An Associative Model of Word Use

a dissertation proposal

Scott A. Waterman

October 18, 1994

Abstract

I present an automatic method for distinguishing the various modes of use of words by categorizing context cues, and describe a model of word usage that characterizes the various syntactic contexts in which words appear without resort to a traditional grammar or part of speech labels. The model is built inductively by identifying patterns of word configurations through observation of large text corpora. The model and the acquisition techniques are based a class of models of word use I call *associative models*. The premise for these models is that the use of a word in a particular context is licensed by associative patterns among the other words in that context and their configuration.

The model is derived by classifying the contexts of words with respect to the items they contain and their configuration as they appear in corpus. The model provides a quantitative measure of word similarity along several syntactic and semantic dimensions, as well as a corresponding hierarchical categorization of words into syntactic and semantic classes.

Applications of the model to practical NLP and language understanding tasks will be shown.

1 Introduction: Lexical Knowledge, Modeling and Acquisition

An immense amount of language data exists in a computer accessible form today. If one wants to use this data to inform the understanding of language, however, one must have a clear model on which to base one's study. Traditional rule-based grammars can be induced from corpus data (e.g. (Fu & Booth, 1975)), but if the researcher does not supply a 'backbone' of high level rules, current techniques must either posit rules in some highly constrained form (CNF, for example), or else suffer through searching an enormous combinatorial space of rules. If one is applying induction techniques in order to investigate the underlying structure of the language, a constrained normal form or a limited initial grammar may prevent discovery of interesting phenomena.

All techniques to model language rely on the presumption of *regular structure* in the language; that is, they assume that words and phrases are used with some regular correspondence to one another and to the situations in which they are used. If this regularity in the syntactic and semantic structure of language is truly present, we should be able to find it directly in the observable artifacts of language — in the spoken sounds or written words we use to communicate.

One of the main difficulties in finding these regularities, however, is the nature of the problem data itself. While linguists, being natural language users, are able to study both sides of the problem, the symbols along with their meaning, the theories and data sets have grown to such a size that many have enlisted the aid of computers to help with the task. The machines, however, are only able (at this point) to directly view one side of the problem, the symbols. Other information, such as syntactic parts-of-speech and semantic labeling, can be hand-encoded for the computer, but only at great cost to the researcher ((Johansson & Stenström, 1991; DeMarcken, 1990)). New methods and new characterizations that are capable of describing language without incorporating meaning, but which still leave the researcher avenues for meaningful interpretation will have to emerge to make this new type of language study practicable.

Many new approaches to this type of acquisition have been emerging:

- probabilistic grammars – (Jelinek et al., 1992; Skakakibara et al., 1993; Ayuso et al., 1992) This technique involves connecting a *rule probability* to the rules of a grammar system, which allows one to rank the various parses of a given input in terms of which is most likely generated by the grammar. This allows one to built large, robust grammars which will like produce multiple parses, and make sense of the output.

  These systems can be trained iteratively through the inside-outside (Baum-Welch) algorithm, which matches the grammar to a training corpus of correct usage, updating rule probabilities to increase the likelihood of generation.

  Like non-probabilistic grammars induction schemes, PCFGs must also either be constrained to a normal form or to a backbone of assumed rules. In trying to induce the structure and organization of language, there is no guarantee that these constraints will correspond with the actual structure.

- n-gram models – (Baker, 1975), (Meteer et al., 1991) n-gram models are Markov models in which a subsequence of input is used to predict the probability or interpretation of following tokens. They have been used extensively in speech recognition, where they are very effective at classifying noisy input. n-gram models have also been applied effectively to part of speech tagging.

  Besides being structurally constrained much like normal-form grammars, there are serious linguistic objections to treating language as a stochastic process. Markov models do not extend well to explain linguistic phenomena, even those as basic as subject–verb agreement.

- Collocation studies – (Smadja, 1989; Grishman & Sterling, 1992) These techniques find distributions of words in conjunction; either studying pairs or triples of adjoining words in sequence, of in particular syntactic relationships (subject–verb, modifier–noun, etc.)

