2
Evolving a Learning Environment

Chapter 2 offers another fairly typical attempt to use the power of computer technology to support learning. Students need iterative practice with timely expert feedback for developing many skills, but computer-based drill and practice is not easy to implement in ways that are fun to use and educationally effective when the task involves interpreting semantics of free text. The `State the Essence` *software used latent semantic analysis (LSA) to solve this problem. It shows how a computer can provide a partial mentoring function, relieving teachers of some of the tedium while increasing personalized feedback to students.*

The software evolved through a complex interplay with its user community during classroom testing to provide effective automated feedback to students learning to summarize short texts. It demonstrates the collaboration among researchers, teachers and students in developing educational innovations. It also suggests collaborative group use of such software.

This case study is interesting not only for describing software design, implementation and adoption within a social context involving researchers, teachers and students, but also for its assessment of LSA, which is often proposed as a panacea for automated natural language understanding in CSCW and CSCL systems. It is an idea that at first appears simple and powerful, but turns out to require significant fine-tuning and a very restricted application. Success also depends upon integration into a larger activity context in which the educational issues have been carefully taken into account. In this case, well-defined summarization skills of individual students are fairly well understood, making success possible.

Interactive learning environments promise to significantly enrich the experience of students in classrooms by allowing them to explore information under their own intrinsic motivation and to use what they discover to construct knowledge in their own words. To date, a major limitation of educational technology in pursuing this vision has been the inability of computer software to interpret unconstrained free text by students in order to interact with students without limiting their behavior and expression.

In a project at the University of Colorado's Institute of Cognitive Science, a research group I worked in developed a system named `State the Essence` that provides feedback to students on summaries that they compose in their own words from their understanding of assigned instructional texts. This feedback encourages the students to revise their summaries through many drafts, to reflect on the summarization process, to think more carefully about the subject matter, and to improve their summaries prior to handing them in to the teacher. Our software uses a technology called latent semantic analysis (LSA) to compare the student summary to the original text without having to solve the more general problem of computer interpretation of free text.

LSA has frequently been described from a mathematical perspective and the results of empirical studies of its validity are widely available in the psychological literature.[1] This report on our experience with `State the Essence` is not meant to duplicate those other sources, but to convey a fairly detailed sense of what is involved in adapting LSA for use in interactive learning environments. To do this I describe how our software evolved through a two-year development and testing period.

In this chapter I explain how our LSA-based environment works. There is no magic here. LSA is a statistical method that has been developed by tuning a numeric representation of word meanings to human judgments. Similarly, `State the Essence` is the result of adapting computational and interface techniques to the performance of students in the classroom. Accordingly, this chapter presents an evolutionary view of the machinery we use to encourage students to evolve their own articulations of the material they are reading.

Section 1 of this chapter discusses the goals and background of our work. Section 2 takes a look at our interactive learning environment from the student perspective: the evolving student-computer interface. A central section 3 "lifts the hood" to see the multiple ways in which LSA is used to assess a student summary and formulate feedback. This raises questions about how LSA's semantic representation contained in our software itself evolved to the point where it can support decisions comparable to human judgments; these questions are addressed in the concluding section 4, which also summarizes our process of software design in use as a co-evolution, and suggests directions for continuing development.

1. Evolution of Student Articulations

Educational theory emphasizes the importance of students constructing their own understanding in their own terms. Yet most schooling software that provides automatic feedback to the students requires students to memorize and repeat exact wordings. Whereas the new educational standards call for developing the ability of students to engage in high-level critical thinking involving skills such as interpretation and argumentation, current software tools to tutor and test students still look for the correct answer to be given by a particular keyword. In the attempt to assess learning more extensively without further over-burdening the teachers, schools increasingly rely upon computer scoring, typically involving multiple choice or single word answers. While this may be appropriate under certain conditions, it fails to assess more open-ended communication and reflection skills—and may deliver the wrong implicit message about what kind of learning is important. Because we are committed to encouraging learners to be articulate, we have tried to overcome this limitation of computer support.

The underlying technical issue involves, of course, the inability of computer software to understand normal human language. While it is simple for a program to decide if a multiple choice selection or a word entered by a student matches an option or keyword stored in the program as the correct answer, it is in general not possible for software to

[1] See the special issue on LSA in *Interactive Learning Environments*, **8** (2) and the LSA web site at lsa.colorado.edu with interactive displays and publications.

decide if a paragraph of English is articulating a particular idea. This is known as the problem of "natural language understanding" in the field of artificial intelligence (AI). While some researchers have been predicting since the advent of computers that the solution to this problem is just around the corner (Turing, 1950), others have argued that the problem is in principle unsolvable (Dreyfus, 1972; Searle, 1980).

The software technique we call latent semantic analysis (LSA) promises a way to finesse the problem of natural language understanding in many situations. LSA has proven to be almost as good as human graders in judging the similarity of meaning of two school-related texts in English in a number of restricted contexts. Thus, we can use LSA to *compare* a student text to a standard text for semantic similarity without having to interpret the meaning of either text explicitly.

