Psychonomic Bulletin & Review
2005, 12 (4), 703-710

Using context to build semantics

PETER J. KWANTES
Defence R&D Canada, Toronto, Ontario, Canada

Latent semantic analysis (LSA) is a model of knowledge representation for words. It works by applying dimension reduction to local co-occurrence data from a large collection of documents after performing singular value decomposition on it. When the reduction is applied, the system forms condensed representations for the words that incorporate higher order associations. The higher order associations are primarily responsible for any semantic similarity between words in LSA. In this article, a memory model is described that creates semantic representations for words that are similar in form to those created by LSA. However, instead of applying dimension reduction, the model builds the representations by using a retrieval mechanism from a well-known account of episodic memory.

How do people learn the meanings of the words they encounter from their exposure to language? More pointedly, how is it that we seem to be able to pick up the meanings of new words easily, without being told what they mean? Landauer and Dumais (1997) offered an answer to the question with latent semantic analysis (LSA).

LSA is a computational model that derives vector-based semantic representations for words from large corpuses of text. A word's semantic representation in LSA is a vector describing how often the word occurs in, potentially, thousands of contexts or documents. The basic idea behind the model is that words that have similar meanings will tend to appear in the same or similar contexts (see Burgess, Livesay, & Lund, 1998, for a model that works on similar principles). For example, key words appearing in documents about automobiles tend not to appear in documents about telephones. The semantic similarity between words, then, reflects the amount of contextual overlap between them.

Semantic similarity between two words goes beyond their co-occurring in the same document or context (i.e., raw co-occurrences), however. The terms *general* and *admiral*, for example, would seem to have a clear semantic similarity by virtue of the fact that they are both military ranks. Despite any similarity the terms may have in semantic space, however, they may rarely occur in the same document; admirals are specific to the navy, whereas generals exist in the other military branches. If the two rarely occur together in the same document, a semantic representation characterized by the extent to which they appear in the same contexts would fail to reveal the extent of their similarity. To uncover *general* and *admiral*'s latent semantic relationship, the system requires a mechanism that can exploit higher order associations between the two terms. For example, the semantic system needs to figure out that *general* and *admiral* are related because, even if the terms rarely occur in the same documents, they appear in contexts that share terms associating them, such as *military*, *commanding*, and *officer*.

How LSA Creates Vector Representations for Words

Landauer, Foltz, and Laham (1998; see also Landauer & Dumais, 1997) provide detailed accounts on how LSA forms semantic representations for words. Briefly, however, LSA starts by entering thousands of documents into its memory. For each document (which could be an encyclopaedia entry or a newspaper article), it tabulates the number of times each word appears in each document. To store word frequency information over several thousand documents, a term $\times$ document matrix is formed in which each word is described as a vector, the elements of which contain the number of times the word occurred in each document. One can think of each word's vector as a point in d-dimensional space, where d is the number of documents, or contexts, in the training corpus.

LSA uncovers the latent higher order information by performing two operations on the term $\times$ document matrix. First, during a preprocessing stage, raw cell frequencies of a term are transformed by the formula ln(frequency + 1) and are divided by the entropy of the term across all the documents in which the term occurs. Entropy is calculated as $-\Sigma\, p \ln p$, where p is the transformed frequency of a term in one document, divided by the total transformed frequency of a term across all documents. According to Landauer and Dumais (1997), the logarithmic transformation "approximates the standard empirical growth functions of simple learning" (p. 216). In weighting an entry by its entropy, each cell also provides information about how strongly a term is anchored to a context.

This work was funded by the Department of National Defence and, in large part, by an Early Career Research Grant to P.J.K. from the University of Queensland. A debt of gratitude is owed to Simon Dennis and Mike Humphreys for guidance in the early stages of this work. Thanks are also extended to Douglas Nelson, James Nairne, and Douglas Hintzman for valuable comments on previous versions of this article. Correspondence regarding this article should be addressed to P. J. Kwantes, DRDC-Toronto, 1133 Sheppard Ave. W., P.O. Box 2000, Toronto, ON, M3M 3B9 Canada (e-mail: peter.kwantes@drdc-rddc.gc.ca).

After preprocessing, a statistical technique called *singular value decomposition* (SVD) is applied to the term × document matrix. SVD is a form of factor analysis. With it, a rectangular matrix is decomposed into three component matrices. When the three matrices are multiplied together, the original (transformed) matrix is nearly perfectly reconstructed.