  (Smadja, 1989) has introduced a surprise statistic called *mutual information* which gives a measure of the selectivity of a given collocation. This and other measures can be used to analyze ideas about semantic influences on argument selection.

- subcat distributions – (Brent, 1991; Basili et al., 1993; Grishman & Sterling, 1992) In these studies, particular verb-argument collocations are used to discover selectional characteristics of verbs.

  These approaches to studying language structure have not been extended much past studying local behavior of a few words. It would seem they can only be applied if there already exists a structural framework which presupposes particular lexical interactions.

- class-based models – (Brown et al., 1992; Basili et al., 1993; Schütze, 1993; Hughs, 1994; Finch & Chater, 1992; Futrelle & Gauch, 1993; Waterman, 1994)

  These are some of the most interesting models to appear recently. These methods use cooccurrence statistics to classify words, working toward the structuralist school's goal of substitution classes. Various comparison measures have been applied to compare the usage of words, resulting in classes divided along syntactic and semantic properties.

  Most of these studies, however, treat the usage of a word monolithically; that is, all appearances of a term are treated together to form a description of the term. The variety in usage of a term, therefore, is not open to examination by these methods. Because of this, they will fail to be able to account for novel and generative uses of terms.

  In addition, none of these methods provides explicit representations for the features that determine class membership.

1.1 Research Goals

de novo methods for acquiring:

- syntactic description of the language
- word classes — semantic syntactic categorizations of words
- word sense discrimination (discovery)
- a quantitative measure of word usage similarity

A representation capable of supporting:

- linguistic intuition
- word sense disambiguation
- an interpretive mechanism linking semantic knowledge and procedures to the syntactic model (GL cospec)

General methods.

- Classification of word use according to actual appearance in corpus.

  Occurrence in corpus is described by some limited-length context of lexical items which surround the word of interest.

  Classification methods based only on general principles of language structure, should be as language-independent as possible, and contain **no** interpretation information.

  Classification of individual words: classify the appearances of a single word to discriminate the *modes of use*

  Classification of all words: classify appearances of all word in lexis to derive syntactic categories.

2 Lexicon Acquisition and the Associative Model

The associative models are based on the supposition that a word's use is dictated and licensed by its associations with the terms with which it is used, and by association with their denoted concepts. All language description, from grammars to Markov models, have this property — grammars only allow a word to appear in those positions surrounded by rule-governed constructions. Markov models predict word appearances based on the probabilities of the preceding terms used.

Only some language descriptions, however, attempt to explicitly model the associations between a word and its context. These *lexically indexed* descriptions, such as LTAGS, link grammars, slot grammar, and word-expert parsing, model language from the perspective of the individual word and its licensed context. This word-centered approach is amenable to model acquisition through corpus studies. By observing the usage of words in corpus, one can get an appreciation of the contextual constraints and limitations that govern their use.

These lexically indexed descriptions, however, all depend on having a division of individual terms and larger construction into syntactic categories. In attempting to build such a lexically indexed model, one needs data marked with these categories in order to be effective. Such marking can only be done accurately by hand, at great expense. Further, there is no guarantee that the syntactic categories used in the marking are fully appropriate to the theoretical model being built.

I present a method for simultaneously acquiring both a lexically indexed description of language, and deriving word categories from *unmarked* corpus data. The technique works through a classification of word contexts into *context categories*, probabilistic descriptions of the context properties found in corpus.

This method is similar in principle to the class-based acquisition methods described above, but differs in that it presents categories based on the context properties found in corpus, as opposed to the distributional properties of a word's usage taken as a whole. Since a single word can be used in multiple ways (e.g. noun/verb alternations in English), being able to distinguish between the various possible uses is an important aspect of language modeling. Since I am modeling the context as a determiner of word use, the present model can distinguish the various *modes of use* of a term.