The technique underlying LSA was originally developed in response to the "vocabulary problem" in information retrieval (Furnas *et al.*, 1987). The retrieval problem arises whenever information may be indexed using different terms that mean roughly the same thing. When one does a search using one term, it would be advantageous to retrieve the information indexed by that term's synonyms as well. LSA maintains a representation of what words are similar in meaning to each other, so it can retrieve information that is about a given topic regardless of which related index terms were used. The representation of what words are similar in meaning may be extended to determine what texts (sentences, paragraphs, essays) are similar in topic. The way that LSA does all this should become gradually clearer as this chapter unfolds.

Because LSA has often proven to be effective in judging the similarity in meaning between texts, it occurred to us that it could be used for judging student summaries. The idea seemed startlingly simple: Submit two texts to LSA—an original essay and a student attempt to summarize that essay. The LSA software returns a number whose magnitude represents how "close" the two texts are semantically (how much they express what humans would judge as similar meanings). All that was needed was to incorporate this technique in a motivational format where the number is displayed as a score. Students would see the score and try to revise their summaries to increase their scores.

In 1996, we (see Notes at end of book) were a group of cognitive scientists who had been funded to develop educational applications of LSA to support articulate learners. We were working with a team of two teachers at a local middle school. We recognized that summarization skills were an important aspect of learning to be articulate and discovered that the teachers were already teaching these skills as a formal part of their curriculum. We spent the next two years trying to implement and assess this simple sounding idea. We initially called our application "`State the Essence`" to indicate the central goal of summarization.

A companion paper (Kintsch *et al.*, 2000) reports on the learning outcomes of middle school students using our software during two years of experimentation. Here I will just give one preliminary result of a more recent experiment I conducted informally, namely, to indicate the potential of this approach in a different context: collaborative learning at the college level. This experiment was conducted in an undergraduate computer science course on AI. The instructor wanted to give the students a hands-on feel for LSA so we held a class in a computer lab with access to `State the Essence`. Prior to class, the students were given a lengthy scholarly paper about LSA (Landauer, Foltz, & Laham, 1998) and were asked to submit summaries of two major sections of the paper as

homework assignments. Once in the lab, students worked both individually and in small teams. First they submitted their homework summary to `State the Essence`, and then revised it for about half an hour. The students who worked on part I individually worked on part II in groups for the second half hour, and vice versa.

Of course, I cannot compare the number of drafts done on-line with the original homework summaries because the latter were done without feedback and presumably without successive drafts. Nor have I assessed summary quality or student time-on-task. However, informal observation during the experiment suggests that engagement with the software maintained student focus on revising the summaries, particularly in the collaborative condition. In writing summaries of part I, collaborative groups submitted 71% more drafts than individual students—an average of 12 compared to 7. In part II (which was more difficult and was done when the students had more experience with the system) collaborative groups submitted 38% more drafts—an average of 22 drafts as opposed to 16 by individuals. Interaction with the software in the collaborative groups prompted stimulating discussions about the summarization process and ways of improving the final draft—as well as the impressive number of revisions. Computer support of collaboration opens up a new dimension for the evolution of student articulations beyond what we have focused on in our research to date. It would be important to develop interface features, feedback mechanisms and communication supports for collaboration to exploit the potential of collaborative learning.

2. Evolution of the Student-Computer Interface

What did the students view on the computer screen that was so motivating that they kept revising their summaries? The companion paper discusses in detail our shifting rationale for the design of the `State the Essence` interface. However, it may be useful to show here what the screen looked like after a summary draft was submitted. In the first year of our testing we built up a fairly elaborate display of feedback. Figure 2-1 shows a sample of the basic feedback.

Figure 2-1 goes approximately here

Note that the main feedback concerns topic coverage. The original text was divided into five sections with headings. The feedback indicates which sections the students' summaries cover adequately or inadequately. A link points to the text section that needs the most work. Other indications show which sentences are considered irrelevant (off topic for all sections), and which are redundant (repeating content covered in other sentences of the student summary). In addition, spelling problems are noted. Finally, warnings are given if the summary is too long or too short. The focus of the feedback is an overall score, with a goal of getting 10 points.

The evolution of the interface was driven primarily by the interplay of two factors:

1. Our ideas for providing helpful feedback (see next section).
2. The students' cognitive ability to take advantage of various forms of feedback (see the companion paper).