The key to constructing semantic representations in LSA lies in one of the component matrices—a diagonal matrix containing *d* singular values. The singular values express how important each of the *d* dimensions is in recreating the original term × document matrix. The greater the magnitude of a singular value, the more variance in the original matrix it captures and, hence, the more important the dimension is to the reconstruction.

The next step for LSA is dimension reduction, wherein only a handful (e.g., 200) of the highest singular values are retained during the reconstruction, whereas the remaining ones are ignored (i.e., set to zero). Now all the terms must be differentiated on only 200 dimensions. Dimension reduction collapses the component matrices in such a way that the system loses some of its ability to discriminate among documents that share terms and among terms that are shared across documents. When reconstruction of the original matrix is attempted, the reduced dimensionality can cause terms that never occurred in the same documents to appear related. For example, *general* and *admiral* may never appear in the same document; however, if their respective documents share several words (e.g., *military*, *officer*, etc.) and the model has trouble distinguishing the documents, LSA conflates the terms *general* and *admiral*.

How well does the technique work? LSA's power has been demonstrated in a variety of domains. Thus far, it has been able to perform as well as a foreign student on the Test of English as a Foreign Language (TOEFL; Landauer & Dumais, 1997), classify documents in a meaning-based query system (Dumais, 1994), match reviewers to submitted papers (Dumais & Nielsen, 1992), and predict some semantic priming data collected in the laboratory (Landauer et al., 1998).

Landauer and Dumais (1997) were quick to point out that they did not believe that the brain performed SVD on co-occurrence information stored in memory. They did, however, make two fairly strong theoretical claims. First, they claimed that whatever psychological mechanisms *are* involved in creating semantic representations, they do so by reprocessing input to memory and that the resultant semantic representations are similar to what is accomplished by SVD. Second, as was discussed above, they claimed that semantic representations are formed from the higher order associations that exist between words. In what follows, I will describe a basic process model that uncovers higher order associations between words to form semantic representations for them. Instead of using SVD or any other dimension reduction technique, however, I will use a retrieval mechanism borrowed from a well-known account of episodic memory.

THE SEMANTICS MODEL

The model is an example of a global-matching model of memory. Global matching refers to the way in which information about a probe/cue stimulus is extracted from memory. Specifically, retrieving an item entails summing over the reactions of memory traces after they have been presented with a probe stimulus.

The model borrows its basic architecture from MINERVA 2 (Hintzman, 1984, 1986, 1988). MINERVA 2 was designed to explain memory phenomena in episodic memory tasks; this article applies the architecture to semantic memory. To the extent that the application is successful, it suggests that the primary distinction between episodic and semantic memory is not architectural but a reflection of the type of data that are stored.

MINERVA 2 postulates that every time one encounters a word, its representation is placed in lexical memory. Part of the word's representation uniquely defines the context in which the word was encountered. The term *context* is vague and, in general, refers to several circumstances in which a reader or listener encounters or processes words. In what follows, however, the issue will be finessed by operationally defining *contexts* as *documents*.

A word's context is treated as a vector of features. I will refer to the vector as the word's *context vector* (cf. Dennis & Humphreys, 2001). It uses a localist representation. For the simulation reported here, the context vector of a word encountered in a document was represented as a vector made up almost entirely of 0s, with a 1 placed at a cell location corresponding to the document number. For example, the context vector for any word that appeared in the first document was [1, 0, . . . , 0, 0]. As a word is encountered repeatedly over the set of documents, its context vector is added to that of the word in memory that it matches. Done in this way, when training is complete, the context vector for a word in memory contains the frequencies with which the word occurred in each document of the training corpus. After all the documents have been processed, memory is identical to the term × document matrix used by LSA.

I assume that a target word is used as a cue to retrieve its associated context vector from memory (see Dennis & Humphreys, 2001, for a computational example of the idea). The semantics model uses the retrieved context vector as a probe (**P**) to construct a semantic representation of the target word. The vector is applied to, and resonates with, the context vector of each trace in lexical memory (**T**). As in MINERVA 2, the extent to which a context vector in memory resonates with, or is activated by, the probe is a function of their similarity. Similarity (*S*) in the semantics model is measured as the cosine between the two vectors:

$$S = \frac{\sum_{i=1}^{N} P_i \times T_i}{\sqrt{\sum_{i=1}^{N} P_i^2} \times \sqrt{\sum_{i=1}^{N} T_i^2}}. \quad (1)$$

The N represents the number of elements contained in the **P** and **T** vectors. The formula for similarity differs slightly from the one used in MINERVA 2. Both models use the dot product between two vectors as the numerator of the equation. The denominator of the equation is different across the two models, because they use different ways to represent knowledge. MINERVA 2 represents items as vectors containing the integers, −1, 0, and 1. To normalize the similarity metric, Hintzman (1984, 1986, 1988) divides the dot product by the number of nonzero feature pairs. The semantics model represents an item as a vector of frequencies; hence, the same strategy will not work.

