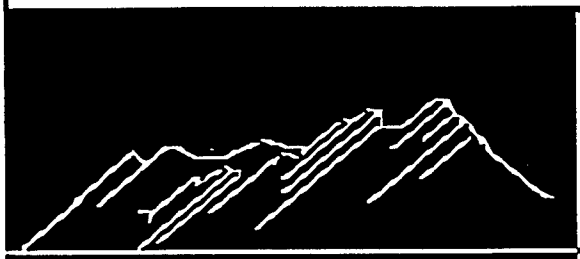
Institute of Cognitive Science



Technical Report

University of Colorado, Boulder

Predication

Walter Kintsch
wkintsch@psych.colorado.edu

Institute of Cognitive Science
University of Colorado
Boulder, CO 80309

Technical Report 99-02

Abstract

In Latent Semantic Analysis (LSA) the meaning of a word is represented as a vector in a high-dimensional semantic space. Different meanings of a word or different senses of a word are not distinguished. Instead, word senses are appropriately modified as the word is used in different contexts. In N-VP sentences, the precise meaning of the verb phrase depends on the noun it is combined with. An algorithm is described to adjust the meaning of a predicate as it is applied to different arguments. In forming a sentence meaning, not all features of a predicate are combined with the features of the argument, but only those that are appropriate to the argument. Hence, a different "sense" of a predicate emerges every time it is used in a different context. This predication algorithm is explored in the context of three different semantic problems: metaphor interpretation, causal inferences, and similarity judgments.

This research was supported by a contract from the Army Research Institute. I thank Tom Landauer, Eileen Kintsch, and the other members of the Colorado LSA Group for their comments and stimulating discussions.

Predication

Most words in most languages can be used in several different ways so that their meaning is subtly or not so subtly modified by their context. Dictionaries, therefore, distinguish multiple senses of a word. Each sense of a word is typically illustrated with an example. To demonstrate the diversity of word senses, consider this selection form Webster's Collegiate Dictionary from the 30 senses listed for the verb *run* (intransitive):

the horse runs
the ship runs before the wind
the cat ran away
the salmon run every year
my horse ran last
the bus runs between Chicago and New York
a breeze ran through the trees
a vine runs over the porch
the machine is running
the colors run
blood runs in the veins
the ship ran aground
the apples run large this year.

The meaning of the predicate *run* is different in each of these examples: *the horse runs* in a different way than *the machine* or *the colors* - and *run away* and *run aground* are different yet, although all of these uses of *run* have a core meaning in common. The exact meaning of a predicate depends on the argument it operates upon. Predication creates new meanings in every context by combining the meaning of the argument and appropriately selected aspects of the meaning of the predicate. It is not the whole meaning of *run* that applies to *the vines running over the porch*, or *the blood running in the veins*, but only features[1] that are relevant to the argument of the predication.

Multiple senses are by no means rare, especially for verbs (hundreds of senses for semantically impoverished verbs like *give* and *take* have been distinguished). Dictionaries, however, don't really claim to be exhaustive in their listing of word senses. However, George A. Miller and his colleagues, with WordNet, have made an explicit attempt to catalogue word senses for use in linguistic and psychological research, as well as for artificial intelligence applications (Fellbaum, 1998). WordNet includes over 160,000 words and over 300,000 relations among them. For instance, the verb *run* has 42 senses in WordNet; in addition, 11 senses are listed for the noun *run*. Thus, WordNet is an extremely ambitious enterprise, hand-crafted with great care. To develop a word net for the entire English language is, however, also an extraordinarily difficult task, for not only can there be no guarantee that the even most dedicated lexicographer has not missed a sense of a word or some relation between words that may suddenly become relevant,

[1] The term feature is used here in a non-technical sense.

but language change assures that new and unforeseeable word uses will forever develop. At best, such a system must remain open and continuously subject to modification.

The proposal made here is very different: there is no need to distinguish between the different senses of a word in a lexicon, and particularly the mental lexicon. The core meaning of each word in a language is well defined, but is modified in each context. Word senses emerge when words are used in certain special, typical contexts. Indeed, every context generates its own word sense. The differences between the contextual meanings of a word may be small or large, but they are always present. The decontextualized word meaning is nothing but an abstraction, though a very useful one. Specifically, in predication the meaning of the predicate is influenced by the argument of the predication.

A claim like this is of course empty unless one can specify precisely how a word meaning is defined and how it is contextually modified to give rise to various senses. Recent developments in statistical semantics have made this possible. Latent Semantic Analysis (LSA) allows us to define the meaning of words as vectors in a high-dimensional semantic space. A context-sensitive composition algorithm for combining word vectors to represent the meaning of simple sentences expressing predication will be described below.

First, LSA will be briefly introduced. Then, the predication algorithm will be discussed. Finally, a number of applications of that algorithm will be described to demonstrate that it actually performs in the way it is supposed to perform for a few important semantic problems: metaphor interpretation, causal inference, and similarity judgments.

LSA: Vectors in Semantic Space

LSA is a mathematical technique to generate a high-dimensional semantic space from the analysis of a large corpus of written text. The technique was originally developed in the context of information retrieval (Deerwester, Dumais, Furnas, Landauer, & Harshman, 1990) and was adapted for psycholinguistic analyses by Landauer and his colleagues (Landauer & Dumais, 1997; Landauer, Foltz, & Laham, 1998).

LSA must be trained with a large corpus of written text. The raw data LSA extracts from this corpus are co-occurrence counts. A matrix is constructed whose columns are words and whose rows are documents. The cells of the matrix are the frequencies with which each word occurred in each document. The data upon which the analyses reported below are based on consist of a training corpus of about 11 million words (what a typical American school child would read from grade 3 through grade 14), yielding a co-occurrence matrix of more than 92,000 word types and more than 37,000 documents. Note that LSA considers only co-occurrence; word order, syntax, or rhetorical structure are not taken into account.

Co-occurrence counts, however, are only the input to LSA which transforms these statistics into something new - a high-dimensional semantic space. LSA does this through dimension reduction. Much of the information in the original co-occurrence matrix is accidental and inessential. Why did an author choose a particular word in a specific place rather than some other alternative? Why did the author choose a particular topic for a specific document rather than something else? LSA discards all of this excess information and focuses only upon the essential semantic information in the corpus. To tell which is essential and which is distracting information, LSA uses a well-known mathematical technique called singular value decomposition, which allows it to select the most important dimensions underlying the original co-occurrence matrix, discarding the rest. The matrix is decomposed into components associated with its singular values or Eigenvalues, which are ordered according to their importance. The 300 most important components define the semantic space. The dimensionality of the space is chosen empirically: a 300-dimensional space compares best with human performance.

LSA thus makes the strong psychological claim that word meanings are represented as vectors in a semantic space of approximately 300 dimensions. But not only word meanings are represented as vectors in this space, documents are similarly represented as well. And new documents - sentences, paragraphs, whole book chapters - can also be represented as vectors in this same space. This is what makes LSA so useful. It allows us to compare arbitrary word and sentence meanings, determine how related or unrelated they are, and what other words or sentences or documents are close to them in the semantic space. A word of caution is necessary here: LSA knows only what it has been taught. If words are used that did not appear in the training corpus, or which are used differently than in the training corpus, LSA, not unlike people, does not recognize them.

The measure that is used to calculate semantic relatedness is the cosine between two vectors. As a first approximation, readers unfamiliar with this concept may think of cosines as analogous to correlation coefficients. The cosine varies from -1 to +1, +1 denoting identity and 0 denoting unrelatedness. Most cosines between words are positive, though small negative values are common (the average cosine for randomly chosen word pairs is .02, with a standard deviation of .03). The more closely two words are related semantically, the higher their cosine. For instance, the singular and plural forms of common nouns have a mean cosine of about .66, with a standard deviation of around .15.

A second measure that is often useful is the length of a vector, which, like the cosine, is defined mathematically. Intuitively, the vector length tells us how much information LSA has about this vector. Thus the length of sentence vectors is generally greater than the length of word vectors, and the length of paragraph vectors is even greater. Words that LSA knows a lot about (because they appear frequently in the training corpus, in many different contexts) have greater vector lengths than words LSA does not know well. Thus, *horse* has a vector length of 2.49, while *porch* has a vector length of.59.

All we can do, however, is compare one vector with another. Inspecting the 300 numbers that compose it tells us nothing, for the dimensions of the semantic space are not identifiable. The only way we can tell what a given vector means is to find out what other words or sentence vectors are close to it. Thus, we can ask LSA to list the words closest to a given vector in the semantic space. The semantic neighborhood of a word tells us a great deal about the word. Indeed, we shall make considerable use of semantic neighborhoods below.

Often we have some specific expectations about how a vector should be related to particular words or phrases. In such cases it is most informative to simply compute the cosine between the vector in question and the semantic landmarks we have in mind. In most of the examples discussed below when we need to determine what a vector that has been computed really means, it will be compared to such landmarks. Suppose we compute the vectors for *horse* and *porch*. To test whether what has been computed is sensible or not, we might compare these vectors to landmarks for which we have clear-cut expectations. For instance, when compared to *gallop*, *horse* should be more related than *porch* (the cosines in fact are .75. and .10, respectively), but compared to *house*, this relationship should be reversed (the cosines now are .08 and .65). This is not a very powerful test, but it is intuitively compelling and simple. What the particular landmarks are is not terribly important, as long as we have clear shared semantic expectations. Someone else might have chosen *race* instead of *gallop*, or *steps* instead of *house*, or many other similar word pairs, with qualitatively equivalent results.

Readers can make their own computations, or check the ones reported here, by using the web site of the Colorado LSA Research group: http://lsa.colorado.edu. To find the semantic neighborhood of *horse*, one types "horse" into the Nearest-Neighbor-box and selects the number of neighbors to be printed out. To find the cosine between *horse* and *porch*, one types "horse" and into one box and "porch" into the other box of the One-to-Many -Comparison .