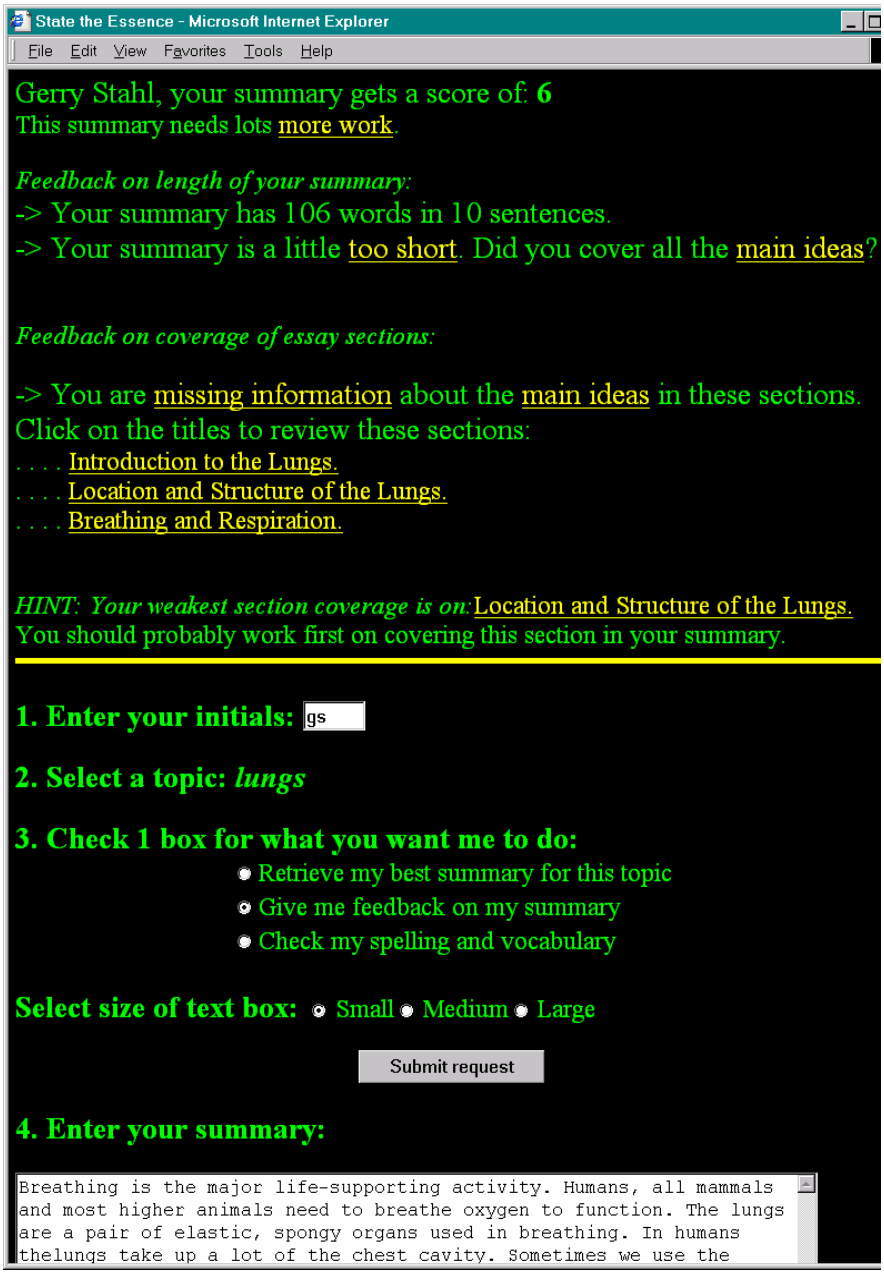

Figure 2-1. View of the early interface showing feedback from a draft summary at the bottom of the screen.

We found that there was a thin line between feedback that provides too little help and feedback that is overwhelming. The exact location of this line depends heavily upon such factors as student maturity, level of writing skills, class preparations for summarization tasks, classroom supports, and software presentation styles.

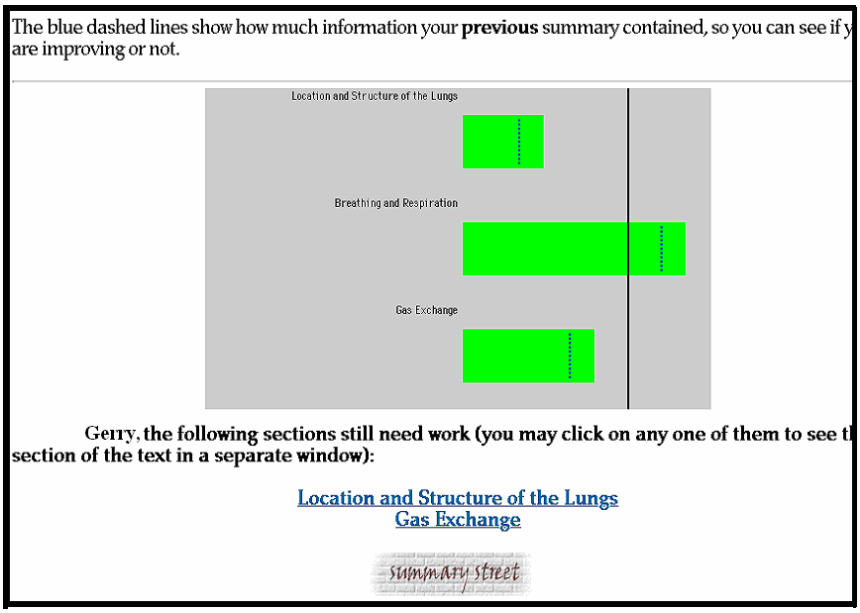

Figure 2-2. View of the later interface showing feedback from a draft summary.