In MINERVA 2, a trace's activation is measured as its similarity to the probe raised to an exponent. In most published simulations that have used MINERVA 2, activation (A) is defined as

$$A = S^3. \quad (2)$$

The purpose of raising S to an exponent, instead of using S itself as a measure of activation, is to reduce the influence that traces with very low similarity to the probe have on the contents that are retrieved from memory.

The semantic model derives A differently. Following Dougherty, Gettys, and Ogden (1999), it reduces the impact of poor matches between the probe and the traces by imposing a lower limit to a trace's activation. For the simulation reported below, $A = S$. However, the similarity between the probe and a trace must reach .1 to contribute to the output of the model; otherwise, A is set to zero. The use of a cutoff was chosen for two reasons. First, the model's memory contains around 86,000 traces. Because there are so many, even if S is raised to an exponent, the impact of several thousand poor matches introduces considerable noise into the model's output. Second, from a more practical perspective, ignoring traces that contribute little or nothing to the model's output saves a great deal of simulation time.

Memory traces are then weighted by their activations and summed across the contents of memory to form a composite of the probe vector, **C**.

$$C_j = \sum_{i=1}^{M} T_{i,j} \times A_i. \quad (3)$$

The M represents the number of traces in memory. **C** is akin to MINERVA 2's calculation of what Hintzman (1984, 1986, 1988) refers to as the *echo content* from memory. The vector that is created by Equation 3 may be thought of as a semantic representation of the word. To illustrate how the model works, an example is worked out in detail in the Appendix.

MINERVA 2 was designed to describe the information that humans retrieve from episodic memory in laboratory tasks. In what follows, the basic ideas behind MINERVA 2 are applied to semantic memory. Specifically, a simulation was run to establish that semantic information about words could be constructed from contextual information stored in memory.

A SIMULATION

The model was applied to 1 year's worth of news articles from the *Sydney Morning Herald*. The corpus contains approximately two million words of text over about 38,000 articles. Before going on to discuss the simulation results, I will go over some details pertaining to the preprocessing on the term × document matrix (i.e., memory) that occurs before retrieval.

Similar to LSA, the first stage of preprocessing involved excluding terms (so-called *stop words*) on the basis of three properties:

1. *Promiscuity*. A word that occurs in almost every document carries little information about the message's topic (e.g., function words such as *the*, *is*, or *was*).

2. *Monogamy*. A word that occurs often, but only in one document, carries little information about what it could mean. In order to get a good semantic representation of a word, there needs to be variety in the contexts/documents in which it appears. In the simulation reported below, a word needed to appear in at least two contexts to be encoded.

3. *Celibacy*. A word that virtually never occurs in the corpus of text does not carry much information about what it could mean. Only words that occurred at least twice in the corpus were included.

After filtering, the resultant matrix contained 86,125 unique terms taken from 38,525 newspaper articles. Then, following Landauer and Dumais (1997), the cells of the term × document matrix were submitted to the same logarithmic transformation. Unlike in Landauer and Dumais, however, the cells were not weighted by the inverse of the term's entropy across documents.

Forty-two words representing three semantic categories (*finance*, *legal*, and *sports*) were selected. The model retrieved the semantic vector for each word. Then, by using the vector cosine as a measure of similarity, a matrix of the similarities between every possible pair of test words was constructed.

RESULTS

Following Burgess et al. (1998), the matrix of similarities was analyzed, using multidimensional scaling (MDS). MDS is a technique that reduces coordinates from high- to low-dimensional space while simultaneously attempting to maintain the appropriate distance among points. For the simulation reported here, MDS reduced the similarity matrix to two dimensions, so that the distances between words could be plotted in (x, y) coordinates and easily visualized. Figure 1 shows the solution yielded by the reduction. As is clear from the figure, words from the same semantic category are clustered close together, as compared with unrelated words. Words that are unrelated tend to be separated in semantic space.

