J. EDUCATIONAL COMPUTING RESEARCH, Vol. 24(3) 305-320, 2001

A SYSTEM TO ASSESS THE SEMANTIC CONTENT OF STUDENT ESSAYS

BENOÎT LEMAIRE
PHILIPPE DESSUS
University of Grenoble-II

ABSTRACT

This paper presents Apex, a system that can automatically assess a student essay based on its content. It relies on Latent Semantic Analysis, a tool which is used to represent the meaning of words as vectors in a high-dimensional space. By comparing an essay and the text of a given course on a semantic basis, our system can measure how well the essay matches the text. Various assessments are presented to the student regarding the topic, the outline and the coherence of the essay. Our experiments yield promising results.

INTRODUCTION

This paper describes Apex (for an Assistant for Preparing EXams), a tool for evaluating student essays based on their content. It relies on a semantic text analysis method called Latent Semantic Analysis (LSA) [I]. Basically, LSA represents each word of a text as a vector in a high-dimensional space such that the proximity between two vectors is closely related to the semantic similarity between the two corresponding words. Since a text is composed of words, this measure of similarity can be easily extended to compare texts. Thus, it is possible to compare two student essays, or a student essay and the text of a course, on a semantic basis.

Our goal is not only to grade essays per se but more generally to provide students with an environment to help them learn a domain by writing an essay, getting feedback, revising the text, etc. This process is compatible with the Hayes and Flower model of text composition [2]. However, our environment differs from

other composition tools or tutors that provide lexical or grammatical information because it decreases the cognitive effort of the writer. Instead of building an intrusive tool, we aim to develop an open-ended learning environment [3] in which the user is viewed as an active constructor of knowledge.

The paper is organized as follows: first, we present LSA and the various intelligent assessors or tutors that are based on this tool; second, we describe our tool, Apex, as well as two experiments we performed.

LATENT SEMANTIC ANALYSIS

Researchers in information retrieval (IR) have tried to identify lexical-semantic relations between words of text corpora. Evens, Vandendorpe and Wang analyzed lexical-semantic relations of words to extract both syntactic and semantic information of bibliographic queries [4]. The authors developed IRS (Information Retrieval System), an IR automated procedure for revealing semantic relations from dictionary definitions. IRS is a system in which each bibliographic reference is an automatically extracted list of keywords used to identify documents. The system views each list as a document-vector. The user writes a paragraph, which is first transformed into a query-vector, and it is then compared with the document-vectors by computing the cosine between the former and the latter. The closer to one is the cosine, the closer to the query is the document retrieved. This method adequately expresses both syntactic and semantic information without complex data preprocessing.

Following these ideas, a new method, called Latent Semantic Analysis, gives promising results for semantic text processing. Before presenting LSA-based intelligent tutors or assessors, we need to describe LSA.

Description of LSA

In order to perform a semantic matching of pieces of text, LSA relies on large corpora of texts to build a semantic high-dimensional space containing all words and texts, by means of a statistical analysis [5, 6].

Basically, the semantics of a word is determined from all of the contexts (namely paragraphs) in which that word occurs. For instance, the word *bike* occurs generally in the context of *handlebars, pedal, ride,* etc. Therefore, if a word like *bicycle* occurs in a similar context, the two words will be considered close to each other from a semantic point of view. Their corresponding vectors in the semantic space will be also close to each other.

This semantic space is built by considering the number of occurrences of each word in each piece of text (basically paragraphs). For instance, with 300 paragraphs and a total of 2,000 words, we get a 300 × 2,000 matrix. Each word is then represented by a 300-dimensional vector and each paragraph by a 2,000-dimensional vector. So far, it is just straightforward occurrence processing.