For our second year, we simplified the feedback, making it more graphical and less detailed. Following a student suggestion, we renamed the system `SummaryStreet`. Figure 2-2 is a sample of feedback to a student summary: here the dominant feature is a series of bars, whose length indicates how well the summary covers each of the original text's sections. The solid vertical line indicates the goal to be achieved for coverage of each section. Dashed lines indicate the results of the previous trial, to show progress. Spelling errors are highlighted within the summary text for convenient correction. The detailed information about irrelevant and redundant sentences has been eliminated and the length considerations are not presented until a student has achieved the coverage goals for every section (these different forms of feedback will be described in the next section).

Figure 2-2 goes approximately here

Naturally, the AI college students in our recent experiment were curious about how the system computed its feedback. They experimented with tricks to tease out the algorithms and to try to foil LSA. What is surprising is that many of the sixth graders did the same thing. In general, learning to use the system involves coming to an understanding of what is behind the feedback. Interacting across an interface means attributing some notion of agency to one's communication partner. Even sixth graders know that there is no little person crouching in their computer and that it is somehow a matter of manipulating strings of characters.

3. Evolution of Feedback Techniques

So how does `State the Essence` figure out such matters as topic coverage? In designing the software we assumed that we had at our disposal a technology—the LSA function—that could judge the similarity in meaning between any two texts about as well as humans can agree in making such judgments. Let us accept that assumption for this section of the chapter; in the following section I will investigate the primary factors underlying this technology. When given any two texts of English words the function returns a number between –1.0 and 1.0, such that the more similar the meaning of the two texts, the higher the result returned. For instance, if we submit two identical copies of the same essay, the function will return 1.0. If we submit an essay and a summary of that essay, the function will return a number whose value is closer to 1.0 the better the summary expresses the same composite meaning as the essay itself. This section will report on how our use of the LSA function in `State the Essence` evolved during our research. This provides a detailed example of how the LSA technology can be adapted to an educational application.

In the course of our research we had to make a number of key strategic design decisions—and revise them periodically: (a) one was how to structure the software's feedback to provide effective guidance to the students. The feedback had to be useful to students in helping them to think critically about their summaries, recognize possible weaknesses and discover potential improvements to try. (b) Another decision was how to measure the overlap in meaning between a summary and the original essay. For this we had to somehow represent the essence of the essay that we wanted the summaries to approach; (c) this led to the issue of determining "thresholds," or standards of cut-off values for saying when a summary had enough overlap to be accepted. (d) Then we had to define a feedback system to indicate clearly for the students how good their summaries were and how much they were improving. I will now review each of these design decisions and discuss how they affected the student process of refining the summary.

a. Providing Guidance

Given the LSA function, we could have developed a simple form on the Web that accepts the text of a student's summary, retrieves the text of the original essay, submits the two texts to the function, multiplies the result of the function by 10 and returns that as the student's score. Unfortunately, such a system would not be of much help to a student who is supposed to be learning how to compose summaries. True, it would give the student an objective measure of how well the summary expressed the same thing as the essay, but it would not provide any guidance on how to improve the summary. Providing guidance—scaffolding the novice student's attempt to craft a summary—is the whole challenge to the educational software designer.

To design our software, we had to clearly define our pedagogical focus. We operationalized the goal of summary writing to be "coverage." That is, a good summary is one that faithfully captures the several major points of an essay. Secondarily, a summary should cover these points concisely: in perhaps a quarter the number of words of the original.

There are other factors that we considered and tried in various versions of the software. For instance, students should progress beyond the common “copy and delete” strategy where they excerpt parts of the original verbatim and then erase words to be more concise; learning to be articulate means saying things in your own words. However, even learning to manipulate someone else’s words can be valuable. We generally felt that the most important thing was for students to be able to identify the main points in an essay. It is also necessary that students learn to use the words that they come across in an essay. For instance a technical article on the heart and lungs has many medical terms that must be learned and that should probably be used in writing a summary. So avoiding plagiarism and reducing redundancy were less primary goals in a system for sixth graders than focusing on coverage.

Spelling is always a concern, although we would not want a focus on spelling to inhibit articulation and creativity. In a software feedback system, correct spelling is necessarily required, if only because misspelled words will not be recognized by the software. Other issues of composition had to be ignored in our software design. We made no attempt to provide feedback on logic or coherence of argument, literary or rhetorical style, and other aspects of expository writing. These were left for the teacher. Our system focused on helping students to “state the essence” of a given text by optimizing their coverage of the main points prior to submitting their compositions to a teacher for more refined and personal feedback. The power of LSA is limited and the limitations must be taken into account when designing its use context, balancing automated and human feedback appropriately.

b. Representing the Essence

The first question in defining our system algorithm was how to represent the main points of an essay so that we would have a basis for comparison with student summaries. Most educational essays are already fairly well structured: pages are divided into paragraphs, each of which expresses its own thought; an essay that is a couple pages long is generally divided into sections that discuss distinct aspects of the topic. For our classroom interventions, we worked closely with the teachers to select or prepare essays that were divided into four or five sections, clearly demarcated with headings. We avoided introduction or conclusion sections and assumed that each section expressed one or more of the major points of the essay as a whole. This allowed us to have the software guide the students by telling them which sections were well covered by their summaries and which were not. That is the central heuristic of our design.