The classification of contexts can naturally be extended into a classification of the words themselves. It most cases, is is appropriate to say that words appearing in similar context will have something in common. These word classes can be derived straightforwardly by comparing the context categories associated with each word.

2.1 The purpose of classification

The intent behind such a division of words into classes comes, as it should for all classifications, from the *descriptive* and thus the *predictive* power that any representation brings to it's subject.

The descriptive power of such a representation comes from the implicit description of the syntactic properties of the language as reflected by the features measured by $s(a,b)$.

- Reduced representation By glossing these features with an abstract set of categories, the classification gives a more manageable representation than the original data, one of the principle goals of scientific investigation.

- Captures regularities The classification captures the regularities of the language w.r.t the features used in $s(a,b)$, making this higher order structure available for investigation.

The predictive power of the representation stems from the regularities expressed within it. An abstraction of this sort seeks to capture the behavior of the system, and in as much as the system has a regular behavior, to predict it's future behavior from the past. The syntactic classes serve to constrain one's expectations on the behavior of their members.

2.2 Word classes

One way of dividing the words $\mathrm{w}_1, \mathrm{w}_2, \ldots$ of a language $\mathcal{L}$ into classes $\omega_1, \omega_2, \ldots, \omega_n \in \Omega$, is to assign words to classes so that each word of a class appears in configurations that are more similar to those corresponding to words in its own class than they are to those from other classes. We can notate this classification using a *partition function*, π, where $\pi(\mathrm{w}_i) = \omega_j$ assigns each word $\mathrm{w}_i \,\epsilon\, \mathcal{L}$ to one class $\omega_j \,\epsilon\, \Omega$.

In order to quantify the notion of similarity between configurations, we introduce a *similarity measure* $s(\cdot,\cdot)$ that captures those features of the configuration that we wish to compare. In the case where we compare the *total* usage of words (all the appearances taken together), we can notate the similarity measure as $s(C(a), C(b))$, where a and b are words of $\mathcal{L}$, and $C(a)$ is some generalization of all the contexts in which a has appeared.

In order to capture the property that words of a class should be more similar to fellow class members than to outside class members, we need to compute the similarity within and between classes, s_w, s_b, respectively. These can be defined with respect to a partition π, a generalization of contexts, $C(a)$, a similarity function between contexts $s\,(C(a), C(b))$, and an averaging or generalizing function $\sum'$. This averaging function is necessary to counteract differences in size among the classes, so that large classes do not accrue greater total measures than small classes.

$$s_w^\pi(a \,\epsilon\, \omega_j) \stackrel{\text{def}}{=} \sum_{b\,\epsilon\,\omega_j}{}' s\,(C(a), C(b)) \tag{1}$$

$$s_b^\pi(a \,\epsilon\, \omega_j) \stackrel{\text{def}}{=} \sum_{b\,\epsilon\,\omega_i \neq \omega_j}{}' s\,(C(a), C(b)) \tag{2}$$

The separation of words into classes can now be stated as a constraint on the values of the between and within class similarities. For a division into n classes, $\omega_1 - \omega_n \,\epsilon\, \mathcal{C}$, we wish to find a partition π which assigns each word $\mathrm{w}_i \,\epsilon\, \mathcal{L}$ into a class ω_j such that

$$s_w(a) > s_b(a) \qquad \forall a \,\epsilon\, \mathcal{L} \tag{3}$$

One may want to find a partition $\pi_{\max}$ which maximizes the distinction between the classes. That task can be represented as finding $\pi_{\max}$ where

$$D(\pi_{\max}) = \max_\pi \sum_{a\,\epsilon\,\mathcal{L}} s_w^\pi(a) - s_b^\pi(a) \tag{4}$$

Note that the difference of within and between class similarities is not the only possible measure of class distinctions.. In general, one would want to find the maximum over some function of the within and between class similarity measures:

$$D(\pi_{\max}) = \max_\pi \sum_{a\,\epsilon\,\mathcal{L}} f(s_w^\pi(a), s_b^\pi(a)) \tag{5}$$

2.3 Diverse usage

The word-based classification just discussed has some serious drawbacks in terms of accurately reflecting the structure of a language. The most serious, perhaps, is the failure to take into account the fact that a single word can take many syntactic roles. This problem came up in the discussion of parts-of-speech, in which we saw that a single word may be used in numerous ways, and can take many different part-of-speech labels. The models discussed above, however, assign each word uniquely to a class, and do not take into consideration this diversity of usage.