The power of LSA, however, lies in the reduction of these dimensions, and in so doing induces semantic similarities between words. All vectors are reduced by a method close to eigenvector decomposition to, for instance, 100 dimensions. The matrix X is decomposed as a unique product of three matrices: $X = T_0 S_0 D_0^{'}$ such that T_0 and D_0 have orthonormal columns and S_0 is diagonal. This method is called singular value decomposition. Then only the 100 columns of T_0 and D_0 corresponding to the 100 largest values of S_0 are kept, to obtain T, S, and D. The reduced matrix $\overline{X}$ such that: $\overline{X} = TSD'$ permits all words and paragraphs to be represented as 100-dimensional vectors. It is this reduction which is the heart of the method because it allows the representation of the meaning of words, by means of the context in which they occur.

If the number of dimensions is too small, too much information is lost. If it is too big, not enough dependencies are drawn between vectors. A size of 100 to 300 gives the best results in the domain of language [7].

This method is quite robust: a word could be considered semantically close to another one although they do not co-occur in texts. In the same way, two documents could be considered similar although they share no words. An interesting feature of this method is that the semantic information is derived only from the co-occurrence of words in a large corpus of texts. There is no need to code semantic knowledge by means of a semantic network or logic formulas.

Various Tests of LSA

We now present several experiments that test LSA. One experiment consisted in building a general semantic space from a large corpus of English texts, then testing it with the synonymy tests of the TOEFL (Test Of English as a Foreign Language) [1]. One word is given, and the task is to identify which word among a choice of four possible answers is semantically closest to it. LSA performed the test by choosing the word with the highest similarity between its vector and the vector of the given word. LSA results compare favorably with the level of foreign students admitted to American universities.

In another study, subjects were asked to write essays from a number of texts. These essays were then ranked by human graders [8]. Their task was to assess the adequacy between the essay and the texts. In parallel, LSA was "trained" with the texts and ranked the essays according to the semantic proximity between each of them and the texts. LSA results compare favorably again with the human results.

LSA also proved to be successful in checking hypertext links from a semantic point of view [9]. In other words, LSA can test whether a link between two pages is appropriate based on their semantic content.

Landauer and Dumais considered LSA a good model of human learning [1]. Their assumption is that humans construct knowledge by deriving word meaning from context while reading [10]. Actually, Rehder et al. show that LSA is an appropriate tool to assess human knowledge in a specific domain [11].

Incorporating LSA into Instructional Software: Tutors and Assessors

LSA was also used in the field of tutoring systems and intelligent assessors. This section is devoted to the presentation of various applications of LSA in that domain.

Although LSA has been used for over ten years in several domains like information retrieval, machine learning, text analysis and so on, few LSA-based studies have been performed to improve instructional software. To our knowledge, the earliest attempt at implementing LSA in an educational tutor was carried out in 1997 by Stahl [12]. Stahl suggests that LSA may be appropriate in education in the following ways:

1. to find the optimal text for learning; LSA can match student essays with texts in the same domain, in order to provide the student with the right amount of new information;
2. to connect students with each other and with relevant experts; LSA can assess the areas of interest or the level of knowledge of the users from their productions and then suggest matches;
3. to automatically assess writing, as we mentioned earlier.

According to these ideas and the previous psychological experiments, we can consider LSA an appropriate model of knowledge representation for designing instructional software. The following sections review various applications.

Application One—Autotutor

Wiemer-Hastings et al. are developing Autotutor, an ITS with multimedia and human-like interaction (the authors plan to rely on speech synthesis and simulated facial movements) in the domain of computer literacy [13, 14]. One of the purposes of Autotutor, for which LSA is required, is to answer unrestricted student questions. It does so by selecting the closest piece of text to the question. The main interest is that the student is not limited to single word questions. Moreover, the system performance degrades gracefully which is usually not the case in natural language systems. However, since the user questions are quite short (10 to 20 words), it is likely that Autotutor does not use all of the power of LSA. Indeed the authors show quite similar performances when relying on keyword matching rather than LSA [15] which is usually not the case in other LSA experiments [16]. Autotutor is a large project in progress, for which further evaluation in context is planned.

Application Two—Language Learning and Game Learning

LSA is not only a tool for matching texts on a semantic basis, but also a model of human learning, since LSA learns semantic similarities between words from the

reading of texts, just as humans do. Actually, it was shown [1] that the rate of vocabulary learning with LSA compares with the human rate between the ages of 2 and 20: when provided with 3,500 words a day, LSA learns 10 new words per day which is roughly close to the human data.