LSA has proved to be a very powerful tool in a number of applications. To mention just three of these, there is first the use of LSA to select instructional texts that are appropriate to a student's level of background knowledge (Wolfe et al.,1998). Second, LSA has been used to provide feedback about their writing to 6th-grade students summarizing science or social science texts (in preparation). The application of LSA that has aroused the greatest interest is the use of LSA for essay grading. LSA grades certain types of essays as well and as reliably as human professionals (Landauer, Laham, Rehder, & Schreiner, 1997). The human-like performance of LSA in these areas strongly suggests that the way meaning is represented in LSA is closely related to the way humans operate. The present paper describes an LSA-based computational model which accounts for another aspect of language use, namely, how meaning can be modified contextually in predication. The model is discussed first and illustrated with some simple examples of predication. Then the model is used to simulate several more complex kinds of language processing, such as metaphorcial predication, causal inference, and similarity judgments.

LSA-Semantics: Predication

A semantic theory requires a definition of both the elements of the theory and the rules for combining these elements. The elements of an LSA-semantics are the word vectors in the semantic space. The standard composition rule for vectors in LSA has been to combine vectors by computing their centroid. Consider propositions of the form PREDICATE[ARGUMENT], where A is the vector corresponding to ARGUMENT and P is the vector corresponding to PREDICATE. According to the standard LSA practice, the meaning of the proposition is given by the centroid of A and P. In n dimensions, if A = $\{a_1, a_2, a_3, \ldots\ldots a_n\}$ and P = $\{p_1, p_2, p_3, \ldots\ldots p_n\}$, the Centroid (A,P) = $\{a_1+p_1, a_2+p_2, a_3+p_3, \ldots\ldots a_n+p_n\}$.This is unsatisfactory, in general, because P in the context of A can mean something different than P in another context, as argued above. Every time we use P in a different context A, we do not predicate all of P about A, but only a subset of properties of P that are contextually appropriate for A. This subset may be quite unusual and specific to that context (as in some of the examples above) or it may be large and diffuse, in which case the centroid may provide an adequate description of the meaning of the whole proposition.

To capture this context dependency, Kintsch (submitted) proposed an alternative composition rule, the predication algorithm. The essential feature of this algorithm is that not all features of the predicate are combined with the argument, but only those that are appropriate. This is achieved by combining LSA with the construction-integration model of text comprehension (Kintsch, 1988, 1998). Specifically, items of the semantic neighborhood of a predicate that are relevant to an argument are combined with the predicate vector, in proportion to their relevance.

The 300 numerical values of a word vector define the meaning of a word in LSA. This is a context-free definition, or rather, meaning is defined with respect to the whole training corpus. Another way of representing aspects of the meaning of a word is by looking at its neighbors in the semantic space. The closest 20 or 100 neighbors tell us something about the meaning of a word, not as much as the vector itself which positions the word in the semantic space with respect to all other words. The closest neighbors, however, index some important features of the word and contexts in which it is used. For the proposition P(A) where P is a predicate and A is an argument, the semantic neighborhood of P of size m is $\{\mathbf{P}_m\}$. $\{\mathbf{P}_m\}$ includes P. Thus, we can ask which of these neighbors are relevant to the argument A that the predicate is modifying and which are not. This involves some straightforward calculations: we compute the cosines between argument A and $\{\mathbf{P}_m\}$; the higher the cosine, the more relevant an item is to A.

At this point, the construction-integration model of Kintsch (1998) can be used to calculate the contextually modified vector for P(A). A self-inhibiting network consisting of A and $\{\mathbf{P}_m\}$ can be constructed with the links between A and the elements of $\{\mathbf{P}_m\}$ being equal to the cosine between the respective terms and inhibitory links between all elements of $\{\mathbf{P}_m\}$. All inhibitory links have equal strengths and the absolute sum of the positive and negative links is set to equal. If activation is spread in such a network, the

result will be that the terms most strongly related to A will become most highly activated, and approximately half of the terms will become completely deactivated.

The meaning of the sentence P(A) can then be computed as the weighted average of A, P and the k most highly activated terms from $\{P_m\}$, where the weights are the final activation values obtained in the spreading activation process. Note that P may have a weight of 0 if it becomes completely deactivated in the spreading activation process.

The size of the semantic neighborhood, m, varies between 20 and 100 terms. The variation in this parameter occurs because in order to select k terms that are highly relevant to both A and P, a smaller or larger neighborhood must be searched, depending on how closely related A and P are. Thus, for most sentences combining familiar terms in expected ways, m = 20 works well, because terms related to A will be found even among the closest neighbors of P. For metaphors, on the other hand, where the predicate and argument can be quite unrelated, the crucial terms are usually not found among the top 20 neighbors of the predicate and m needs to be larger, like100. A neighborhood of 500, on the other hand, is too large: the terms selected from such a large neighborhood by the predication algorithm may only have a quite tenuous relationship to P and hence misrepresent it. The parameter k, the number of argument relevant terms selected from the semantic neighborhood of the predicate, similarly must neither be too small nor too large. If too few terms are selected, a relevant feature might be missed; if too many terms are selected, irrelevant features will be introduced. Values between k = 1 and k = 5 seem most appropriate. When processing is more superficial, as in the similarity judgments discussed below, k = 1 gives the best results. If deeper understanding is required, k-values of 3 or 5 appear optimal. Selecting more than 5 terms introduces unwanted noise.

In the calculations reported below, a computational approximation has been used. In computing the final sentence vector from A, P, and the k selected neighbors of P, instead of weighting these terms by their activation values, all terms are weighted equally. Since only small values of k are used and the differences in activation among the top few terms are usually not dramatic, this computational shortcut has little effect. It greatly simplifies the calculation of predicate vectors, however. Since the most highly activated terms in the neighborhood of a predicate are those with the highest cosine to the argument, one merely has to add the k top-ranked terms to A and P.

Some Simple Examples of Predication

How satisfactory is the proposed predication algorithm? It is difficult to give a strong answer to this question. If there existed a closed set of sentences corresponding to P(A) propositions, one could obtain a random sample, compute the corresponding vectors, and find some way to compare the result with our intuitions about the meaning of these sentences. Instead, all that can be done is to show for a few simple sample sentences that the predication algorithm indeed yields intuitively sensible results, and then focus on some semantic problems that provide a more demanding test . Metaphor interpretation and causal inferences are two such domains which allow a more specific

and less arbitrary evaluation, although we are still pitting LSA calculations against human intuitions. In a third domain, that of similarity judgments, a comparison will be made between a set of experimental data and LSA predictions.

As an example of simple predication, consider the vectors corresponding to *the horse ran* and *the color ran*. First, the closest 20 neighbors to *ran* in the LSA space are calculated.[2] The cosines between these 20 terms and *horse* and *color*, respectively, are calculated. A net is then constructed linking *horse*, respectively *color,* with the 20 neighbors of *ran,* with link strength equal to the cosine between each pair of terms. These networks are integrated, resulting in final activation values for each of the 20 neighbors. These calculations are summarized in Table 1.

neighbors of *ran*	**cosine *neighbor: horse***	**activation (*horse*)**	**cosine *neighbor: color***	**activation (*color*)**
ran	0.21	0.46	0.08	0.24
jumped	0.17	0.09	0.06	0.00
yelled	0.09	0.00	0.04	0.00
stopped	0.21	0.46	0.06	0.00
went	0.16	0.00	0.07	0.09
shouted	0.16	0.00	0.07	0.09
running	0.17	0.09	0.04	0.00
hid	0.16	0.00	0.04	0.00
cried	0.14	0.00	0.05	0.00
grabbed	0.14	0.00	0.03	0.00
saw	0.19	0.28	0.07	0.09
screamed	0.11	0.00	0.05	0.00
hurry	0.11	0.00	0.08	0.24
looked	0.15	0.00	0.09	0.39
yell	0.20	0.37	0.05	0.00
came	0.21	0.46	0.08	0.24
raced	0.19	0.28	0.06	0.00
rushed	0.13	0.00	0.09	0.39
down	0.18	0.19	0.11	0.70
hopped	0.12	0.00	0.02	0.00

Table 1. The 20-term neighborhood of *ran*, with cosines and activation values for *horse* and *color*.

The vector for *The horse ran* computed by predication is therefore the centroid of *horse, ran*, and the 5 most highly activated terms from the neighborhood of run (column

[2] All calculations are based on the TASA-All space with 300 dimension; term-to-term comparisons are used.

3 Table 1), which are *ran , stopped, yell, come* and *saw.* The vector representing the meaning of *The color ran* is obtained in the same way: it is centroid of *color, ran*, and *down, looked, rushed, hurry* and *came.*

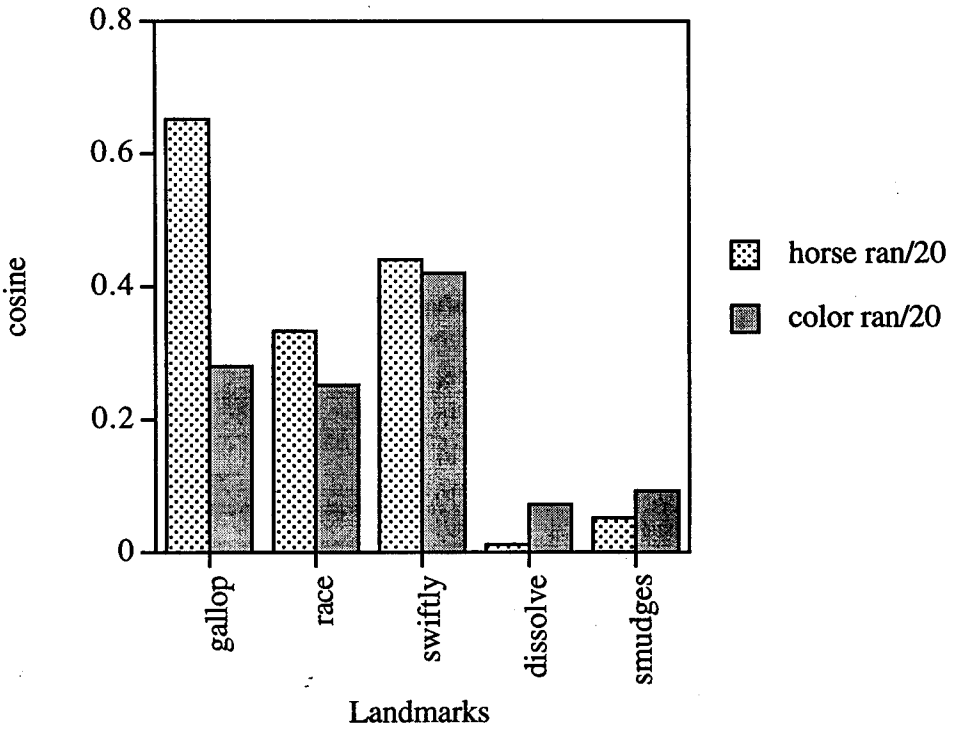


Figure 1a. The meaning *of horse ran* and *color ran* compared to five landmarks; predication is based on a neighborhood of 20.

