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 similarity* between task descriptions and labels on display objects (e.g., menus, tool bars) controls in part the users' search strategies. Nevertheless, none of the models has an objective way to compute semantic similarity. In this study, Latent Semantic Analysis (LSA) was used to compute semantic similarity between task descriptions and labels in an application's menu system. Participants performed twelve tasks by exploration and they were tested for recall after a 1-week delay. When the labels in the menu system were semantically similar 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, usability analysis

INTRODUCTION

In the interaction between 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 experiment described in this paper provides empirical evidence supporting the hypothesis that users act on those interface labels that are semantically related to the users' goals. Several cognitive models have been proposed to simulate the exploration and recall of the action sequences required to perform novel tasks using display-based computer applications. All these models emphasize the use of semantic information as a means to successfully discover and recall the correct action sequences. Independently of the details in each model, they all use some version of semantic similarity as an evaluation function to predict the users' searching behavior. However, as shown here, LSA [4] proves to be a reliable technique to compute semantic similarity and it can be used to automate usability testing.

Outline of the paper

The next section describes the searching strategies users employ when learning a new application by exploration. Later, cognitive models of exploration are summarized emphasizing the role of semantic similarity in these models. An introduction of LSA is provided showing its role as an objective technique to compute automatically the semantic similarity between pieces of text. Finally, the experiment is described in detail and its results are discussed. This study has important implications for any cognitive model of exploration and for the design and automatic testing of interfaces that support learning by exploration.

LEARNING BY EXPLORATION

Polson and Lewis [5] analyzed the exploratory behavior of novice users who have a goal in mind, and who have some experience with particular types of applications or operating systems. In this situation, users engage in search through the interface objects for labels that will lead them to the solution of their task. During this process, the application changes from one state to another. For instance, a menu item is pulled down and a pop-up menu is exposed. When the pop-up menu items appear, the application's state changes to a richer one that contains more information that eventually will help the users find the solution of the task. The different states of the application define a problem space [6]. Novel users should employ some domain-independent method to guide their search through the problem space since they cannot anticipate the state shifts that their actions would produce. This kind of method is called a weak method because it does not contain any specific information about the problem, and means-ends analysis is probably its most used variation.

Two versions 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" [5, p. 205]. Since novel users do not have complete information about the available actions in each application state, they have to rely on their previous knowledge about the operating system or about display-based applications to select a particular action. In modern display-based applications, the most common actions are pulling down menu items, clicking on tool bars, "dragging and dropping" objects, or using "hot keys".

To accomplish a task, users need to estimate the distance between the current state and the desired state (i.e., the solution of the task). Engelbeck [7] observed that 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 [8] also found this behavior in novice users learning Lotus 1-2-3. This heuristic has been called the *label-following strategy* [5], 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 [3,5,9].

Ideally, the application interface would have a set of labels that maximizes the semantic similarity between the users' goals, or the task descriptions, and the object labels that have to be followed to perform the tasks. However, given the hierarchical structure of the menu systems, some labels are very general because they have to describe several different tasks at the same time. For instance, many modern applications have a menu item labeled **Tools**. If the users' goal is to perform a mail merge, the semantic similarity between the terms "mail merge" and "tools" is so low that it is unlikely that novice users will pull down the **Tools** menu in their first attempt. In this case users try other more promising labels or, if they are unsuccessful, they backtrack and try other options [5]. This back-chaining procedure allows them to uncover new labels that might be semantically more similar to their goals than the ones they attempted first.

The interaction of the hill-climbing strategy (label following) and back chaining strategy (backtracking to try less promising labels) constitutes the basic searching mechanism that most cognitive models include to explain exploratory behavior. Note that either strategy assumes that users estimate semantic similarity between labels and goals (or task descriptions). The review of the cognitive models of exploration reveals that each model implements semantic similarity very informally, an approach that is both unreliable and time consuming. Researchers have to include manually the necessary semantic information based on their own intuitions. Unless an objective measure of semantic similarity is available, choosing the right semantic features may vary considerably from one person to another [10].