This model of learning was used in a tutoring system to select the best stimulus to which to expose the student in order for learning to be optimal [17]. LSA is used to represent the domain model and the user model in the same semantic space. Two applications were designed. In the first one, the system selects English texts for French learners to read, according to their understanding of previous texts. The second one extends LSA to knowledge domains other than language. It helps students learn a game by playing with them. It selects a move not to win but in order that the next state of the game is the optimal stimulus for the student.

Application Three—Intelligent Essay Assessor

Much work has been conducted in the field of automatic grading but the systems are mainly based on multiple choice exams [18, 19]. These grading programs are not hard to make. The difficulty lies rather in the design of the propositions which should be close enough to the right answer but still wrong. An alternative to multiple choice tests is to ask the student to write an essay about what he or she knows about a domain and then to compare that text to pre-graded texts.

As Wresch [20] pointed out, automated grading of essays by computer began thirty five years ago with Page's work [21]. Page looked for correlations between simple features of student texts and the corresponding teacher grades. The most statistically significant features were: average word length (r = .51), number of common words (r = –.48), number of commas (r = .34), number of words (r = .32), number of prepositions (r = .25). Although these rough measures have often been criticized, some recent systems are closely related to Page's work (e.g., the Grammatik proofreading software). This approach has a major drawback: it only focuses on surface text features. Neither microstructure nor semantic textual characteristics are taken into account by these programs.

LSA can address these drawbacks because of its ability to capture semantic information. This is the foundation of the Intelligent Essay Assessor (IEA) [22]. IEA is first "trained" on several texts related to the domain. The student essay is then compared with pre-graded essays by means of two kinds of scores:

1. the *holistic score,* which returns the score of the closest pre-graded essay;
2. the *gold standard,* which returns the LSA proximity between the student essay and a standard essay.

An experiment using the holistic score was performed on 188 essays on biology. A correlation of .80 was shown between IEA grades and human graders.

One main interest of IEA is that the student can submit the essay again and again in order to improve the score. Foltz et al. [22] note that the average grade increases

from 85/100 (first submission) to 92/100. However, one problem is that the teacher needs to grade essays beforehand.

Application Four—Summary Street

Summary Street and State of Essence [23, 24] are built on top of LSA. Both systems help students write good summaries. First of all, the student is provided with general advice on how to write a summary: select the most important information, find two or three ideas, substitute a general term for lists of items, not include trivial information, etc. Then the student selects a topic, reads the text, and writes out a summary. LSA procedures are then applied to give a holistic grade to the summary. The system also warns the student if the summary is too long or too short.

Application Five—Select-a-Kibitzer

In the same way, one of the designer of Autotutor has recently built Select-a-Kibitzer [25], an agent-based computer tool that assesses student compositions. Each agent is responsible for providing a kind of advice: coherence, purpose, topic, and overall quality. It is worth noting that these assessments are quite similar to those provided by the previous applications. However, this application emphasizes the negotiated construction of the text by associating each agent to a character.

Now we will present Apex, a similar system that we built on top of LSA, to help students learn a domain. Apex differs from IEA in that it does not rely on pre-graded essays but rather on various semantic comparisons with the course. It also differs from Summary Street and Select-a-Kibitzer since it takes into account the structure of the course to grade an essay.