To show that it was not the MDS itself that created the grouping of words, Figure 2 shows the average similar-

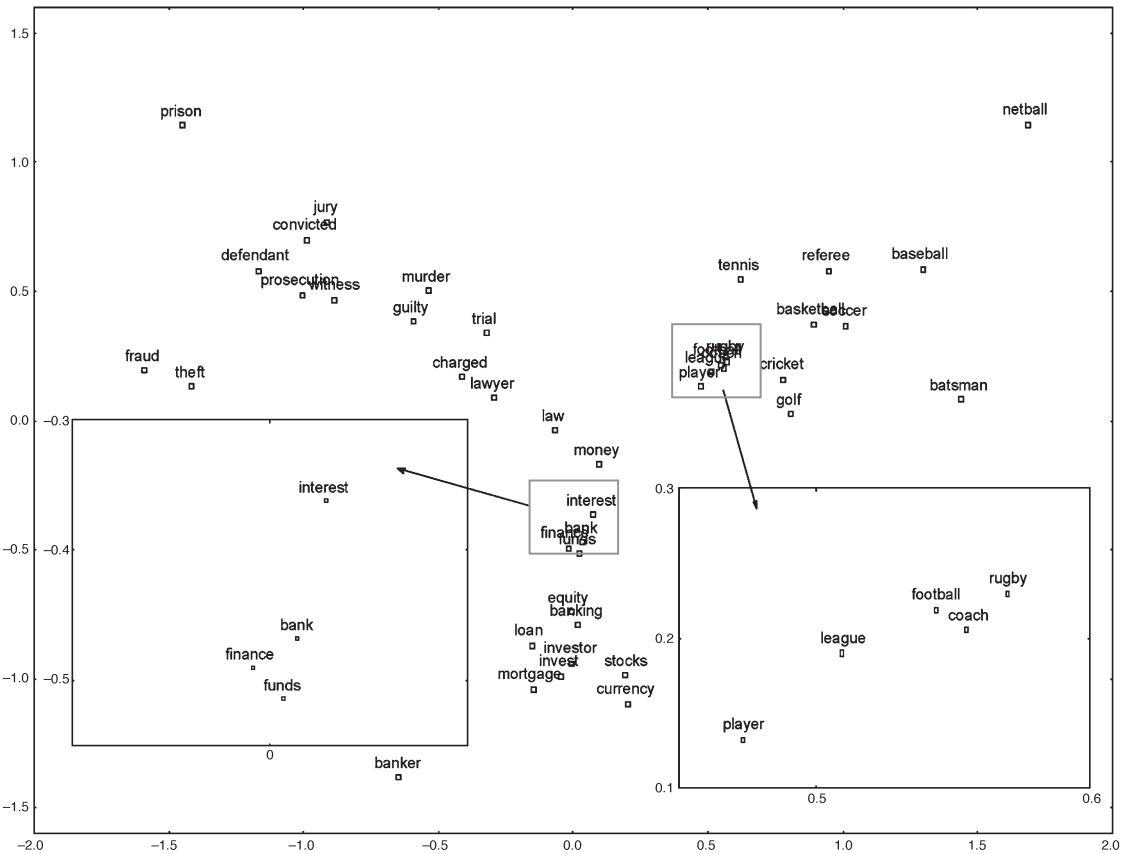

Figure 1. The two-dimensional multidimensional scaling solution for the items used in the simulation. The inserted plots are enlargements of those areas containing overprinted labels.

ity between words within and between categories. For example, the leftmost group of columns graphs the average similarity of finance terms to other finance terms (excluding word matches such as *mortgage–mortgage*), finance terms to legal terms, and finance terms to sports terms. The figure shows clearly that, on average, words within the same semantic category are much more similar than words between categories.

DISCUSSION

Two words may appear in the same context, but they may or may not be semantically related. By the same token, two words that rarely appear in the same document are not necessarily unrelated. In order to capture semantic relationships, the semantics model draws on higher order associations between words. In the example in the Appendix, *human* and *user* were considered to be related concepts, despite not occurring in the same document. Their relationship developed because the documents that they appeared in shared other words, such as *interface* and *system*. Put simply, the semantics model takes what it knows about a word's contexts and uses retrieval to estimate what other contexts might also contain the word.

Semantically related words tend to appear in the same contexts. If such raw co-occurrence information drives semantic similarity in the model, it is unclear whether the retrieval process adds anything useful. To address the issue, the similarities among all the test items' context vectors were calculated. If, for example, two words never occurred in the same document, their context vectors had a similarity of 0. On the other extreme, if two words always occurred together in the documents, their similarity was 1.