COGNITVE MODELS OF EXPLORATION

SOAR Models

The Task-Action Learning (TAL) model [11] 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 experimenter instructions, the semantic features of the tasks, and the labels on 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" [11, 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 [12] simulates learning by exploration. Rieman [13] 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" [12, p. 747]. The IDXL model 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" [12, 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 synonyms has to be explicitly given to the model.

A Comprehension-Based Model of Exploration

The LICAI+ model [14,15] 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 [16], that was originally developed as a model for text comprehension and extended to action planning by Mannes and Kintsch [17]. LICAI+ predicts that successful exploration and recall require semantic matching between the goal representation (or task description) 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 similarity notion of this model 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.

In summary, the available models of learning by exploration share the same intuition about the role of semantic similarity: users tend to act on objects with labels that "seem" to be semantically related to their 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 similarity. This paper proposes that a mathematical model of semantics, such as Latent Semantic Analysis, is a good candidate for computing semantic similarity estimates.

LATENT SEMANTIC ANALYSIS 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 [18] that makes use of statistical procedures to capture the similarity of words and documents in a high-dimensional space [4,19]. 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 [19]. 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) [20,21]. It computes the semantic similarity 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 (see http://lsa.colorado.edu). 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 [11].

THE EXPERIMENT

The main limitation of both the theoretical and the empirical research on the label-following strategy is the lack of a well-defined measure for semantic similarity. In most cases, semantic relationships are established exclusively via some form of literal word overlap. Hence, the only well-defined similarity metric is, in LSA terms, a cosine of 1 (i.e., an exact match). Unless informal intuitive estimates of semantic similarity are included in the models, it is difficult to study situations where there are intermediate degrees of semantic similarity.

This experiment systematically manipulates the interface of Microsoft Excel to show that the semantic similarity (estimated by LSA) between the labels in the menus and the task descriptions predicts the ease of discovering the solution of the tasks. Additionally, a few different degrees and patterns of semantic similarity are used to explore the interaction between label-following and back-chaining strategies.

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 similarity between the labels of the menu system and the task descriptions. 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 participant's voice. Each participant received a package containing an informed consent form, a blue pen, and a notebook with the instructions and the task descriptions.

Tasks

There were four warm-up and eight experimental tasks. All consisted of editing a bar graph using a graphics application. Participants received detailed descriptions of the tasks, but no information about how to perform them (Table 1 shows the experimental tasks descriptions). The eight experimental tasks consisted 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.

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

Table 1. Description of the experimental tasks

The labels for the second and third steps were manipulated so that their semantic similarity with respect to the task descriptions varied. These labels were selected using LSA working under the TASA space. 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. Four degrees of semantic similarity were chosen based on the LSA cosines between the labels and the task descriptions. Good (G) labels had an average cosine of .69 (SD .17) with the task descriptions. The other three degrees of similarity had significantly lower cosines than the G labels had. Therefore, they were classified as three degrees of badness (B). B_1 labels had an average cosine of .25 (SD = .08), B_2 labels had an average cosine of .15 (SD = .02), and B_3 labels had an average cosine of -.05 (SD = .01) with the task description.

For example, a G label for task 1 (*Change the graph type to column*) was **Change Type** (cosine = .64), a B_1 label was **Correct Presentation** (cosine = .22), a B_2 label was **Rendering Practice** (cosine = .18), and a B_3 label was **Troubleshooter Route** (cosine = -.04). Depending on the condition (as explained in the next section) these labels were assigned to either the second or the third step of the task. That is to say, there were four different patterns of semantic similarity for the second and third steps of each task: a G label followed by a G label (G, G), (G, B), (B, G), and (B, B). In the last case, when both the second and the third steps had bad labels, the degree of badness for both labels remains constant: (B_1, B_1), (B_2, B_2), or (B_3, B_3).