To interpret the meaning of these vectors, they are compared to appropriate landmarks. Landmarks need to be chosen so as to highlight the intuitively important features of the sentence. In this example, we want something that is intuitively close to *horse ran* but not to *color ran. Gallop* and *race* serve that purpose. We also need landmarks that are close to *color ran* but not to *horse ran,* such as *dissolve* and *smudges*; finally the landmark *swiftly* was chosen with the expectation that it would be fairly closely related to both sentences.

The cosines between the two sentence meanings and these landmarks are plotted in Figure 1a. As expected, *horse ran* has a high cosine with *gallop, race* and *swiftly,* and

a low cosine with *dissolve* and *smudges*. The cosine between *color ran* and *dissolve* and *smudges* is higher, also as expected. However, the cosine between *gallop* and *race* and *color ran* is even higher - indicating that LSA has not formed a very precise meaning of *color ran* - *ran* is still too strongly associated with *gallop* and *race*, even in the context of *color*.

One reason this might be the case is that LSA just doesn't know much about colors running, and that is as good as it can do. It is, however, also possible that by considering only the 20 closest neighbors of *ran* in the computations, we have not allowed the differences between *horse ran* and *color ran* to emerge. Hence these computations were repeated using the neighborhood of 100 closest terms to *ran* as the starting point. The results are shown in Figure 1b. One could argue that the differences we are looking for are a little clearer in this figure than before, but over-all the picture is much the same. In fact, the cosine values in Figures 1a and 1b are almost perfectly correlated, r = .96.

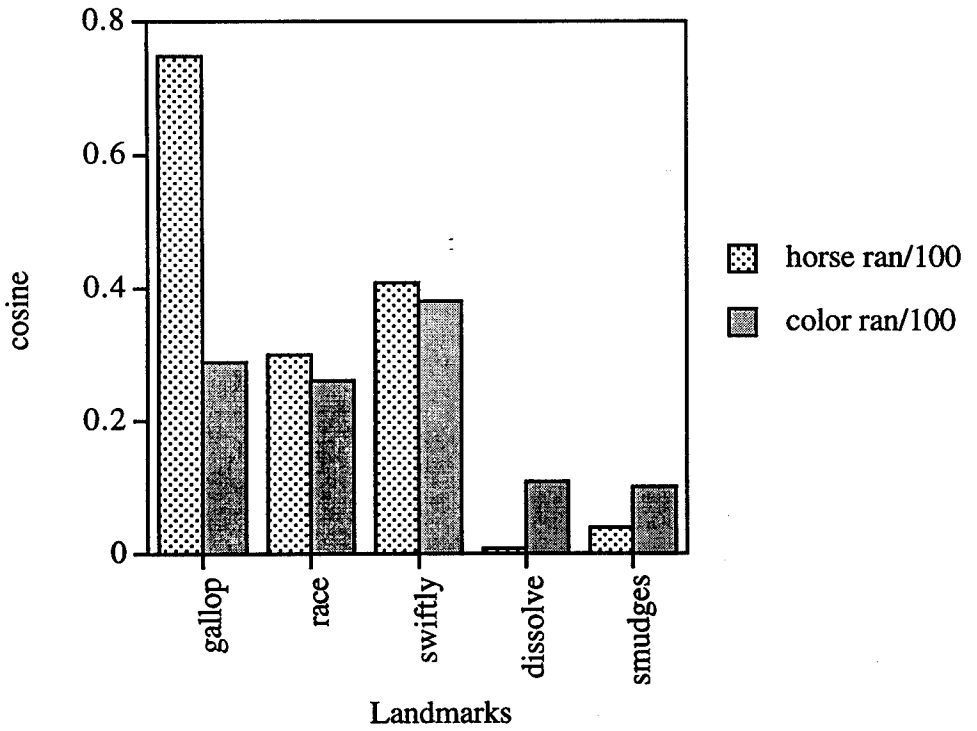


Figure 1b. The meaning *of horse ran* and *color ran* compared to five landmarks; predication is based on a neighborhood of 100.

Obviously, the size of the neighborhood in these computations makes very little difference in this example. Indeed, if we simply take the centroid of *horse ran* and *color*

ran as the meaning representation of these two sentences, our results with respect to the landmarks chosen here do not change dramatically, as seen in Figure 1c. Since the centroid is defined with respect to the total LSA space, this is somewhat analogous to using an infinite neighborhood - all the terms in the LSA space - in computing the predication. However, the cosine between *the horse ran* and *the color ran* is .66 using predication, compared to .32 using centroids, so predication and centroid are by no means equivalent. It is just that with respect to the landmarks chosen in Figure 1, the differences are relatively small. Predication gives more emphasis to the predicate whereas the centroid is more influenced by the difference in the arguments.

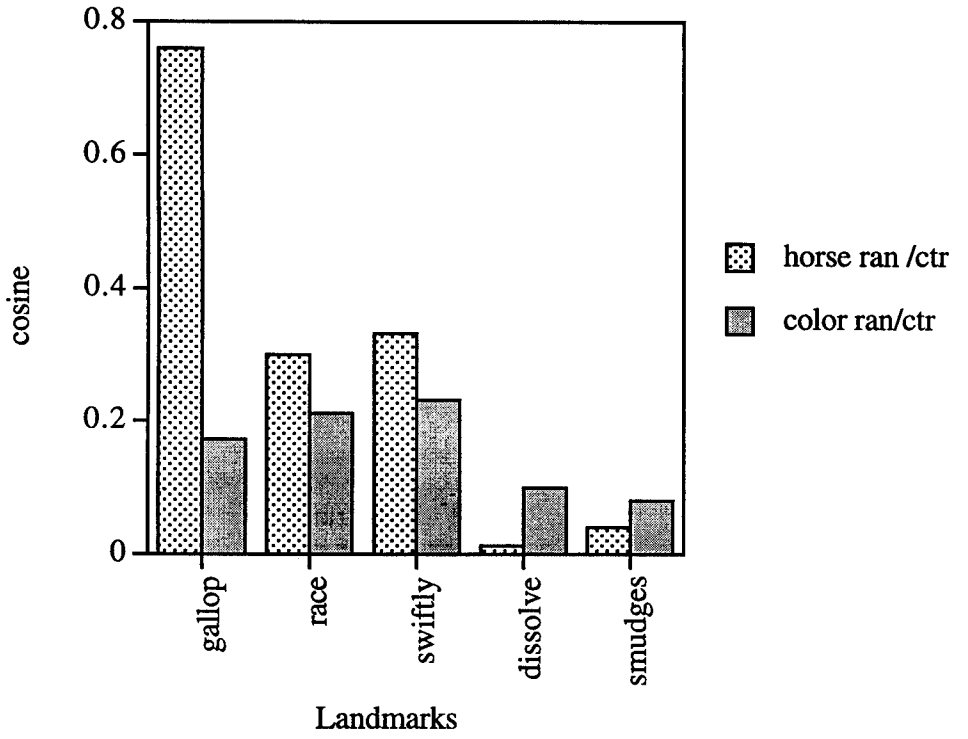


Figure 1c. The meaning *of horse ran* and *color ran* compared to five landmarks; the centroid is used to represent the meaning of the sentences.

Figure 2 compares the meaning of the sentences *the bridge collapsed, the plans collapsed,* and *the runner collapsed* . The landmarks were chosen in such a way that each sentence should be closest to one of the landmarks. The results confirm these expectations. The landmark *break down* is closest to *the bridge collapsed.* Appropriately, *plans collapsed* is closest to *failure.* For the *race* landmark, *runner collapsed* is closest. Thus, these results agree reasonably well with our intuitions about what these sentences mean. This is not the case when the sentence vectors are computed as simply the centroid of the subject and verb. In that case, for instance, *break down* is approxim tely equidistant to all three sentences.

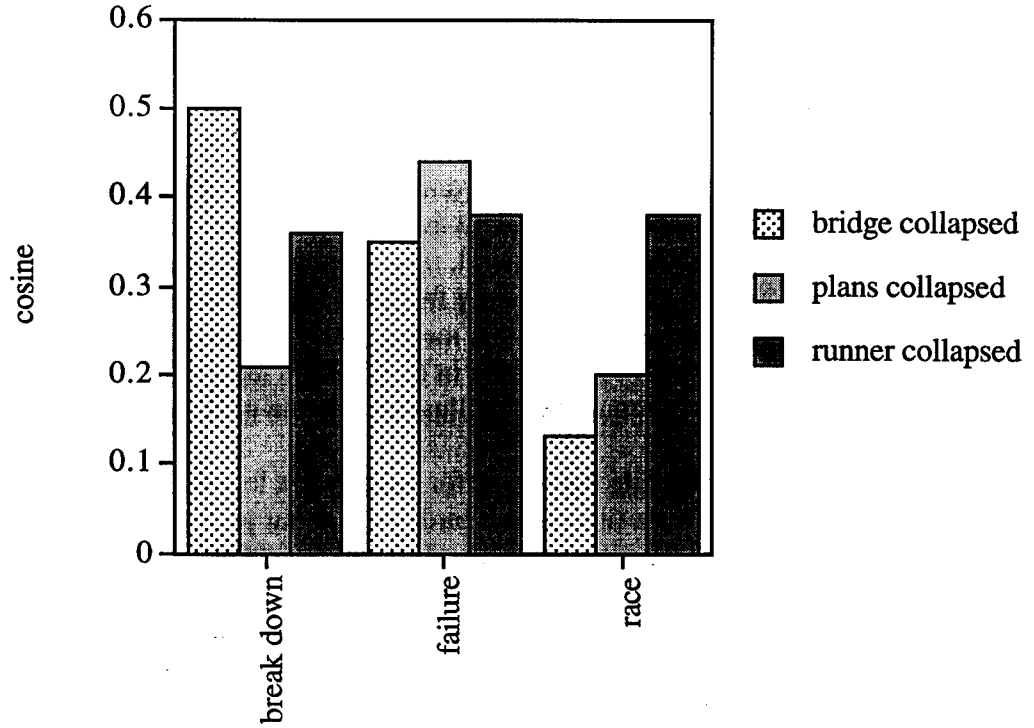


Figure 2. Three sentences with the predicate *collapsed* compared to landmarks.