Thirty-two pairs of context vectors from related words were chosen from the test materials. Each pair of context vectors had a cosine between .01 and .03. For each of the 32 pairs of related words, an unrelated pair with the same cosine was selected. Furthermore, one member of the unrelated pair came from the same category as the words in the related pair. For example, the cosine between the context vectors for the financial terms *mortgage* and *stocks* was .01. They were yoked to an unrelated pair comprising the financial term *funds* and the sports term *soccer*, a pair whose context vectors also had a similarity of .01. The end result was a set of related word pairs and unrelated word pairs that were matched on co-occurrence similarity. When similarity based on raw co-occurrence was held

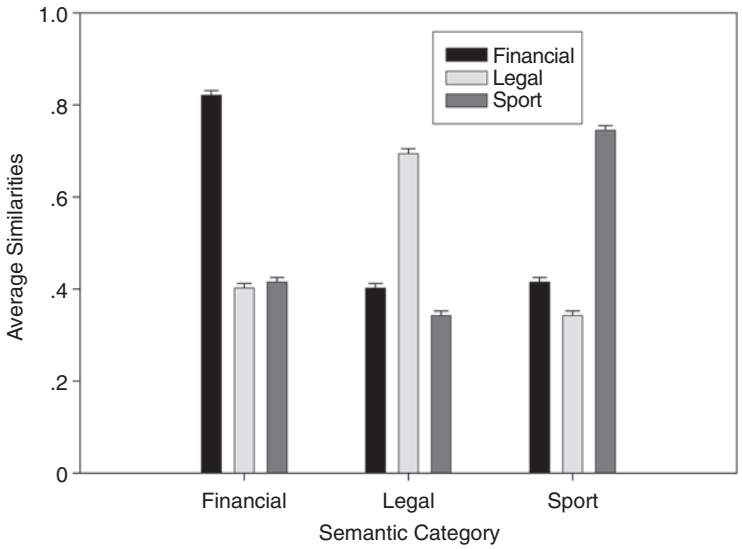

Figure 2. Average within- and between-category similarities, along with standard error bars.

constant, the semantic vectors for related words were reliably more similar to each other ($M = .64$) than those for unrelated words [$M = .45$; $F(1,31) = 12.0$, $p < .0001$]. The analysis confirms that higher order associations, not just raw co-occurrences, play a central role in driving the similarity between words' semantic vectors.

To what extent is the semantics model a process model for the ideas embodied in LSA? For one, both models describe a semantic representation in terms of the contexts in which a word is expected to appear. The important feature that they share, however, is the belief that a semantic representation is characterized by more than raw co-occurrence information. Two terms are related because, in addition to any first-order co-occurrence information, their representations include contextual information from other words that link them indirectly.

The present semantics model and LSA differ in how they gain access to higher order co-occurrence information. LSA uses dimension reduction through SVD to pull in higher order associations. Although Landauer and Dumais (1997) stated that they did not believe that the brain/mind performed SVD, they did implicate dimension reduction as the process driving the creation of semantic representations. The semantics model substitutes dimension reduction with a retrieval mechanism based on a well-known account of episodic memory. Hence, although it may be true that both the semantics model and LSA exploit higher order relationships between words to create semantic representations, they do so by implicating different types of psychological processes.

The semantics model also takes a unique perspective on the representation of semantic knowledge. LSA treats semantic representations as the result of a process that transforms "associative data into a condensed representation [that] captures indirect, higher-order associations" (Landauer & Dumais, 1997, p. 217). By contrast, the semantics model postulates that there is a limited need to represent semantic information in memory. Instead, if memory stores contextual information associated with the objects we encounter, a semantic representation for an item can be *constructed* during retrieval.

The idea that semantics are constructed, rather than stored, may serve as an explanation for how a person's own definition of a word can evolve over time. Imagine a banker who switches occupations to that of ferryboat captain. What does the word *bank* mean to that person? Before taking a job on the ferry, *bank* was associated with money, but now it is associated more with the part of the river that the ferry must avoid hitting. What changed? Over time, the frequency with which the word *bank* was used in financial contexts became overshadowed by boating contexts—a change that, in turn, perhaps influenced which sense of its constructed meaning was dominant. An appealing feature of the interpretation is that the system requires no mechanism for changing a meaning other than the addition of new contextual information to memory. Of course, a banker who becomes a ferryboat captain does not forget where to deposit a paycheck; the semantic representations for both senses of the word *bank* must still be accessible. The system just needs a way to selectively construct semantic vectors for either sense of a polysemous word.