In some ways, however, the word-based classifications do incorporate this diversity, but in an opaque manner. If two words exhibit nearly the *same variety* of usage, then they will be grouped together, because their total usage is similar. However, if their usage only agrees in some aspects (they take a 'from…' argument for example) but differs in others (one never appears in subject position, say, while the other does), then they will be grouped into separate classes, and the method will not discover this overlap in patterns of use. This was problem seems to be the cause of most mis-classifications in (Hughs, 1994).

There is no reason to expect that two words which are used in multiple ways will/should exhibit the *same* variety of usage. Take, for example, the words *walk* and *chair. Walk* is used as a noun, *I took a walk*, as a transitive verb *Will you walk the dog?*, and as an intransitive verb *I walk to work. Chair* is seen primarily as a noun *Please, take a chair*, but occasionally takes a transitive verb reading *Will you chair the session?* and might be seen rather infrequently as intransitive *I have to chair at 3:00.*

A word-based classifier, if it groups *walk* and *chair* together, tells us that they behave similarly, which they do. But, it does not inform us what behaviors they have in common, and especially does not tell us that they have some behaviors *not* in common. The word-based classifier implicitly tells us that they both do A, but doesn't tell us that *walk* does B while *chair* does not. If the classifier instead puts *walk* in a different class from *chair*, then we have learned that their behavior differs, but not that they have some behavior in common.

If, as linguists, we have already recognized that individual words have many meanings and various modes of use, it seems a step backward to consider each word as a monolithic entity, as with the word-based classifiers. To compare a word's total usage against that of others, without investigating some more basic structure, is to deny well accepted linguistic generalizations.

2.4 Context-based classification

What are we looking for if not word-based similarity? **Use-based** (or context-based) similarity.

If instead we could independently classify the situations in which these words are used, we could compare all the behavioral aspects the words display. By classifying the configurations in which the words are used, as a per-instance (per-use) phenomenon, rather than as a per-word property, these modes of use can be separated, identified, and studied.

We can formulate this context-based categorization in much the same way as we did for the word-based categories. First we need a representation for the context in which a word is used. For a given corpus of examples, t, which is composed of a sequence of N symbols, $t_1 - t_N$, let c_i be some characterization of the configuration surrounding the symbol t_i. In the most general sense, this would be the sequences extending to either side of t_i

$$[t_1 - t_{i-1}] \quad t_i \quad [t_{i+1} - t_n] \tag{6}$$

If we believe the claim that the most syntactic interactions are local, we can use a delimited context of n symbols to each side, instead of the entire remainder of the data. This gives the context

$$[t_{i-n} - t_{i-1}] \quad t_i \quad [t_{i+1} - t_{i+n}] \tag{7}$$

The criteria for classification remain essentially unchanged from the word-based categories. We are now looking for a partition π of the contexts, such that $\pi(c_i) = \omega_j$, and the class distinction constraints are satisfied:

$$s_w(c_i) > s_b(c_i) \qquad \forall c_i \,\epsilon\, c_1 - c_n \tag{8}$$

where the within and between class similarities are now defined in terms of similarities between individual contexts, as opposed to the total context used in the word-based measure.

$$s_w^{\pi}(c_i \,\epsilon\, \omega_j) \stackrel{\text{def}}{=} \sum_{c \,\epsilon\, \omega_j}{}' s(c_i, c) \tag{9}$$

$$s_b^{\pi}(c_i \,\epsilon\, \omega_j) \stackrel{\text{def}}{=} \sum_{c \,\epsilon\, \omega_k \neq \omega_j}{}' s(c_i, c) \tag{10}$$

2.5 Building word-level descriptions

Each context c_i represents a single use of the symbol appearing at t_i in context. By classifying the contexts, the words themselves have also been categorized in terms of the contexts in which they are used. This relation between the context-based partition and the individual words used has two natural expressions.