APEX

As we indicated earlier, Apex can be used to grade a student essay with respect to the text of a course; however it can also provide detailed assessments on the content, the outline, and the coherence of a student essay. The environment is designed so that the student can select a topic, write an essay on that topic, get various assessments, then rewrite the text, submit it again, etc. The only information Apex needs to process a student essay is the text of the course. This text has to be marked up by the teacher so that it is divided into topics, and so that each topic is divided into notions. Basically, this structure corresponds to an outline view of a word processor, so it is quite straightforward to mark up such a document. First level titles should begin with #T (for Topic) whereas second level titles should begin with #N (for Notion). Only second level titles are followed by a paragraph. A notion can belong to several topics. However, the text of such a notion is written only once. In case of cross-references, the notion title should begin with #S. Figure 1 gives an example of such a marking-up. The notion title

```
#T The solar system
   #N Introduction to the solar system
      The solar system is composed of . . . .
      . . . . . .
   #N The sun
      The sun is a star which is . . . .
      . . . . . .
   #N The planets
      There are 9 planets in the solar system: . . . .
      . . . . . .

#T Eclipses
   #S Introduction to the solar system
   #N Sun eclipses
      When the moon is between the earth and the sun, . . . .
      . . . . . .
   #N Moon eclipses
      When the earth is between the sun and the moon, . . . .
      . . . . . .
```

Figure 1. Marked-up text of a course to be processed by Apex.

"Introduction to the solar system" is defined in the topic "The solar system" and referred to in the topic "Eclipses."

It is worth noting that we added a very large set of French texts (290,000 words) to the course, in order to improve the semantic knowledge of the system. These texts (three French novels) were not related to the course but they were useful for the system to better "understand" the non domain-dependent terms of the student essay.

Once connected to the system, the student selects a topic that he or she wishes to work on. The student then types a text about this topic in a text editor. At any time, he or she can get an evaluation of the essay.

Different Types of Evaluation

Three kinds of evaluation are available: content-based, outline-based, or coherence-based. After reading any of the three evaluation texts, the student goes back to the essay, modifies the text, and submits it again.

Content-Based Assessment

At the content level, the system identifies how well the notions are covered by requesting LSA to measure a semantic similarity (in the range [–1, 1]) between the

student text and each notion of the selected topic. For instance, if the student selects the topic "Eclipses," he or she has to write an essay about that topic. There are three notions in that topic; therefore the student text will be successively compared with each of them. If the similarity is high, it means that the student has covered well the corresponding notion. Actually, the system provides a message to the student according to the value of the similarity. The current version of Apex relies on four categories:

1. if the similarity $\in$ [–1; 0.1[, the message is "the topic was covered very poorly";
2. if the similarity $\in$ [0.1; 0.5[the message is "the topic was covered poorly";
3. if the similarity $\in$ [0.5; 0.7[the message is "the topic was covered well";
4. if the similarity $\in$ [0.7; 1] the message is "the topic was covered very well."

However, the number of categories as well as the corresponding values and the text of the message can be fully parametrized depending on the content of the course. To do so, the teacher has to edit and modify a text file which is shown in Figure 2.

In addition to this assessment, Apex provides a general grade of the student text. This grade is the average grade for each notion, multiplied by 20 in order to get a grade between 0 and 20 as is usual in the French school system.

Figure 3 shows an example of a content-based evaluation of a student essay in the domain of cognitive ergonomics.

One problem we had to tackle concerns very short student texts. When the student has written out just a few words, the content-based assessment could yield a high score, although one might want the assessment to be considered low.

```
##### Apex parameters (c)  B. Lemaire, P. Dessus,
##### University of Grenoble-2, FRANCE

##### Content-based assessment messages and thresholds
contentThreshold1=0.7
contentMessage1=was covered very well
contentThreshold2=0.5
contentMessage2=was covered well
contentThreshold3=0.1
contentMessage3=was covered poorly
contentThreshold4=-1
contentMessage4=was covered very poorly
...
```

Figure 2. Excerpt of the parameters to be modified by the teacher.

Document length: 658 words.
GENERAL GRADE: 9.7 / 20

The following notions were covered very poorly:
- *Automatism-based errors* (0.26)
- *Baille and Testu's experiment on problem solving* (0.25)

The following notions were covered poorly:
- *Errors based on declarative knowledge* (0.40)
- *Testu's model (1993)* (0.34)

The following notions were covered well:
- *The definition of activity* (0.59)
- *Rasmussen's model* (0.70)
- *Errors based on procedural knowledge* (0.57)

The following notions were covered very well:
- *Applications of Rasmussen's model* (0.77)

Figure 3. Example of a content-based assessment.