The computation of sentence vectors by predication, or for that matter, by the centroid method, depends not only on the cosine between the word vectors, but also on how much information LSA has about these words. Technically speaking, a resultant vector is not only determined by the angle between its components in multi-dimensional space, but also on the length of the component vectors. Longer vectors have a greater influence on the centroid than shorter vectors. This is readily apparent in two dimensions, where it is a direct consequence of the law of parallelograms:

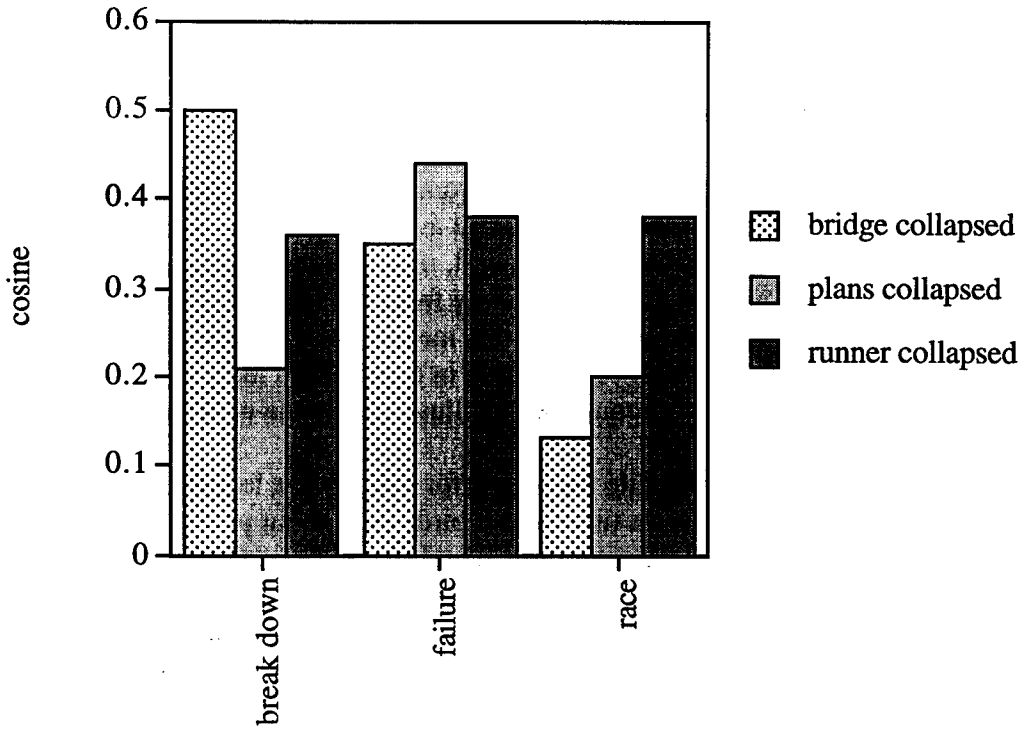
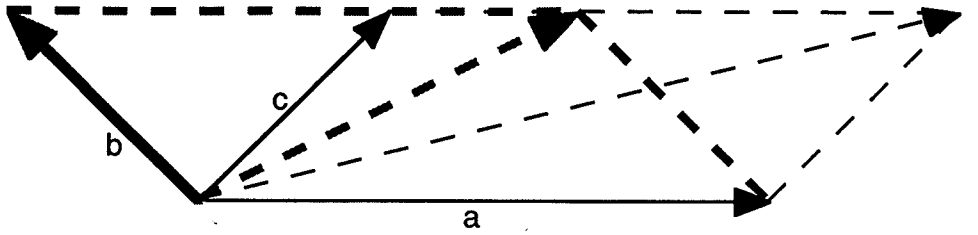


Figure 2. Three sentences with the predicate *collapsed* compared to landmarks.

The computation of sentence vectors by predication, or for that matter, by the centroid method, depends not only on the cosine between the word vectors, but also on how much information LSA has about these words. Technically speaking, a resultant vector is not only determined by the angle between its components in multi-dimensional space, but also on the length of the component vectors. Longer vectors have a greater influence on the centroid than shorter vectors. This is readily apparent in two dimensions, where it is a direct consequence of the law of parallelograms:



The resultant of vector a and b is not that different from that of a and c in spite of the fact that the angle between a and b is three times as large as the angle between a and c, because a is three times as long as either b or c. In general, if two vectors differ by a factor of k, the maximum angle the resultant can move away from the longer vector is proportional to 1/k in two dimensions. In terms of LSA this means that if we take the centroid of two terms of unequal length, it will be weighted in favor of the term with the greater vector length. This has important consequences, as illustrated in the next example.

The vector for *bird* has length 2.04 while the vector for *pelican* has length 0.15 - reflecting the simple fact that LSA knows a lot more about birds than about pelicans. When the two terms are combined, the longer vector completely dominates: the cosines between *bird+pelican* and the individual terms *bird* and *pelican* are 1.00 and .68, respectively. For comparison, the cosine between *bird* and *pelican* is .64. In other words, *pelican* doesn't make a dent in *bird.*

That has serious consequences for predication. Of course, the centroid does not distinguish at all between *the bird is a pelican* and *a pelican is a bird.* Predication does, but with very asymmetric results. *A pelican is a bird* turns the poor pelican into a bird, almost totally robbing it of its individuality, as shown in Figure 3. *Pelican is a bird* behaves like a *bird* with respect to the five landmarks in Figure 3 - closer to *sings beautifully* than to *eat fish* and *sea*! If we combine a short and a long vector, no matter what the direction of predication, we get back basically the long vector - if the differences in vector length are as pronounced as in the case of *bird* and *pelican,* which differ by a factor of 13.

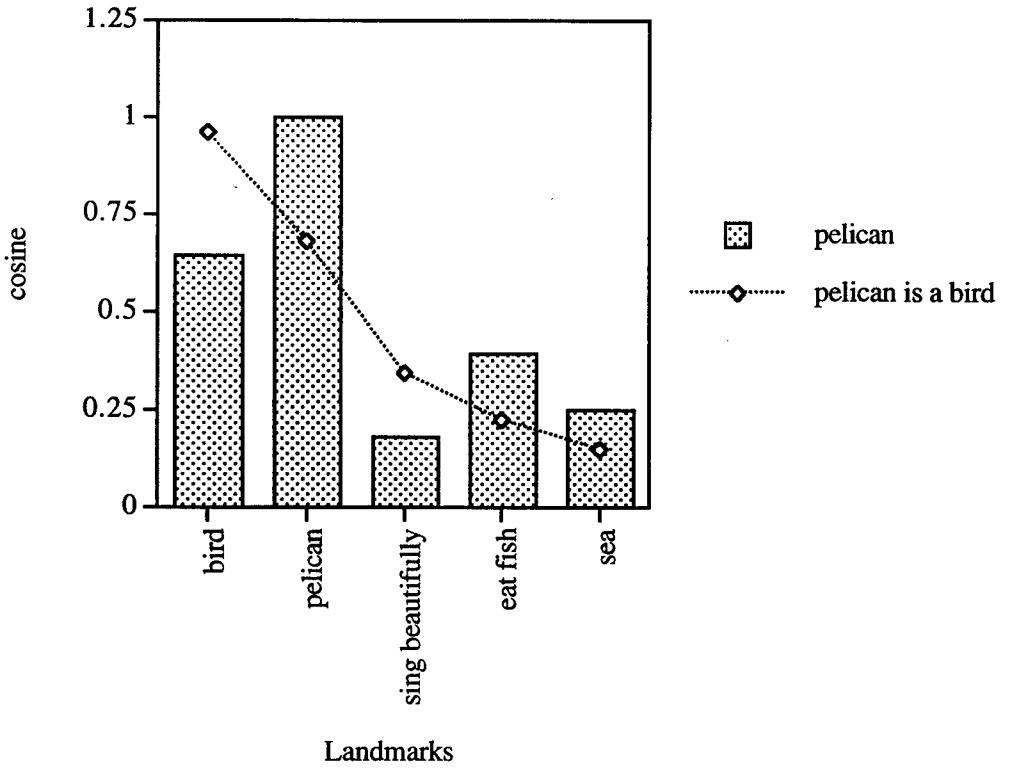


Figure 3.Cosines between the vectors for *pelican* and *pelican is a bird* and five landmarks.

Figure 4 illustrates what happens when the direction of predication is reversed. For LSA the meaning of *a bird is a pelican* is about the same as the meaning of *bird* by itself. Since LSA knows very little about pelicans, we are not adding much meaning or knowledge to bird by saying it is a pelican.