2.5.1 The word distribution

One is the *word distribution* of the categories. Since each of the contexts c_i grouped in a single class ω_j (that is, $c_i \,\epsilon\, \omega_j$) corresponds to a word $\mathrm{w}_k = t_i$, we can separate the contribution for each word w_k to the category ω_j as a subset of the category.

$$\omega_j/\mathrm{w}_k \stackrel{\mathrm{def}}{=} \left\{ c_i \,\epsilon\, \omega_j \;\middle|\; t_i = \mathrm{w}_k \right\} \tag{11}$$

An interesting statistic of these subsets is their proportional constituency in the category. We can represent this proportional make-up as the size of ω_j/w_k relative to the size of ω_j

$$\mathrm{w}_k(\omega_j) \stackrel{\mathrm{def}}{=} \frac{||\omega_j/\mathrm{w}_k||}{||\omega_j||} \tag{12}$$

The collection of all such proportions is the *word distribution* of the category, $W(\omega_j)$

$$W(\omega_j) \stackrel{\mathrm{def}}{=} [\mathrm{w}_1(\omega_j), \mathrm{w}_2(\omega_j), \ldots, \mathrm{w}_V(\omega_j)]^T \tag{13}$$

2.5.2 The class distribution

The inverse relation between categories and words is the *class distribution* assigned to a given word.

Since each occurrence of a word is tied to a configuration class through its context, we can describe the total usage of a word in terms of the distribution of categories that it takes.

Analogously to the word distribution of a class, we can define the proportion of a given class, ω_j, with respect to the total of all classes assigned to a given word, w_k. We can define this as the ratio of the size of the subset of all assignments of the word to the given category to the total number of appearances of the word. While this total is really $||\{t_i \,\epsilon\, t \mid t_i = \mathrm{w}_k\}||$, we can use the obvious shorthand $||\mathrm{w}_k||$.

So the subset of assignments can be defined:

$$\mathrm{w}_k/\omega_j \stackrel{\mathrm{def}}{=} \left\{ c_i \,\epsilon\, \omega_j \;\middle|\; t_i = \mathrm{w}_k \right\} \tag{14}$$

and the proportion of such assignments to the total appearances

$$\omega_j(\mathrm{w}_k) \stackrel{\mathrm{def}}{=} \frac{||\mathrm{w}_k/\omega_j||}{||\mathrm{w}_k||} \tag{15}$$

giving us the class distribution of a given word, $\Omega(w_k)$

$$\Omega(\mathrm{w}_k) \stackrel{\mathrm{def}}{=} [\omega_1(\mathrm{w}_k), \omega_2(\mathrm{w}_k), \ldots, \omega_K(\mathrm{w}_k)]^T \tag{16}$$

2.6 Limitations

Because of the nature of language, there will always be limitations on empirical studies such as this. Two sides of this issue are discussed here.

2.6.1 Sparse data.

In any data collection, there will be phenomena that are missing. This is especially true for language, as a generative and continuously changing process. There will be words that are missing from the collection, and common constructions that are simply not present.

Some words will appear only once, perhaps, and their contexts may not overlap well with those of words they share many properties with.

2.6.2 Limited data.

Text corpora present only part of the phenomena that make up language. Language is used in everyday experience to *communicate*, to exchange ideas and to make reference to the observable world. This *semantic* aspect of language is completely absent from a text corpus, and requires an agentive capacity — a connection with the real world — in order to be utilized. As this thesis describes only a representation theory, this aspect of language must be overlooked for the moment. However, as I will discuss below, I believe that this representation will be a natural and effective one to use for incorporating language knowledge in a theory of agency.