One possible way to arrive at the appropriate meaning of a polysemous target word was suggested by Hintzman (1986). He recommended activating a subset of traces in memory that contain relevant context information (e.g., memory traces having to do with finance) by presenting the model with a related word. Then, when the system retrieves the vector for a word such as *bank*, memory traces having to do with finance and the like will contribute more heavily to the resulting semantic representation than will traces having to do with boats or rivers.

To explore the idea, semantic vectors were retrieved for the following polysemous words: *bank*, *light*, *race*, and *play*. Without reference to a specific context, the semantic representation that is retrieved from memory contains information about every sense of a word. For example, the semantic vector of *bank* should be similar to those of the associates *river* and *mortgage*. To control which sense of *bank* dominates the semantic vector, memory traces were primed by a disambiguating context vector (e.g., *boat* or *interest*) during the retrieval of *bank*'s semantic vector. In more formal terms, the semantic vector (**C**) was formed by collapsing across the memory traces after being activated by both the prime (A^{prime}) and the probe (A^{probe}) words. That is,

$$C_j = \sum_{i=1}^{M} T_{i,j} \times \left(A_i^{\text{prime}} \times A_i^{\text{probe}} \right). \quad (4)$$

The disambiguating context word was used separately in a congruent (*interest–bank* → *mortgage*) and an incongruent (*interest–bank* → *river*) case.

Table 1 contains the cosines between the semantic vectors for the target words and their associates after memory has been primed by a context word. To show that the activation of memory with the context word does not *cause* the similarity between the targets and the associates, the table also contains the similarities of the targets and the associates without memory's being primed with context words. In each case, the similarity between the target and the associate is higher when it is preceded by a congruent context word than when it is preceded by an incongruent one.

The idea that we are capable of storing contextual information associated with an item is not new. Dennis and Humphreys (2001) proposed a model of episodic memory that uses the interaction between experimental context and preexisting contextual information in memory to explain how prior experience with test materials (e.g., printed word frequency) affects performance on those items in episodic memory tasks. It is exciting to consider that the same context information that can cause interference or facilitation in an episodic memory task can also be used to build semantic representations.

Table 1
Similarities Between Polysemous Words and Two Related Associates

Target Word	Associates	No Context	Context Word	
			Boat	Interest
Bank	River	.73	.81	.67
	Mortgage	.86	.75	.89
			Dark	Heavy
Light	Bright	.93	.95	.91
	Weight	.83	.81	.84
			Color	Champion
Race	Ethnic	.54	.65	.48
	Winner	.99	.85	.96
			Director	Sport
Play	Actor	.80	.84	.75
	Baseball	.66	.54	.74

The model described here is far from a complete explanation of how people develop semantic representations. First, like LSA, the model knows nothing about morphology. It does not know, for example, that *toes* is the plural of *toe*. Indeed, apart from the information that it gets from contextual usage, the model has no way of knowing that the two are even related. Second, also like LSA, the semantic representations constructed by the model are very crude and contain no information about *how* two terms are related. For example, the semantic representations for *carburetor* and *car* may be similar, but they contain no information about the functional relationship between them.

Finally, it is worth speculating about how well the model could account for some human data. Steyvers, Shiffrin, and Nelson (2005) compared the ability of LSA's semantic vectors and those created from free association data to predict semantic similarity effects in three different memory tasks. They found that a semantic representation based on free associations (Nelson, McEvoy, & Schreiber, 1999) did the better job of predicting human performance. The model described in this article creates semantic representations that are similar in form to those in LSA: A word is treated as a list of contexts in which the word would be expected to appear. Given the similarity between LSA and the semantics model, future work should examine the extent to which the two models share the same shortcomings.

At this time, the semantic model does not possess representations for the orthography or phonology of the words it knows. As a result, a fair test of the model's ability to simulate human performance in experimental tasks, such as lexical decision or recognition memory, is beyond its capability. Kwantes and Mewhort (1999) documented some success in building a model of printed word identification from a MINERVA 2–style architecture. Given their success, future extensions of the semantics model may include orthographic and phonological representations. However, before a word identification module is joined to the semantics model, the first, necessary step will be a demonstration that a process model of semantics can be constructed using the same basic retrieval mechanism.

CONCLUSION

LSA is a powerful theory of human knowledge acquisition. It works by uncovering higher order associations between words to create semantic representations for them. The semantics model also constructs semantic representations for words by enlisting higher order information. It does so, however, by using a retrieval mechanism inspired by a well-known account of episodic memory. The retrieval mechanism was proposed as a plausible alternative to the mechanism set forth by Landauer and Dumais (1997) that, while unspecified, "reprocesses . . . input in some manner that has approximately the same effect [as SVD]" (p. 218). Apart from its alignment with LSA, the similarity between the semantic model and its episodic ancestor, MINERVA 2, suggests strongly that the same basic memory system could underlie the retrieval of both episodic and semantic knowledge.