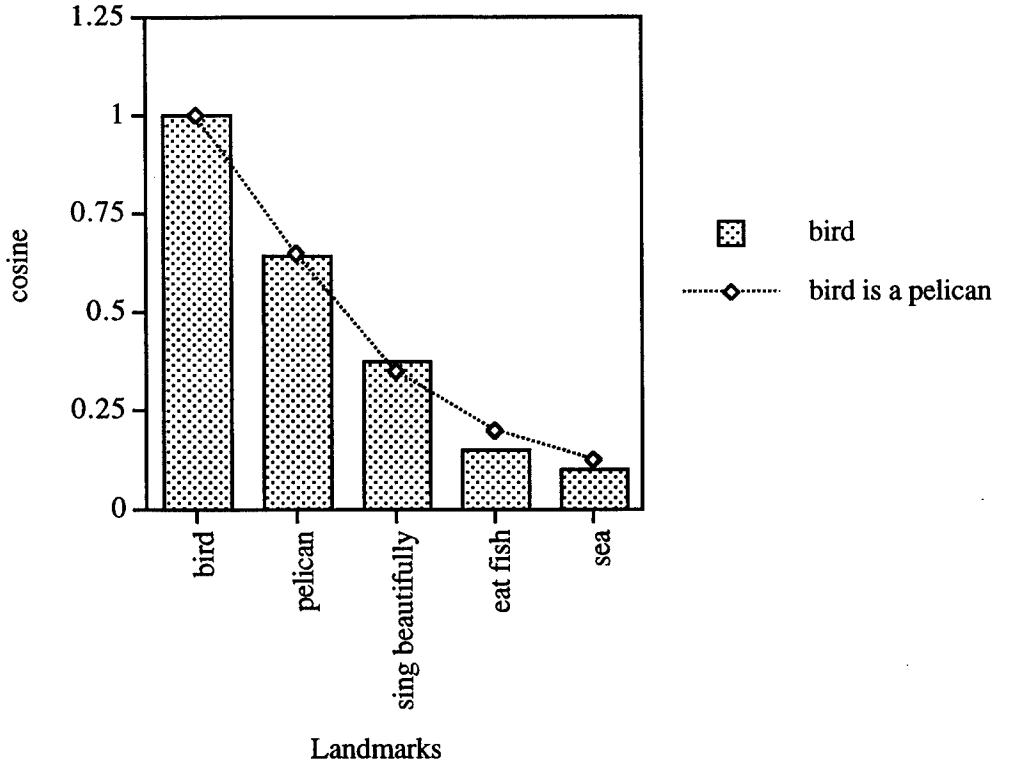


Figure 4. Cosines between the vectors for *bird* and *bird is a pelican* and five landmarks.

What happens here is exactly the opposite as in information theory. In information theory terms, to say that *pelican is a bird* provides no information at all, because we already know that. But to say *bird is a pelican* is very rich in information because it excludes numerous other possibilities. LSA does not measure information, but knowledge; not the new information provided by a sentence, but what we already know about it. About a pelican we do not know much more than that it is a bird; therefore the statement *bird is a pelican* does not add much to our knowledge - it eats a little bit more fish than birds do anyway, and sings a little bit less beautifully. On the other hand, *pelican is a bird* modifies our knowledge of *pelican* by emphasizing its general bird-features and de-emphasizing its individuality as a pelican. The language marks these distinctions. We say *The bird is a pelican*, providing information about some specific bird. Or we say *A pelican is a bird*, referring to the generic pelican. In the first case, we provide information, in the latter we provide knowledge. The informationally empty *The pelican is a bird*, and the epistomologially empty *A bird is a pelican* are not common linguistic expressions.

Wolfe & Kintsch (submitted) have pointed out a similar distinction between information and knowledge in a text. On the one hand, there is the new information a text provides; it is represented theoretically by the textbase. On the other hand, there is the

existing background knowledge relating the elements of the textbase in sometimes quite different ways (e.g., in a story, where unexpected things are supposed to happen, not merely what we already know). LSA provides a representation of this background knowledge. Finally, the situation model is the integration of the new information provided by the text and the knowledge that was already there.

Metaphors

As long as we are dealing with simple, familiar sentences, the results obtained with the predication algorithm are less than spectacular and often do not differ much from computations using the simpler centroid method. We need to turn to semantically more demanding cases to appreciate the full power of predication. The first of these cases is metaphor comprehension. This topic is discussed more fully in Kintsch (submitted). It will be briefly summarized here, because it is crucial for an understanding of predication.

Psychologists studying metaphor comprehension generally agree that metaphoric predication is just like any other predication in terms of the psychological processes involved. (for reviews see Glucksberg & Keysar, 1994; Glucksberg, 1998; Gibbs, 1994). Thus, we need to show that applying the predication algorithm to metaphors in exactly the same way as it is applied to other sentences yields sensible interpretations of metaphors. Kintsch (submitted) did just that. He showed that the interpretations of metaphors arrived in that way agree with our intuitions reasonably well and, furthermore, demonstrated that some of the major phenomena in the experimental literature on metaphor comprehension can be simulated in this way, as direct consequences of the predication algorithm.

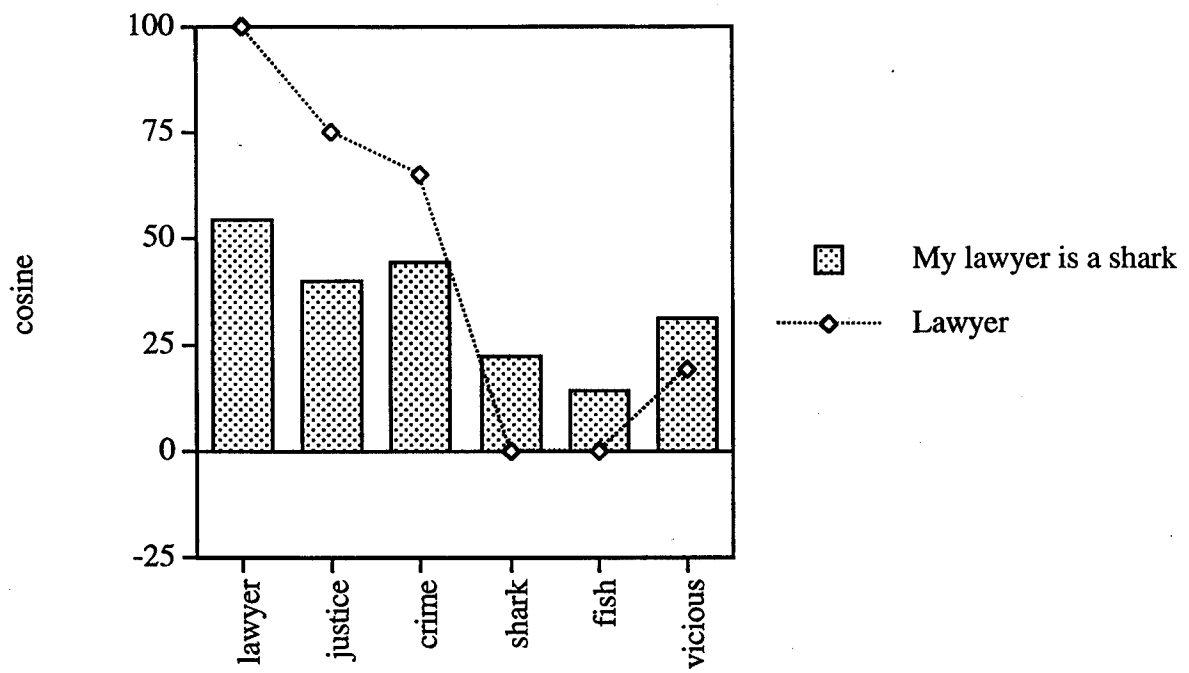


Figure 5. *My lawyer is a shark* and *lawyer* compared to six landmarks.

Glucksberg (1998) discusses in some detail the metaphor *My lawyer is a shark.* Figure 5 presents the results of a comparison of the vector for *lawyer* alone and the vector computed by predication for *My lawyer is a shark* with landmarks chosen to highlight both the relevant and irrelevant features of the metaphor. By itself, *lawyer* is strongly related to concepts like *jury* and *crime* (*justice* and *judge* and many others would have served equally well), not at all related to *shark* and *fish* (*fins* or *swim* could have been chosen as alternative landmarks), but lawyers are considered just a little bit *vicious* (once again, there is nothing special about choosing this landmark, rather than, say, *predatory* or *aggressive*). Predicating *shark* about *lawyer* changes this picture considerably. The *lawyer*-properties are de-emphasized (but still strong). The interesting thing is what happens to the *shark*-properties: *vicious* is emphasized, in agreement with most people's intuitions that *My lawyer is a shark* means something like *My lawyer is vicious.* But it does not mean exactly that, otherwise we might have said so in the first place. There is also a little bit of *shark* and *fish* in it, and if we look at *bloodthirsty*, we would see that elevated, too. Thus, the meaning of a metaphor is not fully captured by a literal paraphrase, but is richer, more expressive, and fuzzier than corresponding literal expressions.

Figure 5 totally depends on the use of the predication algorithm. If the meaning of the metaphor were computed as the centroid of the words, the results do not make sense. The centroid of *lawyer* and *shark* is somewhere in semantic no mean's land: very strongly related to *shark* and *fish* (cosines of .83 and .58, respectively), but less strongly related to *vicious* (with a cosine of .17).

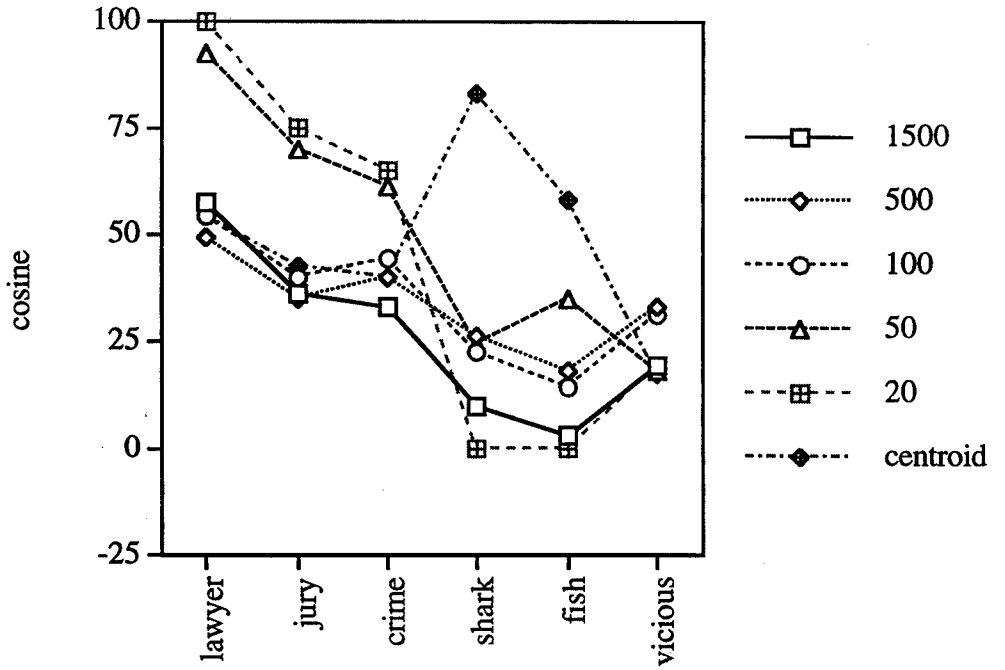


Figure 6. The predication vector for *My lawyer is a shark* computed with different values of m, the size of the semantic neighborhood of *shark*.