Therefore, if the number of words of the student text is below 300 words (that value can be easily changed in the parameter file), all the content thresholds described earlier are raised so that the assessment is more severe: the shorter is the text, the closer to 1 becomes the thresholds. Basically, with *N* being the number of words of the essay, all content thresholds *T* are changed to:

$$T \quad (1 \quad T) \, \frac{300 \quad N}{300}$$

We also designed *Apex on-line,* a version which runs continuously in order for the student to get a real time assessment. While the student is typing out texts, the Apex window constantly refreshes the assessment. Instead of displaying the quite long natural language message about which notion is well or poorly covered, the system displays a scale on which all the notions of a topic are represented as numbers. The far right part of that scale corresponds to a similarity of 1, the far left to a similarity of 0. Therefore, the student can immediately identify the notions that are covered badly. The scale looks like this:

```
BAD . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . GOOD
          2  6        4     3             7 1         5    8
```

In this example, the student knows that he or she has to work mainly on notions 2 and 6 to improve the text.

Outline-Based Assessment

At the outline level, the system displays the most similar notion of the course for each paragraph of the essay. The goal is to provide the student with an outline view of the essay. If the similarity computed by LSA is too low, the system prints that there is no predicted notion. This threshold is currently set to 0.4 but this can be changed easily by means of the parameter file we mentioned earlier. Figure 4 shows an example of an outline-based assessment. For instance, the first paragraph of the student text has been recognized by Apex as being concerned with the definition of activity. If this is not what was intended, the student should probably rework this paragraph.

Coherence-Based Assessment

LSA has been used already to measure text coherence and has proven to be successful [26]. At the coherence level, the system relies on LSA to measure semantic proximities between adjacent sentences. Therefore Apex can detect coherence breaks. Then it gives an average measure of coherence and if necessary an example of an important conceptual break between two sentences. Figure 5 shows an example of a coherence-based assessment: the student is required to work on the text again, and in particular to correct the linking of ideas between sentence 2 and sentence 3.

Software Information

LSA was designed by Bellcore Labs from which we obtained the source code, written in C under Unix. Apex is also written in C under Unix. The approximate

```
Paragraph 1: Before describing Rasmussen's model, it is neces...
Predicted notion (0.72): The definition of activity
Paragraph 2: At level n, Rasmussen considers the behavior as ...
Predicted notion (0.77): Applications of Rasmussen's model
Paragraph 3: The hierarchy is justified by the necessity to t...
No predicted notion.
Paragraph 4: Suppose we select a specific application of the...
Predicted notion (0.91): Applications of Rasmussen's model
Paragraph 5: Reason has established a list of possible errors...
Predicted notion (0.60): Rasmussen's model
...
```

Figure 4. Example of an outline-based assessment.

```
coherence sentence 1 - sentence 2:  0.67
coherence sentence 2 - sentence 3:  0.16
coherence sentence 3 - sentence 4:  0.58
coherence sentence 4 - sentence 5:  0.79
coherence sentence 5 - sentence 6:  0.52
...
Poor inter-sentence coherence (0.49)
For instance, there is a big conceptual break between
the following sentences:
2: The activity is a set of unobservable behaviors that...
3: There are generally two kinds of procedures in the...
```

Figure 5. Example of a coherence-based assessment.

duration for the assessments, with a 192 Ko course text, and a 1.9 Mo additional French text, on a workstation are:

- 300 ms per notion for content-based assessment;
- 3 seconds per paragraph of the student text for outline-based assessment;
- 150 ms per sentence of the student text for coherence-based assessment.

EXPERIMENTS

This set of experiments is based on a graduate course on the sociology of education at our university.

Correlation with Human Grades

In this experiment, we took 31 essays that were written a year ago by students at the university of Grenoble. We typed them out and ran Apex in order to get the general grade between 0 and 20 for each student essay. We first compared these grades with the grades the teacher had given a year ago. We got a significant correlation ($r = .59$, $p < .001$).