REFERENCES

Burgess, C., Livesay, K., & Lund, K. (1998). Explorations in context space: Words, sentences, discourse. *Discourse Processes*, **25**, 211-257.

Dennis, S., & Humphreys, M. S. (2001). A context noise model of episodic word recognition. *Psychological Review*, **108**, 452-478.

Dougherty, M. R. P., Gettys, C. F., & Ogden, E. E. (1999). MINERVA-DM: A memory process model for judgments of likelihood. *Psychological Review*, **106**, 180-209.

Dumais, S. T. (1994). Latent semantic indexing (LSI) and TREC-2. In D. K. Harman (Ed.), *The Second Text Retrieval Conference (TREC-2)* (Special Publication 500-215, pp. 105-116). Washington, DC: Department of Commerce, National Institute of Standards and Technology.

Dumais, S. T., & Nielsen, J. (1992). Automating the assignment of submitted manuscripts to reviewers. In N. Belkin, P. Ingwersen, & A. M. Pejtersen (Eds.), *SIGIR'92: Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 233-244). New York: ACM Press.

Hintzman, D. L. (1984). MINERVA 2: A simulation model of human memory. *Behavior Research Methods, Instruments, & Computers*, **16**, 96-101.

Hintzman, D. L. (1986). "Schema abstraction" in a multiple-trace model. *Psychological Review*, **93**, 411-428.

Hintzman, D. L. (1988). Judgments of frequency and recognition in a multiple-trace memory model. *Psychological Review*, **95**, 528-551.

Kwantes, P. J., & Mewhort, D. J. K. (1999). Modeling lexical decision and word naming as a retrieval process. *Canadian Journal of Experimental Psychology*, **53**, 306-315.

Landauer, T. K., & Dumais, S. T. (1997). A solution to Plato's problem: The latent semantic analysis theory of the acquisition, induction, and representation of knowledge. *Psychological Review*, **104**, 211-240.

Landauer, T. K., Foltz, P. W., & Laham, D. (1998). An introduction to latent semantic analysis. *Discourse Processes*, **25**, 259-284.

Nelson, D. L., McEvoy, C. L., & Schreiber, T. A. (1999). *The University of Florida word association norms*. Available at www.usf.edu/FreeAssociation.

Steyvers, M., Shiffrin, R. M., & Nelson, D. L. (2005). Word association spaces for predicting semantic similarity effects in episodic memory. In A. F. Healy (Ed.), *Experimental cognitive psychology and its applications* (pp. 237-249). Washington, DC: American Psychological Association.

APPENDIX
A Worked-Out Example

Table A1
Raw Co-Occurrence Frequencies From the Nine Documents in the Training Corpus

Content Word	Document								
	c1	c2	c3	c4	c5	m1	m2	m3	m4
Human	1	0	0	1	0	0	0	0	0
Interface	1	0	1	0	0	0	0	0	0
Computer	1	1	0	0	0	0	0	0	0
User	0	1	1	0	1	0	0	0	0
System	0	1	1	2	0	0	0	0	0
Response	0	1	0	0	1	0	0	0	0
Time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
Survey	0	1	0	0	0	0	0	0	1
Trees	0	0	0	0	0	1	1	1	0
Graph	0	0	0	0	0	0	1	1	1
Minors	0	0	0	0	0	0	0	1	1

Consider the following nine short documents used for illustration purposes by Landauer and Dumais (1997). The first five documents have to do with human–computer interaction, and the last four have to do with mathematical graph theory.

Human machine *interface* for ABC *computer* applications
A *survey* of *user* opinion of *computer system response time*
The *EPS user interface* management *system*
System and *human system* engineering testing of *EPS*
Relation of *user* perceived *response time* to error measurement
The generation of random, binary, ordered *trees*
The intersection *graph* of paths in *trees*
Graph minors IV: Widths of *trees* and well-quasi-ordering
Graph minors: A *survey*

The matrix in Table A1 contains the frequency information for all the content words that occur in at least two documents (i.e., the italicized words in the documents). The matrix also represents the contents of memory for the semantics model. That is, each term is described in memory as a vector of frequencies across documents (a *context vector*). In the next step, the frequencies in the cells of the matrix (f) are transformed by the formula, $\ln(f + 1)$. The transformed matrix is shown in Table A2.