To compute the predication vector in Figure 5 a semantic neighborhood of m = 100 was used. When a predicate and argument are semantically related, features that are relevant to both can usually be found even with lower values of m. Thus, for the calculations in the previous section m was typically set to equal 20. For metaphors, where argument and predicate can be quite unrelated as in the present example, a larger semantic neighborhood must be searched to find three or five terms relevant for the predication. Figure 6 shows that for m = 20 and 50, predication fails (for m = 20, the concept *lawyer* is not modified at all – there are no terms among the 20 closest neighbors of shark that are semantically related to *lawyer*). For m = 100 or 500, the best results are achieved. As the neighborhood grows too large (m = 1500) the procedure picks up random noise. For comparison, the results achieved with the centroid are also shown in Figure 6.

One of the salient facts about metaphors is that they are, in general not reversible. reversed metaphors either mean something different, or they do not mean much at all. Kintsch (submitted) has shown that the predication algorithm yields results in agreement

with these intuitions. *Surgeon* is related semantically to *scalpel*, but not to *axe*; the reverse is true for *butcher*. *My surgeon is a butcher* has a cosine of .10 with *scalpel* and a cosine of .42 with *axe*; the reversed metaphor, *My butcher is a surgeon* has a cosine of .25 with *scalpel* and .26 with *axe*. On the other hand, reversing *My lawyer is a shark* does not yield any clear interpretation at all.

Two interesting phenomena about the time course of metaphor comprehension are also discussed in Kintsch (submitted). First, it has been shown (Glucksberg, McGlone, & Manfredini, 1997) that the time it takes to comprehend a metaphor is increased when the literal meaning is primed. Thus, after reading *sharks can swim*, *My lawyer is a shark* requires more time to comprehend than after a neutral prime. The literal prime activates those features of *shark* that are related to *swim*. Hence, when the metaphoric sentence is being processed, the wrong features start out with a high activation value and it takes several integration cycles to deactivate the literal features and activate the metaphoric features. As the reverse of that, a metaphoric prime can slow down the comprehension of a literal sentence (Gernsbacher, Keysar, & Robertson, 1995). If *My lawyer is a shark* precedes *sharks can swim* in a sentence verification task, verification times are longer than if a neutral prime is used. The account that the predication model gives is essentially the same as in the first case. The metaphor activates features like *vicious* and deactivates features like *fish*, so when *sharks can swim* must be verified, the wrong features are active and it requires several cycles of the integration process to deactivate these features and at the same time boost the activation of the features that are relevant to *swim*.

Thus, Kintsch (submitted) goes beyond demonstrating that the predication model yields intuitively sensible interpretations of metaphors. He also shows that some of the major phenomena about metaphor comprehension in the psycholinguistic literature are readily accounted for within that framework. This is of interest, on the one hand, because it proves that metaphor comprehension can indeed be treated in the same way as literal predication, and on the other hand, because it provides a good demonstration of how the predication algorithm extends the range of phenomena that LSA can account for.

Causal Inferences

Many sentences imply causal consequences or causal preconditions. Thus, *The doctor drank the water* implies pragmatically (though not logically) the causal precondition that *the doctor was thirsty*, and *The student washed the table* implies the causal consequence that *the table was clean*. Usually, these are described as causal inferences, though some authors, such as Kintsch (1998), argue that the term inference is misleading in this context. When we read *The student washed the table* we do not usually, in addition, draw an inference that *the table is clean*. Rather, comprehending that sentence automatically makes available this information, without any extra processing. Long-term working memory assures that there will be a link between the sentence *The student washed the table* and its anticipated causal consequence, *the table was clean*. LSA provides a computational model of how long-term working memory functions in

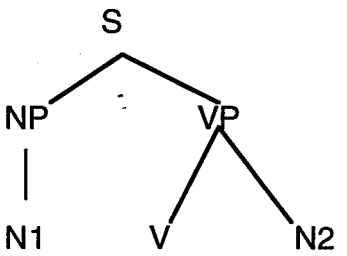
cases like these. Kintsch and his colleagues (in press; Kintsch, Patel, & Ericsson, submitted) have argued that the semantic space functions as the retrieval mechanism for working memory. Thus, if understanding *The student washed the table* involves computing its vector in the semantic space, closely related vectors such as *The table was clean* automatically become available in long-term working memory and may be subject to further processing (e.g., in a sentence verification task).

It remains to show that predication indeed delivers the right kind of results. Are sentence vectors in LSA, computed by predication, closer to causally related inferences than to causally unrelated but superficially similar sentences? Specifically, is the vector for *The student washed the table* closer to *The table was clean* than to *The student was clean*?

Predication presupposes a syntactic analysis. This syntactic analysis is not part of LSA, but one can easily imagine how an existing or future statistical syntactic parser could be combined with LSA to compute the necessary structural information. Here, the syntactic analysis is simply done by hand. We are concerned with subject-verb-object sentences, that is, propositions of the form

PREDICATE{ARGUMENT1<AGENT>, ARGUMENT2<OBJECT>}.

The corresponding syntactic structure is given by



The syntax suggests that propositions of this form involve two separate predication operations: first V is predicated about N2, in the same way as discussed for simple predication above; then VP is predicated about N1.

Specifically, in Step 1 the neighborhood of size m for the predicate V is obtained. The size of the neighborhood m = 20, unless otherwise noted. We select those terms from this neighborhood that are most relevant to N2: a network consisting of N2 and all neighbors, with link strengths equal to the cosine between N2 and each neighbor, is integrated and the k (k = 5 unless otherwise noted) terms with the highest activation values are used to approximate the vector for V-N2. In Step 2 the neighborhood is calculated for the complex predicate (V+N2), consisting of V, N2 and the k most relevant neighbors selected in Step 1. N1 is then used to determine the relevant terms from that neighborhood. The sentence vector, then, is approximated by the centroid of N1, V, N2, the k neighbors selected in Step 1, and the k neighbors selected in Step 2.

Thus, LSA, guided by a syntactic parse of the sentence, constructs a vector that represents the meaning of the proposition as a whole. To evaluate how well a predication vector captures the intuitive meaning of a proposition, causal inferences will be chosen as landmarks. For example,

The student washed the table --consequence--> *the table is clean*

or

The doctor drank the water --precondition--> *the doctor was thirsty.*

The vector representing the meaning of the sentence *the student washed the table*, computed by the predication procedure outlined above should be closer to the correct inference *the table is clean* than to the incorrect inference *the student is clean.*

As Table 2 shows, this is not generally the case when the meaning of the sentence is represented by the centroid of the three words. In fact, for the four examples analyzed here, the centroid makes the wrong inference in three cases. As we have seen above, the centroid is heavily influenced by vector length, so that semantically rich terms, like *hunter*, will always dominate semantically sparse terms, like *elk*. The predication procedure is able to overcome this bias in four of the five cases analyzed here. For instance, *the doctor drank the water* is strongly biased towards *the water was thirsty*, but predication manages to reverse that bias. Similarly for

the student washed the table --> the table was clean,
the student dropped the glass --> the glass was broken.

However, the wrong conclusion is reached for

the hunter shot the elk --> the hunter was dead.

But even where predication fails to detect the correct inference, the cosine for *the elk was dead* increased twice as much as the cosine for *the hunter was dead* as a result of predication over a centroid based comparison. Apparently predication does something right, but may fail for parametric reasons.

The student washed the table	the student was clean	**the table was clean**
centroid	.70	**.71**
predication	.62	**.83**
The student dropped the glass	the student was broken	**the glass was broken**
centroid	.76	.66
predication	.87	**.91**
The doctor drank the water	**the doctor was thirsty**	the water was thirsty
centroid	.59	*.86*
predication	**.83**	.78
The hunter shot the elk	the hunter was dead	**the elk was dead**
centroid	.66	.54
predication	.73	.70

Table 2. Cosines between five subject-verb-object sentences and causal inferences computed by centroid and predication.

There are two parameters that need to be explored: the size of a predicate neighborhood was set at m = 20, and the number of most relevant terms chosen to represent the predicate vector was set at k = 5. While a comprehensive examination of the full parameter space has not been feasible, exploratory calculations suggest that the choices of parameter values are not necessarily optimal.

The calculations for *the hunter shot the elk* were repeated with the size of the predicate neighborhood m = 100. Increasing the neighborhood size, however, did not improve the performance of LSA in this case. Indeed, the bias in favor of *hunter dead* was slightly increased: the cosine between the sentence vector computed with m = 100 and *hunter dead* turned out to be .77, versus .72 for *elk dead*. What seemed to happen was that as the number of possible selections increased, the argument could select terms it liked that were, however, too distant from the predicate. For instance, when the neighborhood of *elk shot* is so large, rather distant terms like *bow* and *arrow* can be selected because they are so close to *hunter*, biasing the meaning of the sentence in inappropriate ways.

Better results were obtained by manipulating k, the number of terms used to approximate the predication vector. A smaller value of k than 5 which was used so far might work better, because in some cases the first three of four words that were selected from a neighborhood appeared to make more sense intuitively than the last ones. Hence the computations for *the hunter shot the elk* were repeated with k = 3. This resulted in

some improvement, but not enough: the cosine between *the hunter shot the elk and hunter dead* became .69, versus .68 for *elk dead.*

Another possibility is that LSA just does not know enough about elks. If the more famioliar word *deer* is substituted for *elk*, things improve. For k = 3, we finally get

the hunter shot the deer --> the deer is dead.

The cosine between *The hunter shot the deer* and *The deer is dead* is .75, whereas the cosine with *The hunter is dead* is now only .69.

We can conclude then, that for the right parameter values the predication procedure gives the right kind of results. However, so far only a small number of examples have been analyzed. Furthermore, a systematic selection of sentences to be analyzed is needed before a thorough exploration of the parameter space would be meaningful. It appears, however, that predication may give a satisfactory account of causal inferences in comprehension.