Only the labels for the second and third steps were manipulated because they appear in similar contexts: submenus (pop-up menus). The labels for the other three steps had a fixed semantic similarity. The first step (top-level menu item) had a semantic similarity between B_1 and B_2 with respect to the description of the tasks under that menu. The fourth step (radio button or check box in a dialog box) had a semantic similarity of G. Finally, the fifth step ("Ok" button) had a semantic similarity between B_2 and B_3 with respect to any of the task descriptions.

Design

A menu structure template was created having eight top-level items: **File**, **Edit**, **General**, **Assign**, **Transformation**, **Appearance**, and **Tools**. Figure 1 shows the menu structure template under the **Assign** menu. Tasks 1 and 3 were under the **General** menu, tasks 2 and 8 were under the **Assign** menu, tasks 5 and 6 were under the **Transformation** menu, and tasks 4 and 7 were under the **Appearance** menu.

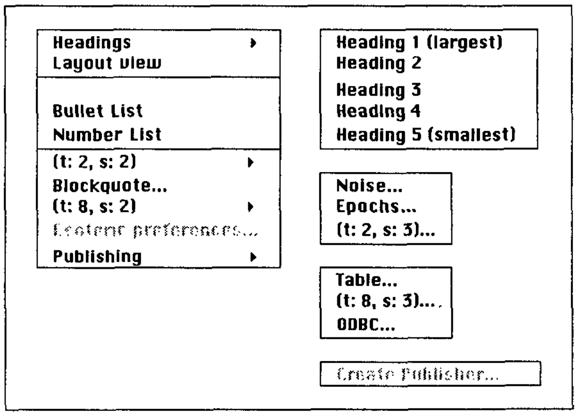

Figure 1. Menu structure template for the **Assign** menu. Data in parenthesis refer to task and step numbers. For instance, (t:2, s:3) is the slot for the third step of task 2.

Different conditions were created so the slots for the second and third steps could be filled out in a balanced way. Four patterns of similarity (G, G), (G, B), (B, G), and (B, B) were equally distributed across tasks, and across conditions. Since the degree of badness (B_1, B_2, and B_3) was fixed for each subject, it was considered a between factor. As an extra experimental factor, four of the eight experimental tasks were explicitly instructed, whereas the other four were explored (see next section). This factor was also balanced across tasks and across conditions. In total, 24 different conditions were created for this experiment.

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 instructed to pay close attention to what they did because they had to repeat it in one week. They were also informed that they would be explicitly instructed in 4 of the 12 tasks. 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 the participants a piece of paper with the task descriptions 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. Additionally, there were no effects related to the configuration of the 24 menu structures that were used. None of these factors was included in further analyses. In this paper, the effects of type of learning: explicit instructed versus explored, and the results from the verbal protocols are not analyzed.

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, over sessions, and over degree of badness to analyze the effect of similarity pattern: (G, G), (G, B), (B, G), and (B, B). As expected, the pattern (G, G) was performed much faster and required fewer hints than the other three patterns ($F(1, 47) = 195.3$, $p < .0001$, $F(1,47) = 189.8$, $p < .0001$, for time and for hints, respectively). Likewise, the (B, B) pattern was performed much more slowly and required more hints than the other three patterns ($F(1, 47) = 183.9$, $p < .0001$, $F(1,47) = 156.1$, $p < .0001$, for time and for hints, respectively). Finally, there was no significant difference between the (G, B), and the (B, G) patterns ($F(1,47) = 1.71$, $p = .19$, $F(1, 47) = 2.19$, $p = .14$, for time and for hints, respectively). Figure 2 shows the effect of similarity pattern and the effect of degree of badness on task performance time. The interaction between task configuration and degree of badness was not significant.

