningful device methods: A computational model. *Cognitive Science, 20*, 301-356.
11. Kitajima, M. and Polson, P.G. (1997). A Comprehension-Based Model of Exploration. *Human-Computer Interaction, 12*, 439-462.
12. Kitajima, M., Soto, R., and Polson, P.G. (1998). LICAI+: A Comprehension-Based Model of The Recall of Action Sequences. In F. Ritter and R.M. Young (Eds.), *Proceedings of the Second European Conference on Cognitive Modelling (Nottingham, April 1-4, 1998)* (pp. 82-89). Nottingham, UK: Nottingham University Press.
13. Kintsch, W. (1998). *Comprehension: A paradigm for cognition*. New York, NY: Cambridge University Press.
14. Mannes, S.M. and Kintsch, W. (1991). Routine Computing Tasks: Planning as Understanding. *Cognitive Science, 15*, 305-342.
15. Howes, A. (1994). A model of the acquisition of menu knowledge by exploration. *Proceedings of CHI'94 Conference on Human Factors in Computing Systems*, pp. 445-451. New York, NY: ACM Press.
16. Deerwester, S., *et al.* (1990). Indexing by Latent Semantic Analysis. *Journal of the American Society For Information Science, 41*(6), 391-407.
17. Landauer, T.K., Foltz, P., and Laham, D. (1998). An Introduction to Latent Semantic Analysis. *Discourse Processes, 24*, 259-284.
18. Landauer, T.K. and Dumais, S.T. (1997). A solution to Plato's problem: The latent semantic analysis

theory of acquisition, induction, and representation of knowledge. *Psychological Review, 104*(2), 211-240.

19. Landauer, T.K. and Dumais, S.T. (1996). How come you know so much? From practical problem to theory. In D. Hermann, *et al*. (Eds.), *Basic and applied memory: Memory in context* (pp. 105-126). Mahwah, NJ: Erlbaum.
20. Landauer, T.K. and Dumais, S.T. (1994). Latent semantic analysis and the measurement of knowledge. In R.M. Kaplan and J.C. Burstein (Eds.), *Educational testing service conference on natural language processing techniques and technology in assessment and education* . Princeton, N.J.: Educational Testing Service.
21. Soto, R. (1998). *Learning and Performing by Exploration: Label Quality Measured by Latent Semantic Analysis*. Unpublished master thesis, University of Colorado, Boulder, CO.
22. Ericsson, A.K. and Simon, H.A. (1980). Verbal Reports as Data. *Psychological Review, 87*(3), 215-251.
23. Payne, S.J. (1991). Display-based action at the user interface. *International Journal of Man-Machine Studies, 35*, 275-289.