Judgments of Similarity

Another domain where the predication model will be applied is that of similarity judgments. The cosines between concepts computed by LSA do not correlate highly with similarity judgments. Mervis et al. (1975; reprinted in Tversky & Hutchinson, 1986) reports similarity judgments for a 20 x 20 matrix of fruit names. The correlation between these judgments and the cosines computed from LSA is statistically significant, but low, r = .32. Similarly, for the data reported below in Table 4, the correlation between similarity judgments and the corresponding cosines is r = .33. These results appear to be representative.[3] In fact, there is no reason to believe that these correlations should be higher. It has been generally recognized for some time now that similarity judgments do not directly reflect basic semantic relationships but are subject to task- and context-dependent influences. Each similarity judgment task needs to be modeled separately, taking into account its particular features and context.

It makes a difference how a comparison is made, what is the predicate and what is the argument. Tversky and Hutchinson (1993) point out that we say *Korea is like China*, but not *China is like Korea*, presumably because the latter is not very informative and thus violates Gricean maxims. The predication model provides an account for this observation. In *Korea is like China*, *Korea* is the argument and *China* the predicate; thus, the resulting vector will be made up of *Korea* plus *China-as-relevant-to-Korea* - just as *A pelican is a bird* was made up of *pelican* plus *bird-as-relevant-to-pelican.* (Obviously, *is* and *is-like* do not mean the same, but this by no means irrelevant distinction must be neglected here). On the other hand, for *China is like Korea*, we compute a vector composed of *China* and *Korea-as-relevant-to-China.* The results are quite different. To

[3] Similar low correlations are obtained for free- association matrices. For instance, r = .38 for the frequency of responses to a list of words related to *butterfly* (Deese, 1961) and the cosines between the respective words.

say *China is like Korea* is, indeed, much like *saying The bird is a pelican* - both statements are semantically uninformative! The cosine between *China and Korea-as-relevant-to-China* is .98, that is we are saying very little new when we predicate Korea about China in terms of the semantics of the two concepts. However, to say Korea is like China, yields a cosine of only .77 between *Korea* and *China-as-relevant-to-Korea* - our rich information about China modifies our concept of Korea successfully, whereas the little we know about Korea is so much like China anyway that it has not much of an impact on our concept of China.

The reason for the asymmetry in the previous example lies in the difference in the amount of knowledge LSA has about *China* and *Korea*: the vector length for the former is 3.22, versus 0.90 for the latter. When the vector length of the words being compared is more equal, the order of comparison may not make a difference. Thus, *for Buttons are like pennies* and *Pennies are like buttons*, the cosine between *buttons* and *pennies-like-buttons* is .32, which is about the same, .28, as the cosine between *pennies and buttons-like-pennies*. Even if there are differences in vector length when the words being compared are basically unrelated, order differences may be minor. For *Buttons are like credit cards* and *Credit cards are like buttons*, roughly equal cosines are obtained for the two comparisons (.04 and .07, respectively), in spite of the fact that *credit cards* has a vector length of 3.92, ten times as much as *buttons*.

The literature on similarity judgments is huge and complex and it is not at all clear at this point just which phenomena the predication model can account for and what its limits are. However, one systematic comparison with a small but interesting data set will be described here. Heit and Rubenstein (1994) report average similarity judgments for 21 comparisons with two different instructions. In one case, subjects were told to judge the similarity between a pair of animal names focusing on "anatomical and biological characteristics, such as internal organs, bones, genetics, and body chemistry". In another conditions, subjects were asked to focus on "behavioral characteristics, such as movement, eating habits, and food-gathering and hunting techniques" (p. 418). These instructions made a great deal of difference. For instance, *hawk-tiger* was judged highly similar with respect to behavior (5.72 on a 10-point scale) but not with respect to anatomy (2.29), whereas *shark-goldfish* were more similar with respect to anatomy (5.75) than with respect to behavior (3.60). Intuitively, one would expect such results - the question is whether LSA has the same intuitions or not.

In terms of the predication model, either "anatomy" or "behavior" were predicated about each animal name to be judged. What was compared was $Animal_1$-with-respect-to-behavior and $Animal_2$-with-respect-to-benavior on the one hand, and $Animal_1$-with-respect-to-anatomy and $Animal_2$-with-respect-to-anatomy on the other. Specifically, the semantic neighborhoods of both instruction sentences quoted above were determined, and the terms most relevant to the to-be-compared words were selected and combined with the word vector. Table 4 shows that LSA predicted the results of Heit and Rubenstein very well indeed. There are eight comparisons (rows 1-8 in Table 4) for which anatomical similarity was greater by at least one point than behavioral similarity. For these comparisons the cosines for the with-respect-to-anatomy comparisons were greater

(equal in one case) than those for the behavioral comparison. There were four word pairs for which the behavioral similarity was rated at least one point higher than the anatomical similarity (rows 18-21). In all these cases the cosines for the behavioral comparisons were higher than for the anatomical comparisons. On the other hand, for the nine word pairs for which the empirical results were inconclusive (average ratings differed less than one point, rows 9-17), LSA matched the direction of the difference only in 4 cases. Average results are shown in Figures 7 for the empirical data and Figure 8 for the predicted results.

			Data Anatomy	Data Behavior	cosine Anatomy	cosine Behavior
1	shark	trout	9.56	4.88	0.67	0.48
2	hawk	chicken	6.44	3.08	0.61	0.35
3	hawk	robin	7.29	4.6	0.64	0.37
4	shark	goldfish	5.75	3.6	0.46	0.38
5	mosquito	ladybug	5.43	3.53	0.22	0.15
6	bat	mouse	4.99	3.46	0.18	0.17
7	mosquito	grasshopper	4.43	3.01	0.9	0.9
8	bee	praying mantis	4.43	3.09	0.74	0.6
9	snake	turtle	4.07	3.14	0.52	0.79
10	bee	ant	5.15	4.35	0.81	0.48
11	snake	lizard	6.47	5.96	0.58	0.86
12	bat	giraffe	2.03	1.64	0.31	0.16
13	whale	bear	3.29	3.1	0.07	0.08
14	whale	tuna	5.56	5.87	0.4	0.26
15	whale	rabbit	2.51	2.83	0.03	0.12
16	snake	worm	4.9	5.25	0.41	0.58
17	bat	sparrow	4.81	5.17	0.48	0.19
18	bee	hummingbird	3.4	6.64	0.4	0.81
19	hawk	tiger	2.29	5.72	0.14	0.45
20	shark	wolf	2.32	6.08	0.14	0.25
21	mosquito	vampire bat	3.19	7.18	0.31	0.44

Table 4. Rated similarity for pairs of animal names as a function of two instructional conditions (Anatomy & Behavior); after Heit and Rubenstein (1994). Cosines are computed after predicating either "anatomy" or "behavior" about each animal name.

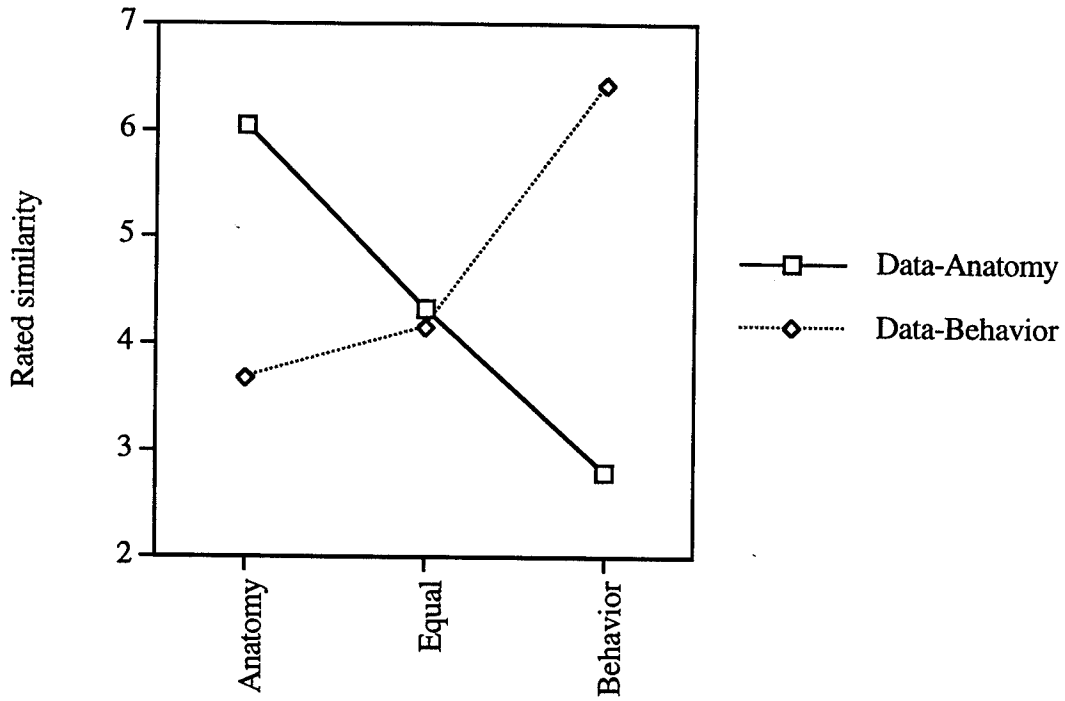


Figure 7 . Average similarity ratings between pairs of animal names for pairs rated as anatomically similar, behaviorally similar, or neutral as a function of two instructional conditions; after Heit and Rubenstein (1994).