Consider the word *human*. After *human* is identified, we construct the semantic vector by using its context vector as a probe (**P**) to activate the contents of memory. The extent to which a trace (**T**) is activated by the probe is a function of their similarity, as calculated by Equation 1 in the text. The final column contains the activation of each trace when the context vector for *human* is used as a probe. Notice that the activation of the memory trace for *human* is 1, because the two are perfectly matched.

APPENDIX (Continued)

Table A2
Matrix in Table A1 After Its Cells Have Been Transformed

Content Word	Document c1	c2	c3	c4	c5	m1	m2	m3	m4	
Human	0.69	0	0	0.69	0	0	0	0	0	$A = 1.0$
Interface	0.69	0	0.69	0	0	0	0	0	0	$A = 0.5$
Computer	0.69	0.69	0	0	0	0	0	0	0	$A = 0.5$
User	0	0.69	0.69	0	0.69	0	0	0	0	$A = 0.0$
System	0	0.69	0.69	1.1	0	0	0	0	0	$A = 0.53$
Response	0	0.69	0	0	0.69	0	0	0	0	$A = 0.0$
Time	0	0.69	0	0	0.69	0	0	0	0	$A = 0.0$
EPS	0	0	0.69	0.69	0	0	0	0	0	$A = 0.5$
Survey	0	0.69	0	0	0	0	0	0	0.69	$A = 0.0$
Trees	0	0	0	0	0	0.69	0.69	0.69	0	$A = 0.0$
Graph	0	0	0	0	0	0	0.69	0.69	0.69	$A = 0.0$
Minors	0	0	0	0	0	0	0	0.69	0.69	$A = 0.0$

Note—The rows are context vectors (i.e., memory traces) for the 12 content words. The column on the right side contains the activation (A) of each trace when probed with the word *human*.

Table A3
Contents of Memory After Being Activated by the Context Vector, P, for the Term *Human*

Content Word	Document c1	c2	c3	c4	c5	m1	m2	m3	m4
Human	0.69	0	0	0.69	0	0	0	0	0
Interface	0.35	0	0.35	0	0	0	0	0	0
Computer	0.35	0.35	0	0	0	0	0	0	0
User	0	0	0	0	0	0	0	0	0
System	0	0.37	0.37	0.58	0	0	0	0	0
Response	0	0	0	0	0	0	0	0	0
Time	0	0	0	0	0	0	0	0	0
EPS	0	0	0.35	0.35	0	0	0	0	0
Survey	0	0	0	0	0	0	0	0	0
Trees	0	0	0	0	0	0	0	0	0
Graph	0	0	0	0	0	0	0	0	0
Minors	0	0	0	0	0	0	0	0	0

Note—The values in the cells of each trace are those of the corresponding cells in Table A2, multiplied by the A for each trace.

In the next step, the features of each trace are weighted by their activations. In the matrix shown in Table A3, each cell in a trace represents the corresponding cell from Table A2, multiplied by the trace's activation found in the final column. The semantic vector, **C**, is created by summing the activated features across all the traces, using Equation 3 in the text. Continuing with the example in which *human* is the probe, when the columns in Table A3 are summed, the resulting vector, [1.4, 0.7, 1.1, 1.6, 0.0, 0.0, 0.0, 0.0, 0.0], is the semantic representation for *human*.

Table A4 shows the semantic vectors for the terms *human*, *user*, and *minors* after separate retrievals from memory. If we measure the similarity of the semantic vectors to each other, using the vector cosine, we find that cosine(*human*, *user*) = .62 and cosine(*human*, *minor*) = .04. The semantics model deduced that *human* and *user* were related because, although the two terms never co-occurred in a document, both of them appeared in documents that used terms such as *system* and *interface*. Likewise, the system deduced that *human* and *minors* were much less related because, in addition to never co-occurring in a document, the terms are used in documents that use different types of words and, therefore, belong to different topics.

Table A4
Retrieved Semantic Vectors, C, for the Terms *Human*, *User*, and *Minors*

Content Word	Document c1	c2	c3	c4	c5	m1	m2	m3	m4
Human	1.4	0.7	1.1	1.6	0.0	0.0	0.0	0.0	0.0
User	0.6	2.8	1.6	0.9	1.8	0.0	0.0	0.0	0.3
Minors	0.0	0.3	0.0	0.0	0.0	0.3	0.8	1.5	1.6

(Manuscript received October 8, 2003;
revision accepted for publication September 29, 2004.)