So the idea is to compare a student summary with the main points of each section of the original text and then provide feedback based on this. The question is how to formulate the main points of a section for LSA comparison. There are several possible approaches:

(1) Use previously graded student summaries of text sections and determine how close a new summary is to any of the high-ranked old summaries. This method obviously only works when a text has been previously summarized by comparable students and has been carefully graded. This was not possible for most of our experiments.

(2) Have adults (researchers and/or teachers) laboriously hand-craft a “golden” summary of each section. This was our original approach. Typically, we had two summaries by the teachers and a couple by researchers; we then created one golden

summary for each section that synthesized all of the ideas contained in the adult summaries. We would then use this summary as a section target. In addition, each adult's complete set of section summaries was conglomerated for use as a target for the summary as a whole. The software compared the entire student summary to the "golden" target summary for each section and selected the highest LSA score to determine how well the student covered that section's points. Similarly, it also compared the entire student summary to each of the expert whole summaries to compute the student's score. That gave students a number of alternative adult summaries to target. This approach worked well. However, it required too much preparatory work. Each time we wanted to use a new essay in a classroom we would have to carefully prepare between a dozen and two dozen section summaries. This used too much teacher and researcher time and clearly would not scale up.

(3) Use the original text for comparison. This did not allow for feedback on coverage of each section.

(4) Use each section of the original text for a series of comparisons. The problem with this was setting thresholds. It is much easier to write a summary that gets a high LSA rating for some texts than it is for others. How do we know what score to consider good enough to praise or bad enough to criticize? Where adults hand-crafted expert target summaries we understood roughly what a 0.75 versus a 0.30 LSA score meant, but this was not the case for an arbitrary page of text. This led to our other major challenge: how to set standards of achievement in cases where we did not have a large base of experience.

c. Setting Standards

Setting thresholds is always an issue. The easier the method of defining comparison texts, the harder it is to set effective thresholds for them.

One idea we considered was to use past student summaries as a statistical basis for scoring new attempts. But that only worked for essays that had been used in past trials, and most of our experiments introduced new texts. So as an alternative to past summaries, we tried comparing hundreds of randomly selected short texts to the essay section to gain a measure of how hard the essay is to summarize (the random texts were selected from the corpus used for the LSA scaling space—see next section). We found that if a student summary does, say, four or five standard deviations better than a random text, it is probably fairly good. This approach was easy to automate and we adopted it. However, there were sometimes significant discrepancies between how hard it is for students to reach these thresholds for one essay section compared to another. We could adopt the attitude that life is just that way, and students need to learn that some things are harder to say than others. But we have some ideas on how to address this issue and we will revisit the issue in section 4 as part of our plans for future work.

d. Computing the Basic Feedback

Whatever approach we use to represent the sections and the whole text for LSA comparisons and whatever method we use to set the thresholds for what is considered an adequate or an inadequate comparison, we always compare a given student draft to each section and to the whole text in order to derive a score.

In our early version of `State the Essence`, we took the best LSA result from comparing the student summary to each expert whole summary. We multiplied this by 10 to give a score from 0 to 10. In addition to calculating this score, we computed feedback on coverage of individual sections. For each essay section, we took the best LSA result from comparing the student summary to each expert section summary. We compared this to thresholds to decide whether to praise, accept, or require more work on the section. Praised sections increased the student's score; criticized sections decreased it. We made additional adjustments for problems with summary length, redundancy, irrelevance, and plagiarism.

In the later version of the system, `SummaryStreet`, we compared the student summary draft with each section of the original text, as well as with the whole essay. The results of the LSA evaluations of the sections are compared to the automatically generated thresholds for the sections and the results are displayed graphically.

e. Refining the Summary

For a human, constructing a summary is a complex design problem with manifold constraints and sub-goals. Sixth graders vary enormously in their ability to do this and to respond to standardized guidance feedback. Telling a student that a particular section has not been covered adequately provides some guidance, but does not specify very clearly what has to be done. How does the student identify the main points of the section that are not yet covered in the summary? Primarily, the feedback points the student back to a confined part of the text for further study. The system even provides a hypertext link to that section so the student can reread it on the computer screen. The student can then try adding new sentences to the summary and resubmitting to see what happens. By comparing the results of subsequent trials, the student can learn what seems to work and what does not. The principle here is that instant and repeated feedback opportunities allow for learning through student-directed trial, with no embarrassing negative social consequences to the student for experimenting.