2.6.3 Computational factors.

There are also practical computational issues involved in such a study. Since most of the classification techniques available are $O(n^2)$ at best, computational resources increase dramatically with the size of the experiments. Storage and memory capacity are also a factor as the number of contexts involved increases. These factors constrain the possible size of experiments to on the order of 1e6 contexts in total.

Storage size is also affected by the context representation, increasing semi-exponentially with the number of the conditioning variables, and semi-linearly with the length of the context. In reducing context length and conditioning to meet these requirements, the affect of long-distance linguistic interactions on the classifications will be reduced.

3 Context Representations

3.1 Characterizing individual contexts

In the most general sense, the context of a word's occurrence in corpus is the remainder of the corpus, along with an indication of where the word occurred. If we treat treat a given body of text as a sequence of words, we can specify this context as

$$[t_1 - t_{i-1}] \quad t_i \quad [t_{i+1} - t_n] \tag{17}$$

Due to the size of this sequence, manipulation of this representation would be unwieldy, at best. Again, if we accept the claim that most important syntactic interactions occur locally, we can use the reduced description instead.

We can arbitrarily limit the length of the sequence to either side. An (n, m)-*delimited context*, or simply an (n, m)-*context*, is the subsequence extending n items to the left and m items to the right:

$$[t_{i-n} - t_{i-1}] \quad t_i \quad [t_{i+1} - t_{i+m}] \tag{18}$$

If the left and right sequence lengths are identical ($n = m$), then this becomes simple an n-*context*.

3.2 n-grams and n-contexts

The n-*gram* representation has long been used by the speech recognition community for phoneme and word sequence modeling, and has been adopted more recently for more general linguistic studies. There is a direct correspondence between the (n, m)-context and the $n-$gram representations.

An n-gram is an n-item subsequence of a corpus. Often, statistics on n-grams are used to estimate occurrence probabilities of words or sequences of words in language. Usually, these statistics take the form of marginal distributions on the word value of the first or last item in the sequence.

For instance a left tri-gram (3-gram) model would use all 3-item subsequences of a given body of text. This is equivalent to a $(3,0)$-context. The distribution of the first two items of each 3-gram can then be used to estimate the probability of the third:

$$\{[t_{i-2}, t_{i-1}, t_i]\}_{i>2} \qquad \Longrightarrow \qquad \Pr(t_i = w_j | t_{i-2}t_{i-1} = \mathrm{w}_a\mathrm{w}_b) \tag{19}$$

These probabilities can then be interpreted as transition probabilities on a hidden Markov model, indicating the probability of encountering a word w_j after having seen the pair $\mathrm{w}_a\mathrm{w}_b$.

Note that this Markov model can be seen as a classification of of words into categories corresponding to all pairs $\mathrm{w}_a\mathrm{w}_b$. Thus for a vocabulary of size V, we have V^2 categories named by the pairs $\omega_{a,b} \leftrightarrow \mathrm{w}_a\mathrm{w}_b$, $1 \leq a, b \leq V$, and category membership defined as the estimated probability of occurrence.

$$\omega_{a,b}(\mathrm{w}_j) = \Pr(\mathrm{w}_j | \mathrm{w}_a\mathrm{w}_b) \tag{20}$$

Note that this model requires the estimation of V^3 parameters ,V^2 probabilities for each of V words (V^n parameters in general for an n-gram model), making the process of estimation rather difficult, and making the size of the model rather cumbersome.

These implicit n-gram classes also suffer from a lack of linguistic meaning. No interesting generalization is created by identifying the words that come after a specific pair $\mathrm{w}_a\mathrm{w}_b$.

One could reverse the conditioning in interpreting the statistics, calculating instead $\Pr(t_{i-2}t_{i-1} = \mathrm{w}_a\mathrm{w}_b | t_i = \mathrm{w}_j)$. This has an classifier interpretation as V categories with V^2 members each, with membership defined as

$$\mathrm{w}_j(\mathrm{w}_a\mathrm{w}_b) = \Pr(\mathrm{w}_a\mathrm{w}_b | \mathrm{w}_j) \tag{21}$$

3.3 Characterizing multiple contexts

In order to make comparisons between all the uses of two words, as with the word-based models, or to make comparisons between contexts grouped by other principles, a representation is needed that can encompass multiple contexts.