Then all essays were ranked by two judges who were teachers in similar domains. They had studied the relevant part of the course several times before the task. They graded each of the 31 texts in the same way that Apex does: each text was given 14 grades corresponding to the estimated adequacy between each of the 14 notions. Then an average grade was computed. We got significant correlations with Apex grades ($r_1 = .59$, $p < .001$; $r_2 = .68$, $p < .0001$). However these aggregate measures could not be refined due to a lack of variability in the 14 notions.

All of these results are in the range of correlations found in the literature. Wiemer-Hastings et al. had also compared LSA grades and human grades and none of the correlations were above .5 [15]. Foltz's similar correlations [8] were in

the range [.3; .55]. Wolfe et al. [27] indicate correlations around [.6; .7]. Kintsch et al. [23] mention a value of .64. The highest correlations ever found [.8; .86] were obtained by Foltz et al. [22]. These repeated experiments show that LSA is a adequate tool for assessing knowledge, one that rivals the ability of humans.

Effects of Apex Feedback on the Quality of Student Essays

We designed a second experiment to assess the effect of Apex on the holistic quality of students essays. Three groups of students were given 45 minutes to write an essay:

1. a control group ($N = 11$) in which subjects wrote out an essay on a word processor;
2. an *Apex-demand* group ($N = 11$), where students wrote out their essays with the same word processor as the control group and were provided with help upon request;
3. an *Apex on-line* group ($N = 9$) in which the only difference with the latter group is the on-line help provided by Apex without explicit requests from the student.

Our hypothesis was that students in the *Apex on-line* group would write out better essays than students of the two other groups. The reason is twofold: first, the solicitation of Apex help in the *Apex-demand* group is supposed to increase the writer's cognitive effort, and second, the help provided is supposed to improve the essay quality as opposed to the control group. This is coherent with the results of Zellermayer et al. [28].

A question was selected that covered 14 "notions" of the course according to the teacher. The 31 students wrote their essays. All essays were ranked by three judges: two of them (graders 1 and 2) were teachers in similar domains and the last one was a student who had passed the course. Each judge gave 14 grades for each text by the same procedure used in the first experiment.

Results are twofold (see Table 1). First, correlations between the human scores and the Apex ones are in a similar range to those obtained elsewhere (see supra).

Table 1. Experiment 2: Correlations of Grades between Graders and Apex

	2	3	Apex
1	0.83**	0.72**	0.38
2		0.78**	0.45*
3			0.46*

*$p < .05$
**$p < .0001$

However, these correlations are smaller than the correlations between humans grades (around .8).

Second, results (see Table 2) show no significant differences between groups which is coherent with a similar experiment [23]. We plan to design another experiment to study the student writing process instead of just the performance. Methods such as think-aloud protocols [29] or retrospective interviews [30] would be appropriate.

Table 2. Experiment 2: Group Mean, Standard Deviations, and ANOVA Results

	Control	Apex-Demand	Apex On-Line
Average speed of typing (char/s)			
M	1.53	1.36	1.26
SD	0.48	0.33	0.28
Number of words of the essay			
M	478.91	418.36	478.44
SD	193.40	120.68	151.35
Mean Apex grades (0.20 range)			
M	9.93	9.63	9.82
SD	0.36	0.65	0.31
Mean grades of grader 1			
M	8.36	6.64	7.56
SD	4.9	5.66	4.13
Mean grades of grader 2			
M	11.91	9.91	12.11
SD	6.47	6.41	4.17
Mean grades of grader 3			
M	13.27	9.18	10.89
SD	7.8	7.57	7.3

Note: ANOVA results:
Speed typing (length of text typed: 209 characters): $F(2,28) = 1.33$, ns.
Number of words of the essay: $F(2,28) = .51$, ns.
Mean Apex grades: $F(2,28) = 1.11$, ns.
Mean grades of grader 1: $F(2,28) = .33$, ns.
Mean grades of grader 2: $F(2,28) = .45$, ns.
Mean grades of grader 3: $F(2,28) = .81$, ns.