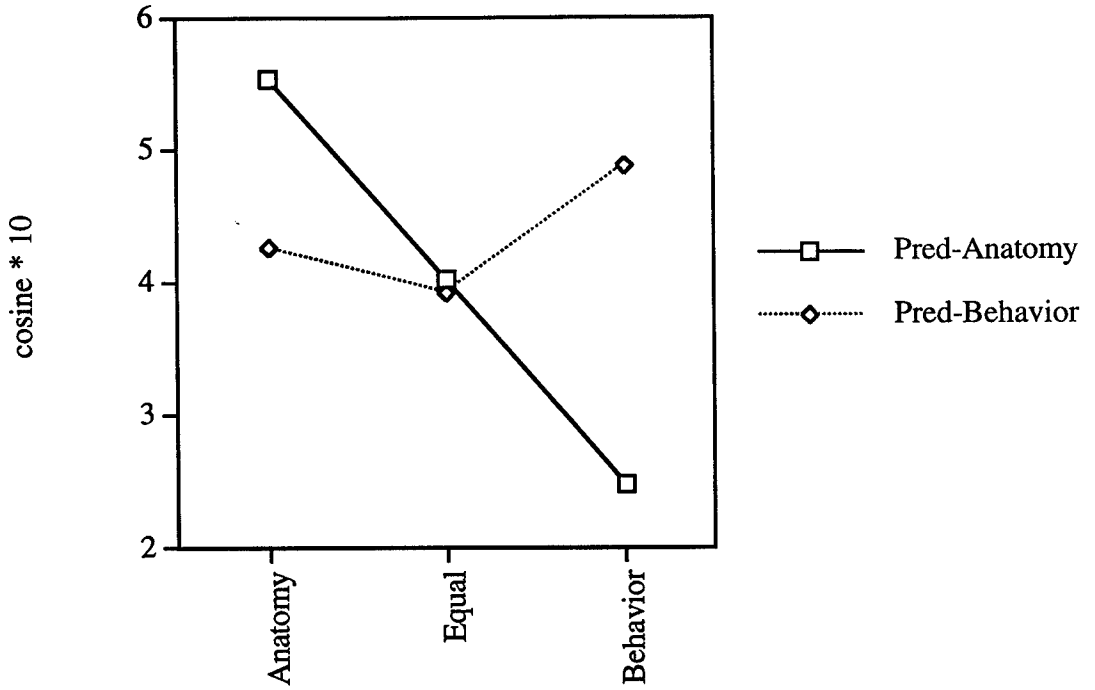


Figure 8. Average cosines between pairs of animal names for pairs rated as anatomically similar, behaviorally similar, or neutral as a function of two instructional conditions.

The predictions reported here are based on computations using a semantic neighborhood of size m = 50 and a selection of one term from that neighborhood to be combined with the vector for each word (k = 1). Larger values of k yielded somewhat less satisfactory predictions. The correlation between the rating differences Anatomy-Behavior and the corresponding cosine difference were r = .62, r = .51, and r = .40 for k = 1, 3, or 5, respectively. Calculations based on a semantic neighborhood of m = 20, however, produced poor results. Only in very few cases could something relevant to the animal names be found in the anatomy neighborhood when only 20 terms were used, so that for this choice of parameter value behavior almost completely dominated anatomy, since even in a small neighborhood of behavior terms like *eating* were to be found, which are more or less relevant to all animals.

Thus, for similarity judgements, especially when not very similar words are to be compared, such as *bat* and *giraffe*, a fairly large semantic neighborhood must be considered in predication (m = 50), but not too much from that neighborhood becomes integrated into the judgment (k = 1). For the familiar subject-verb-object sentences discussed in the previous section, on the other hand, there was no need to work with such a large neighborhood since relevant terms could reliably be found within a much smaller neighborhood (m = 20). But predication had a much greater effect there than with similarity judgments - much more information from that neighborhood appeared to be integrated into the resulting sentence vector (the most convincing results were obtained for k = 3 or 5). Metaphors were different again, in that a much larger neighborhood had

to be considered (m = 100), because the kind of argument relevant terms that predication selected from the predicate neighborhood tended to be not as strongly related to the predicate as in familiar sentences. For instance, for *My Lawyer is a shark*, most of the close neighbors of *shark* were irrelevant to *lawyer*, and one had to go way down the list to terms only moderately related to *shark* to find *lawyer*-relevant terms for the integration.

One may speculate, therefore, that the way predication is instantiated in the brain is as a parallel activation process where all neighbors sufficiently strongly related to the predicate are activated and tested for their relevance to the argument. How much of that information is then actually used in constructing the integrated sentence vector may be task dependent. When fairly deep understanding is required, as in causal inferences or metaphor understanding, quite a bit of the most relevant information from the predicate becomes integrated with the argument. On the other hand, in a more superficial task such as similarity judgment, less information from the predicate neighborhood is being used.

Conclusions

LSA-Semantics. LSA is a new theory of word meaning. It is a theory that has considerable advantages over other approaches to lexical semantics, starting with the fact that it is a completely explicit mathematical formalism that does not depend on human intervention. It has also been strikingly successful in practical applications and has provided a solution to one of the toughest previously unresolved puzzles in the psychology of language - to explain the astonishing rate of vocabulary acquisition in children (Landauer & Dumais, 1997). Nevertheless, not everyone has been willing to take LSA seriously as the basis for a semantic theory. Too many of the traditional concerns of semantics have been outside the scope of LSA. The predication algorithm that is proposed in the present paper rectifies this situation to some extent. By combining LSA with the construction-integration model LSA can be made to account for the way in which syntax determines meaning, at least for some simple, basic cases. It is not clear where the limits of the predication model are in this respect at this point. However, even if the proposed approach eventually yields a full and satisfactory account of predication, other fundamental semantic problems remain for LSA, for example, concerning the classification and distinction among such semantic relations as hypernymy and hyponymy, meronymy, antonymy and so on.

But, although LSA is still only incomplete as a semantic theory, it provides an interesting and promising alternative to the dominant conceptions of lexical semantics. Providing a computational model of how the syntactic and semantic context can modify and shape word meanings, makes it possible to think about a lexicon in which word senses do not have to be distinguished. Words in LSA are represented by a single vector in a high-dimensional semantic space, however many meanings or senses they might have. The meanings and senses emerge as a result of processing a word in its syntactic and semantic context. They are therefore infinitely sensitive to the nuances of that context - unlike predetermined definitions, that will never quite do justice to the demands

of complex linguistic contexts. Kintsch (1988, 1998) has argued that such a theory is required for discourse understanding in general; here, this argument is extended to the mental lexicon, and made precise through the computational power of LSA.

Centroid and Predication. Centroid and Predication are two different composition rules for an LSA semantics. The analyses reported here indicate that predication gives intuitively more adequate results than centroid. This is clearly so for metaphoric predicates, causal inferences, and similarity judgments, and probably so for simple predication. But if predication is the better rule, why has the centroid rule been so successful in many applications of LSA, such as essay grading?

It may be the case that the only time really important differences arise between these rules are in simple sentences out of a larger context, where specific semantic interpretations are at issue, as with metaphoric predication or causal inference. In the context of longer sentences or paragraphs, centroid and predication probably yield very similar results. The more predicates appear in a text, the more neighborhood terms are introduced, so that their effects very likely would cancel each other. Enriching semantically a brief sentence can make an appreciable difference, as was demonstrated above, but enriching every phrase and sentence in a long text probably has very little effect and may get us right back to the centroid of the terms involved.

For the fine detail, predication seems superior to centroid. But the fine detail may not weigh very much when it comes to the meaning of a longer passage, such as an essay. Even for short sentences, centroid and predication give very similar results. The vector for *the hunter shot the deer* computed by centroid and predication has a cosine of .97. Nevertheless, when we compare the centroid vector with *the hunter was dead* and *the deer was dead*, the centroid vector is much closer to the hunter being dead (cosine =.65) than the deer being dead (cosine = .35); when the vector computed by predication is compared with these inferences on the other hand, it is closer to *the deer was dead* (cosine=.75) than to *the hunter was dead* (cosine=.69). Centroid and predication are almost the same for most purposes, except when we need to make certain subtle but crucial semantic distinctions.

References

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., &. Harshman, R. (1990). Indexing by Latent semantic Analysis. *Journal of the American Society for Information Science*, 41, 391-407.

Fellbaum, C.(Ed.) (1998) WordNet: An electronic lexical database. Cambridge, England: Cambridge University Press.

Gernsbacher, M. A., Keysar, B., & Robertson, R. W. (1995). *The role of suppression in metaphor interpretation.* Paper presented at the annual meeting of the Psychonomic Society, Los Angeles.

Gibbs, R. W. Jr. (1994a). Figurative thought and figurative language. In M. A. Gernsbacher (Ed.), *Handbook of psycholinguistics* (pp. 411-446). San Diego: Academic Press.

Glucksberg, S. (1998). Understanding metaphors. *Current Directions in Psychological Science, 7*, 39-43.

Glucksberg, S & Keysar, B (1990) Understanding metaphorical comparisons: Beyond similarity. *Psychological Review*, 97, 3-18.

Glucksberg, S., McGlone, M. S., & Manfredini, D. A. (1997). Property attribution in metaphor comprehension. *Journal of Memory and Language, 36*, 50-67.

Heit, E. & Rubenstein, J. (1994) Similarity and property effec ts in inductive reasoning. *Journal of experimental Psychology: Learning, Memory, and Cognition.* 20,411-422.

Kintsch, W. (1988). The use of knowledge in discourse processing: A construction-integration model. *Psychological Review, 95*, 163-182.

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

Kintsch, W. (submitted) A computational theory of metaphor comprehension.

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

Landauer, T. K., Foltz, P., & Laham, D. (1998). An introduction to Latent Semantic Analysis. *Discourse Processes*, 25, 259-284.

Landauer, T. K., Laham, D., Rehder, B., & Schreiner, M. E., 1997). How well can passage meaning be derived without using word order? A comparison of Latent Semantic Analysis and humans. In M. G. Shafto & P. Langley (Eds.),Proceedings of the 19th annual meeting of the Cognitive Science Society (pp. 412-417). Mawhwah, NJ: Erlbaum.

Mervis, C. B., Rips, L., Rosch E., Shoben, E. J., & Smith, E. E. (1975) Relatedness of concepts. Unpublished data.

Tversky, A. & Hutchinson, J. W. (1986) Nearest neighbor analysis of psychological spaces. Psychological Review, 93, 3-22.

Wolfe, M. B., Schreiner, M. E., Rehder, R., Laham, D., Foltz, P. W., Landauer, T. K., Kintsch, W. (1998). Learning from text: Matching reader and text by Latent Semantic Analysis. *Discourse Processes*, 25,

Wolfe, M. B. & Kintsch, W. (submitted) Story Recall: Joining Comprehension Theory and Memory Theory