Repeated additions of material by a student, driven by the coverage requirement, inevitably lead to increasing length, soon exceeding the boundaries of a concise summary. In our early system, we continuously gave length feedback: a word count and a warning if the maximum length was being approached or exceeded. The composite score was also affected by excessive length, so it fluctuated in complex ways as more material was added. Dealing with the trade-off that was implicitly required between coverage and conciseness seemed to be more than most sixth graders could handle—although it might be appropriate for older students. So in our later system, `SummaryStreet`, we withheld the length feedback until the coverage thresholds were all met, letting the students pursue one goal at a time.

To help with the conciseness goal, we gave additional, optional feedback on relevance and repetition at the sentence level. This provided hints for the students about individual sentences in their summaries. They could view a list of sentences—or see them highlighted in their summary—that were considered irrelevant to the original essay or were considered redundant with other sentences in the summary. These lists were computed with many more LSA comparisons.

For the relevance check, each sentence in the student draft summary was compared (using LSA) with each section of the essay. A sentence whose comparison was well

above the threshold for a section was praised as contributing significantly to the summary of that section. A sentence whose comparison was below the thresholds for all the sections was tagged as irrelevant.

To check for overlapping, redundant content, each sentence in the student draft summary was compared with each other sentence of the summary. Where two sentences were very highly correlated they are declared redundant. Similarly, one could compare summary sentences with each sentence in the original to check for plagiarism, where the correlation approached 1.0. Again, this detailed level of feedback is very difficult for most sixth graders to use effectively.

A final form of feedback concerns spelling. This does not make use of the LSA function, but merely checks each word to see if it is in the lexicon that LSA uses. Because the LSA vocabulary combines a general K-12 textual corpus with documents related to the essay being summarized, most correctly spelled words used in student summaries are included in it.

As the preceding review indicates, the techniques for computing feedback in `State the Essence` evolved considerably over a two-year period. Perhaps most interesting is the variety of LSA computations that can be integrated into the feedback. From the original idea of doing a single LSA comparison of student summary to original essay, the system evolved to incorporate hundreds or even thousands of LSA computations. These comparisons are now used to automatically set a variety of system thresholds and to evaluate summaries at the sentence, section and holistic levels.

At least at the current state of the technology, testing and fine tuning of many factors are always necessary. The final product is an opaque system that returns reasonable feedback in about a second and seems simple. But to get to that point each component of the system had to be carefully crafted by the researchers, reviewed by the teachers and tested with students. This includes the style of the text, its division into sections, the representation of the essence of each section, the values of multiple thresholds, the presentation of the feedback and various factors discussed in the next section, including the composition of the scaling space and the choice of its dimensionality.

Another conclusion to be drawn from the history of the evolution of our techniques is the importance of tuning system feedback to the needs and abilities of the audience, rather than trying to exploit the full power that is computationally possible. I will reflect on this process in the next section as well as taking a closer look at how it is that the LSA function can do what it does in the computations just described.

4. Co-Evolution of the Software in Use

This chapter adopts an evolutionary view of software development. The experience of our project with `State the Essence` can be summed up by saying that a *co-evolution* has taken place among the various participants. The research goals, the software features, the teacher pedagogy, the student attitudes and the classroom activities have changed remarkably over the two years. They have each changed in response to the other factors so as to adapt to each other effectively. Such an effective *structural coupling* (Maturana & Varela, 1987) between the development of the software and the

changing behavior of the user community may constitute a significant indicator for a successful research effort.

Some of these changes and interactions among the researchers, teachers and students were documented elsewhere (Kintsch *et al.*, 2000). The present chapter focuses more on the software development process in relation to student cognition. Section 1 argued that the educational point of the project is to promote evolution at the level of the individual student's ability to articulate his or her understanding of instructional texts. Preliminary impressions from an experiment discussed in that section suggest that collaborative uses of the software may be even more powerful than individual uses. At a larger scale, significant changes in the classroom as a community were informally observed in the interactions during single classroom interventions as well as during the school year, even when the software use was nominally being conducted by students on an individual basis. Students tended to interact with friends around use of the software, helping each other and sharing experiences or insights. Section 2 reviewed the evolution of the software interface as it adjusted to student difficulties, and section 3 traced this back to shifts in approaches at the level of the underlying algorithms. One can go a step deeper and see the use of the basic LSA technology in our software as a product of a similar evolutionary adaptation.

Evolution of the Semantic Representation

At one level, the semantic representation at the heart of LSA is the result of a learning process. It is equivalent to the connections in AI neural networks that learn to adjust their values based on experience with training data. It can be argued that an LSA analysis of a corpus of text has learned from that corpus much of what a child learns from the corpus of text that the child is exposed to (Landauer & Dumais, 1997). One difference is that LSA typically analyses the corpus all at once rather than sequentially, but that does not make an essential difference. In certain applications it might be important for LSA to continually revise its values—to continue learning. For instance, in `State the Essence` it might be helpful to add new student summaries to the corpus of analyzed text as the system is used, to take into account the language of the user community as it becomes available.