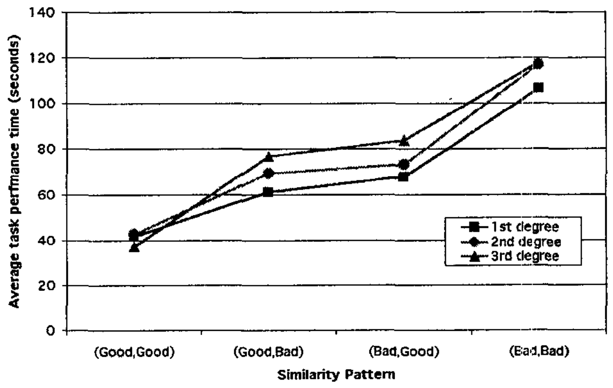

Figure 2. Effects of similarity pattern and degree of badness on **task** performance time. (3rd degree of badness represents the worst semantic similarity)

Step Data

Since only the semantic similarity of the second and third steps was manipulated, the data from those two steps were used to analyze the effects of semantic similarity and degree of badness. Collapsing over task configuration, sessions, and step number (second and third), the more semantically similar the label and the task description were, the faster the step was performed and the fewer hints were required. Good steps (average 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 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, 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, 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, cosine of -.05) were performed, on average, in 23.83 s (SD = 6.47) and required .22 hints (SD = .09).

Figure 3 shows a linear trend in the effect of semantic similarity 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 from more similar to less similar), 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 from more similar to less similar), on average, .04 more hints were required per step.

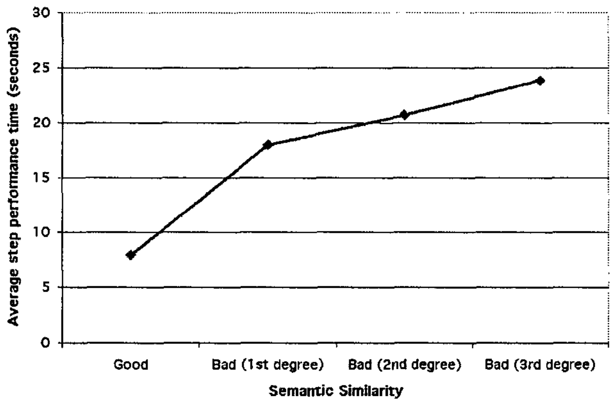

Figure 3. Effect of semantic similarity on step performance time. (3rd degree of badness represents the worst semantic similarity)

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.

DISCUSSION

Semantic similarity between task descriptions and menu labels reliably predicted the ease of discovering and recalling the experimental tasks. The semantic similarity, estimated by LSA cosines, predicted users' performance not only at the task level, but also at the individual step level. Different similarity patterns showed different reaction times: (G, G) was the fastest pattern, whereas (B, B) was the slowest. Additionally, it was shown that all subjects had very poor recall when they were away from the application interface.

Several models of learning by exploration have been reviewed here, all of which describe an attention mechanism that is driven by semantics. They agree that users select actions based on the semantic similarity between the task descriptions (or goals) and the display labels. These models assume 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, to decide what object to act on, users semantically match object labels, task descriptions, and long-term memory knowledge. LSA can thus be applied appropriately to any of these models to estimate semantic similarity. Certainly, LSA cannot replace any of the models because there are other issues in exploration and action planning (e.g., backtracking) that could not be explained by a purely semantics based model. However, LSA can be a valuable addition to any of the models and provides a powerful tool to test the usability of applications.

Usability testing

It is impossible to design an application in which, for each task description, only highly semantically related labels are used in the action sequence. The hierarchical structure of the menu forces the designer to select very broad and sometime ambiguous labels at the top of the hierarchy. However, a good interface should guarantee that the correct label is always the one with the highest semantic similarity among the available labels. In order to evaluate the differences in semantic similarity between labels and task descriptions, an objective method, such as LSA, is desirable. So far, theorists and designers have used very informal estimates of semantic similarity. This study suggests that this may not be necessary.