CONCLUSION

This paper describes a system that can assess student essays based on their content from different points of view. Our first experiment provide promising results. In particular, we show a significant interesting correlation between human grades and Apex grades. This can lead to the development of automatic grading systems not only based on multiple choice exams or surface features of essays, but rather on semantic features of unrestricted essays. This approach can also be used in a distance learning context since students can connect to the system and freely submit essays. The system evaluates the essays as many times as required by the students, without getting bored. Moreover, there is no need for the teacher to code any domain knowledge. The text of the course is all that is required.

Apex is only concerned with the content: the teacher is essential for evaluating spelling, syntax and the general structure of the essay. However, Apex evaluation of the content has some limits. Actually, LSA (which is the heart of Apex) has no way of detecting when a sentence has syntactic errors or when some common words are missing. Therefore, a student could fool LSA by writing an essay with only keywords. However, we think that if a student can provide the right keywords to fool LSA, then this student should have a good knowledge of the domain and that is exactly what we want to measure.

ACKNOWLEDGMENTS

We would like to thank Susan Dumais and the Bellcore Labs to have allowed us to use and hack the code of the basic LSA programs. We are also grateful to Pascal Bressoux who provided us with the text from one of his courses and who helped us in formatting this course for our system, and to Diana Amadis, Ira Noveck, and Erica de Vries for their editorial inputs. Many thanks also to our typists and graders, Severine Parsoud and Dorothée Tourniaire.

REFERENCES

1. T. K. Landauer and S. T. Dumais, A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction and Representation of Knowledge, *Psychological Review, 104*:2, pp. 211-240, 1997.
2. J. R. Hayes and L. S. Flower, Identifying the Organization of Writing Processes, in *Cognitive Processes in Writing,* L. W. Gregg and E. R. Steinberg (eds.), Erlbaum, Hillsdale, pp. 3-30, 1980.
3. S. M. Land and M. J. Hannafin, A Conceptual Framework for the Development of Theories in Action with Open-Ended Learning Environments, *Educational Technology Research and Development, 44*:3, pp. 37-53, 1996.
4. M. Evens, J. Vandendorpe, and Y. C. Wang, Lexical-Semantic Relations for Information Retrieval, in *Humans and Machines,* S. Williams (ed.), Ablex, Norwood, pp. 73-100, 1995.