The mathematical details of LSA have been described elsewhere, as have the rigorous evaluations of its effectiveness. For purposes of understanding the workings of `State the Essence` in a bit more depth and for appreciating both the issues that we addressed as well as those issues that remain open, it is necessary to review some of the central concepts of LSA at a descriptive level. These concepts include: scaling space, co-occurrence, dimensionality reduction, cosine measure and document representation.

Scaling space. The representation of meaning in LSA consists of a large matrix or high-dimensionality mathematical space. Each word in the vocabulary is defined as a point in this space—typically specified by a vector of about 300 coordinates. The space is a "semantic" space in the sense that words which people would judge to have similar meanings are located proportionately near to each other in the space. This space is what is generated by LSA's statistical analysis of a corpus of text. For `State the Essence`, we use a large corpus of texts similar to what K-12 students encounter in school. We supplement this with texts from the domain of the essays being summarized, such as

encyclopedia articles on the heart or on Aztec culture. The semantic space is computed in advance and then used as a "scaling space" for determining the mathematical representations of the words, sentences and texts of the student summaries. It may seem counter-intuitive that a mathematical analysis of statistical relations among words in written texts could capture what people understand as the meaning of those words—akin to learning language from circular dictionary definitions alone. Yet experiments have shown that across a certain range of applications, LSA-based software produces results comparable to those of foreign students, native speakers or even expert graders.

Co-occurrence. The computation of semantic similarity or nearness (in the space) of two words is based on an analysis of the co-occurrence of the two words in the same documents. The corpus of texts is defined as a large number of documents, usually the paragraphs in the corpus. Words that co-occur with each other in a large number of these documents are considered semantically related, or similar. The mathematical analysis does not simply count explicit co-occurrences, but takes full account of "latent" semantic relationships—such as two words that may never co-occur themselves but that both co-occur with the same third word or set of words. Thus, synonyms, for instance, rarely occur together but tend to occur in the same kinds of textual contexts. The LSA analysis not only takes full advantage of latent relationships hidden in the corpus as a whole, but scales similarities based on relative word frequencies. The success of LSA has shown that co-occurrence can provide an effective measure of semantic similarity for many test situations, when the co-occurrence relationships are manipulated in sophisticated ways.

Dimensionality reduction. The raw matrix of co-occurrences has a column for every document and a row for every unique word in the analyzed corpus. For a small corpus this might be 20,000 word rows x 2,000 document columns. An important step in the LSA analysis is dimensionality reduction. The representation of the words is transformed into a matrix of, say 20,000 words x 300 dimensions. This compression is analogous to the use of hidden units in AI neural networks. That is, it eliminates a lot of the statistical noise from the particular corpus selection and represents each word in terms of 300 abstract summary dimensions. The particular number 300 is somewhat arbitrary and is selected by comparing LSA results to human judgments. Investigations show that about 300 dimensions usually generate significantly better comparisons than either higher or lower numbers of dimensions. This seems to be enough compression to eliminate noise without losing important distinctions.

Cosine measure. If one visualizes the LSA representation of words as a high-dimensionality mathematical space with 300 coordinate axes, then the vector representing each word can be visualized as a line from the origin to a particular point in the space. The semantic similarity of any two words can be measured as the angle between their vectors. In LSA applications like `State the Essence`, this angle is measured by its cosine. For two points close to each other with a very small angle between their vectors, this cosine is about 1.0. The larger the angle between the two words, the lower the cosine. While it might seem that nearness in a multi-dimensional space should be measured by Euclidean distance between the points, experience with LSA has shown that the cosine measure is generally the most effective. In some cases, vector length is also used (the combination of cosine and vector length is equivalent to Euclidean distance). We are considering adopting vector length measures in `State the Essence` as well, to avoid problems we have encountered—discussed in the next section.

Document representation. In our software, LSA is not used to compare the meanings of individual words but to assess content overlap between two documents (sentences, summaries, essay sections, whole essays). It is standard practice in LSA applications to represent the semantics of a document with the vector average of the representations of the words in the document, massaged by some factors that have proven effective empirically. Thus the two documents we are comparing are taken to be at the centroid (vector average) of their constituent words within the same scaling space as their individual words. We then use the cosine between these two centroid points as the measure of their semantic content similarity. On language theoretic grounds this may be a questionable way to compute sentence semantics. One might, for instance, argue that "there is no way of passing from the word as a lexical sign to the sentence by mere extension of the same methodology to a more complex entity" (Ricoeur, 1976, p. 7), because while words may just have senses defined by other words, sentences refer to the world outside text and express social acts. In response to such an argument, one might conjecture that the confines of our experiment protect us from the theoretical complexities. `State the Essence` is only looking for overlapping topic coverage between two documents. Because of this operational focus, one might speculate that it is the simple similar inclusion of topical words (or their synonyms) that produces the desired experimental effect. However, we have done some informal investigations that indicate that it is not just a matter of topical words that influences LSA's judgments; the inclusion of the proper mix of "syntactic glue" words is important as well. Nevertheless, it may be that the LSA-computed centroid of a well-formed sentence performs on average adequately for practical purposes in the tasks we design for them because these tasks need not take into account external reference (situated deixis) or interactional social functions. For instance, we do not expect LSA to assess the rhetorical aspects of a summary.