LSA could be used in addition 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 by which 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.

An alternative method is to hire expert designers to evaluate each of the labels of the application and their semantic similarity to the task descriptions. However, this method may be disadvantageous for various reasons. First, it may be highly unreliable because of the variance in opinion between one expert and another [24]. Second, it may be very time consuming, especially when tasks can be described in different ways, or when the application labels suffer minor changes during the design process. Finally, hiring expert evaluators can be very expensive.

LSA offers a more convenient method because it can rapidly estimate the semantic similarity of several alternative task descriptions and labels. When the application interface changes or when tasks are added or reformulated, the re-computation of the similarity estimates can be done very efficiently. Additionally, the estimation of semantic similarity can even be performed over the Internet (http://lsa.colorado.edu).

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 discovered 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 from 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 it does not seem to be a good way to describe a graph legend to a novice user. Using computer manuals or other more technical materials to train LSA may result in better and more accurate models.

Conclusions

This study showed that semantic similarity accurately predicts the ease of learning new computer tasks. The degree of similarity between the object labels to be acted on and the task descriptions drives the exploratory behavior of novel users. LSA proved to be a reliable way to estimate semantic similarity and, therefore, can be applied to any of the cognitive models that has been developed to explain users' exploratory behavior. 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.

ACKNOWLEDGMENTS

Partial support was provided by NASA Grant NCC 2-904. This paper is based on the author's master thesis [25]. 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 and three anonymous reviewers provided very helpful comments on earlier versions 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. Landauer, T.K., Foltz, P., and Laham, D. (1998). An Introduction to Latent Semantic Analysis. *Discourse Processes, 24*, 259-284.

5. Polson, P.G. and Lewis, C.H. (1990). Theory-based design for easily learned interfaces. *Human-Computer Interaction, 5*(2-3), 191-220.

6. Newell, A. and Simon, H.A. (1972). *Human Problem Solving*. Englewoods Cliffs, NJ: Prentice-Hall.

7. Engelbeck, G.E. (1986). *Exceptions to generalizations: implications for formal models of human-computer interaction.* Unpublished masters thesis, University of Colorado, Boulder, CO.

8. Muncher, E. (1989). *The acquisition of spreadsheet skills.* Unpublished masters thesis, University of Colorado, Boulder, CO.

9. 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.

10. Landauer, T.K., Galotti, K.M., and Hartwell, S. (1983). Natural Command Names and Initial Learning: A study of Text-Editing Terms. *Communications of the ACM, 26*(7), 495-503.

11. Howes, A. and Young, R.M. (1996). Learning consistent, interactive and meaningful device methods: A computational model. *Cognitive Science, 20*, 301-356.

12. 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.

13. Rieman, J.F. (1994). *Learning Strategies and Exploratory Behavior of Interactive Computer Users.* Unpublished Doctoral Dissertation, University of Colorado, Boulder, CO.

14. Kitajima, M. and Polson, P.G. (1997). A Comprehension-Based Model of Exploration. *Human-Computer Interaction, 12*, 439-462.

15. 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.

16. Kintsch, W. (1998). *Comprehension: A paradigm for cognition.* New York, NY: Cambridge University Press.

17. Mannes, S.M. and Kintsch, W. (1991). Routine Computing Tasks: Planning as Understanding. *Cognitive Science, 15*, 305-342.

18. Deerwester, S., *et al.* (1990). Indexing by Latent Semantic Analysis. *Journal of the American Society For Information Science, 41*(6), 391-407.

19. 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.

20. 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.

21. 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.

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.

24. Nielsen, J. (1992). *Applying Heuristic Evaluation to a Highly Domain-Specific User Interface.* Technical memorandum. Morristown, NJ: Bellcore.

25. Soto, R. (1998). *Learning and Performing by Exploration: Label Quality Measured by Latent Semantic Analysis.* Unpublished master thesis, University of Colorado, Boulder, CO.