5. S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshmann, Indexing by Latent Semantic Analysis, *Journal of the American Society for Information Science, 41,* pp. 391-407, 1990.
6. T. K. Landauer, P. W. Foltz, and D. Laham, An Introduction to Latent Semantic Analysis, *Discourse Processes, 25*:2-3, pp. 259-284, 1998.
7. S. T. Dumais, Improving the Retrieval of Information From External Sources, *Behavior Research Methods, Instruments, & Computers, 23*:2, pp. 229-236, 1991.
8. P. W. Foltz, Latent Semantic Analysis for Text-Based Research, *Behavior Research Methods, Instruments, & Computers, 28*:2, pp. 197-202, 1996.
9. P. Dessus, Semantic Checking of Hypertext Links, in *Hypertextes, Hypermédias et Internet,* J.-P. Balpe, A. Lelu, S. Natkin, and I. Saleh (eds.), Hermès, Paris, pp. 119-129, 1999 (in French).
10. R. G. Fukkink and K. de Glopper, Effects of Instruction in Deriving Word Meaning from Context: A Meta-Analysis, *Review of Educational Research, 68*:4, pp. 450-469, 1998.
11. B. Rehder, M. E. Schreiner, M. B. Wolfe, D. Laham, T. K. Landauer, and W. Kintsch, Using Latent Semantic Analysis to Assess Knowledge: Some Technical Considerations, *Discourse Processes, 25*:2-3, pp. 337-354, 1998.
12. G. Stahl, Allowing Learners to be Articulate: Incorporating Automated Text Evaluation into Collaborative Software Environments, *Proposal to the McDonnell Foundation,* 1997.
13. P. Wiemer-Hastings, K. Wiemer-Hastings, and A. Graesser, Improving an Intelligent Tutor's Comprehension of Students with Latent Semantic Analysis, in *Artificial Intelligence in Education (AIED'99),* S. Lajoie and M. Vivet (eds.), IOS Press, Amsterdam, pp. 535-542, 1999.
14. P. Wiemer-Hastings, K. Wiemer-Hastings, and A. Graesser, Approximate Natural Language Understanding for an Intelligent Tutor, in *Proceedings of the 12th Florida Artificial Intelligence Research Symposium (FLAIRS'99),* 1999.
15. P. Wiemer-Hastings, How Latent is Latent Semantic Analysis? in *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI'99),* Stockholm, 1999.
16. B. Lemaire, Models of High-Dimensional Semantic Spaces, in *Proceedings of the 4th International Workshop on MultiStrategy Learning (MSL'98),* Desenzano, 1998.
17. B. Lemaire, Tutoring Systems Based on Latent Semantic Analysis, in *Artificial Intelligence in Education (AIED'99),* S. Lajoie and M. Vivet (eds.), IOS Press, Amsterdam, pp. 527-534, 1999.
18. D. Blank, G. Holmes, R. Wells, and P. Wolinski, Interactive Gradebook: The Missing (Hyper)Link, *ACM SIGCSE,* 1998.
19. D. W. Stockburger, Automated Grading of Homework Assignments and Tests in Introductory and Intermediate Statistics Courses Using Active Server Pages, *Behavior Research Methods, Instruments, & Computers, 31*:2, pp. 252-262, 1999.
20. W. Wresch, The Imminence of Grading Essays by Computer—25 Years Later, *Computers and Composition, 10*:2, pp. 45-58, 1993.
21. E. Page, The imminence of Grading Essays by Computer, *Phi Delta Kappan, 47,* pp. 238-243, 1966.
22. P. W. Foltz, D. Laham, and T. K. Landauer, Automated Essay Scoring: Applications to Educational Technology, *Proceedings of the ED-MEDIA'99 Conference,* AACE, Charlottesville, 1999.

23. E. Kintsch, D. Steinhart, G. Stahl, and the LSA Research Group, Developing Summarization Skills through the Use of LSA-Based Feedback, *Interactive Learning Environments, 8*:2, pp. 87-109, 2000.
24. G. Stahl, R. dePaula, and the LSA Research Group, Evolution of an LSA-Based Interactive Environment for Learning to Write Summaries, *Interactive Learning Environments,* to appear.
25. P. Wiemer-Hastings and A. Graesser, Select-a-Kibitzer: A Computer Tool that Gives Meaningful Feedback on Student Compositions, *Interactive Learning Environments, 8*:2, pp. 149-169, 2000.
26. P. W. Foltz, W. Kintsch, and T. K. Landauer, The Measurement of Textual Coherence with Latent Semantic Analysis, *Discourse Processes, 25*:2-3, pp. 285-307, 1998.
27. M. B. Wolfe, M. E. Schreiner, B. Rehder, and D. Laham, Learning from Text: Matching Readers and Texts by Latent Semantic Analysis, *Discourse Processes, 25*:2-3, pp. 309-336, 1998.
28. M. Zellermayer, G. Salomon, T. Globerson, and H. Givon, Enhancing Writing-Related Metacognitions through a Computerized Writing Partner, *American Educational Research Journal, 28*:2, pp. 373-391, 1991.
29. R. B. Kozma, The Impact of Computer-Based Tools and Embedded Prompts on Writing Processes and Products of Novice and Advanced College Writers, *Cognition and Instruction, 8*:1, pp. 1-27, 1991.
30. A. Fatton and K. Severinson-Eklundh, How to Support In-Process Planning in a Computer-Based Writing Environment, in *Proceedings of the European Writing Conference,* Barcelona, 1996.

Direct reprint requests to:

Dr. Benoit Lemaire
Educational Sciences Laboratory
University Grenoble-II
BP47
F-38040 Grenoble Cedex 9
France