This overview of the key concepts of the LSA technology suggests that LSA is not an approach that came ready-made based on some a priori principle and that can be applied automatically to every situation. Quite to the contrary, the method itself has evolved through iterative refinement, under the constant criterion of successful adaptation to comparison with human judgment. The force driving the evolution of the LSA technology as well as that of our application has always been the statistical comparison with human judgments at a performance level comparable to inter-human reliability. In its application to summarization feedback, our use of LSA has significantly evolved to a complex use of many LSA-based measures, blended into an interaction style carefully tuned to the intended audience through repeated user trial.

Evolution into the Future

Nor is the use of LSA in `State the Essence` fixed now as a result of our past work. There are a number of technical issues that must be further explored. There are also practical improvements needed if this software is to be deployed for classroom use beyond the research context.

At least four technical issues that have already been mentioned in passing need further attention: space composition, threshold automation, vector length measurement and plagiarism flagging.

Space composition. As noted, our scaling spaces for the middle school students were based on a corpus of documents that included both generic K-12 texts and domain-specific texts related to the essay being summarized. It is still not clear what the optimal mix of such texts is and the best way of combining them. Clearly, it is important to include some domain-specific material so that the space includes meaningful representations of technical terms in the essay. It is also important to have the general vocabulary of the students well represented in order to give valid feedback when they express things in their own words. The problem is that two distinct corpora of text are likely to emphasize different senses of particular words, given the considerable polysemy of English words. Mathematical techniques have been proposed for combining two LSA spaces without disrupting the latent relationships determined for each space, and we must explore these techniques under experimental conditions. The creation and testing of an LSA scaling space is the most computationally intensive and labor intensive part of preparing an intervention with `State the Essence`. If we are to make this learning environment available for a wide range of essays in classrooms, we must find a way of preparing effective scaling spaces more automatically.

Threshold automation. The other technical aspect that needs to be further automated is the setting of reasonable thresholds for a diversity of texts and for different age levels of students. We have already experimented with some approaches to this as described above. Yet we still find unacceptable divergences in how easy it is for students to exceed the automatically generated thresholds of different texts. We have noticed that some texts lend themselves to high LSA cosines when compared to a very small set of words—sometimes even a summary a couple of words long. These are texts whose centroid representation is very close to the representation of certain key words from the text. For instance, a discussion of Aztecs or solar energy might include primarily terms and sentences that cluster around the term “Aztec” or “solar energy.” According to LSA measurements, these texts are well summarized by an obvious word or two.

Vector length measurement. We suspect that the use of both vector lengths and cosines to measure overlapping topic coverage between two texts will address the threshold problem just discussed—at least partially. But we need to experiment with this. The rationale for this approach is that vector length corresponds to how much a text has to say on a given topic, whereas cosine corresponds to what the topic is. Thus, a document consisting of the single word “Aztec” might be close to the topic of an essay on the Aztecs and therefore have a high cosine, but it would not be saying much about the topic and thus would have a small vector length. The inclusion of vector lengths within LSA-based judgments would allow `State the Essence` to differentiate between a quick answer and a more thoughtful or complete summary. Here, again, the software must evolve in response to tricks that students might use to achieve high scores without formulating quality summaries.

Plagiarism flagging. Of course, the simplest way to get a good LSA score is to just copy the whole essay as one’s summary. This is a winning strategy for topic coverage. The length is too long, so one must then cut the unnecessary details. Here, the sixth grader faces a task that requires distinguishing essential points from inessential details—a task that many sixth graders must still learn. A related alternative approach is to copy topic sentences from each section or paragraph of the original and use them for one’s summary. Again, this requires an important skill that `State the Essence` is

intended to help teach: identifying topic ideas. So, it is probably a decision best left to the teacher to decide how much copying of vocabulary, phrases and even whole sentences is acceptable in a given exercise. Perhaps for older students, such as college undergraduates, the system should object to any significant level of plagiarism. It is still necessary to define the boundaries of what one considers to be plagiarism, such as reusing and/or reordering sentence clauses. In the end, such matters may have to be automatically flagged for subsequent teacher review and judgment.

Of course, there is still much else to do before `State the Essence` is ready for widespread deployment. In addition to wrapping up these open research issues and continuing to refine the system's functionality and interface, there is the whole matter of packaging the software for easy use by teachers and of integration with curriculum. Another possibility is to include `State the Essence` as a tool within larger interactive learning environments like `CSILE` (van Aalst *et al.*, 1999) or `WebGuide` (see chapter 6). Perhaps all that can be said now is that we have taken `State the Essence` far enough to suggest its potential educational utility and to demonstrate how LSA technology can be integrated into an interactive, constructivist, student-centered approach to facilitating student articulation.