If we want to consider all the uses of a particular word, k for instance, we can treat them as a set and simply list them. Here is such a list using α's for the left items, and β's for the right, each context identified by a superscript:

$$\begin{array}{ll} [\alpha_n^1 \ldots \alpha_2^1\, \alpha_1^1\ \mathrm{k}^1\ \beta_1^1\, \beta_2 1 \ldots \beta_m^1] & = c_k^1 \\ [\alpha_n^2 \ldots \alpha_2^2\, \alpha_1^2\ \mathrm{k}^2\ \beta_1^2\, \beta_2 2 \ldots \beta_m^2] & = c_k^2 \\ \vdots & \\ [\alpha_n^l \ldots \alpha_2^l\, \alpha_1^l\ \mathrm{k}^l\ \beta_1^l\, \beta_2 l \ldots \beta_m^l] & = c_k^l \end{array} \tag{22}$$

While this sort of representation contains all the available information, it is not well suited to the tasks of generalization and comparisons; the tabular list makes no abstractions over the original data. In order to make a comparison to a class of contexts represented this way, we would need to compare against every context listed, and average the results by some means. Analyzing the data by any means would involve looking at each element separately, and and fails to capture the abstraction desired in forming such a category in the first place.

3.4 Simply conditioned models

There are many ways to reduce the size of this set description, and make it more manageable. One family of such reduction I will call the *simply conditioned models.*

The first such model, the 0^{th}-order conditioned model, is the easiest to describe. For a set of (n, m)-contexts, each of the $n+m$ positions is represented by a vector of the probabilities of an item occurring in that position:

$$\cdots \begin{bmatrix} p(\alpha_2 = w_1) \\ p(\alpha_2 = w_2) \\ \vdots \\ p(\alpha_2 = w_3) \end{bmatrix} \begin{bmatrix} p(\alpha_1 = w_1) \\ p(\alpha_1 = w_2) \\ \vdots \\ p(\alpha_1 = w_3) \end{bmatrix} \quad k \quad \begin{bmatrix} p(\beta_1 = w_1) \\ p(\beta_1 = w_2) \\ \vdots \\ p(\beta_1 = w_3) \end{bmatrix} \begin{bmatrix} p(\beta_2 = w_1) \\ p(\beta_2 = w_2) \\ \vdots \\ p(\beta_2 = w_3) \end{bmatrix} \cdots \tag{23}$$

If we replace each of the vectors of positional probability with a simpler notation, we can write this more concisely as

$$\cdots \quad Pr(\alpha_2) \quad Pr(\alpha_1) \quad k \quad Pr(\beta_1) \quad Pr(\beta_2) \quad \cdots \tag{24}$$

Of course, this representation suffers a few problems. One is that for natural languages the number of possible words, the number of possible α's and β's, is potentially infinite; this makes the representation for a finite set of contexts an infinite array of probabilities. Even with a fixed vocabulary of size V, we need $V(n+m)$ parameters for each set of contexts. This can be remedied by including only probabilities for those items that have been seen, leaving any excluded values implicitly zero. Another approach is to only compute probabilities for some select set of items that the researcher feels has indicative power. For instance, (Schütze, 1993) uses only the 5000 most frequent words of English in his study, while (Hughs, 1994) uses only the 25 most frequent items. Other possibilities include high frequency function words, such as prepositions, case markers, and determiners.

A more serious problem is the loss of useful information in this representation. None of the probabilities given for the positional values are at all correlated in the representation. This means, for instance, that the two sets of $(0, 2)$-contexts

$$\begin{aligned} s_1 &= \{[kAB],\ [kBA]\} \\ s_2 &= \{[kAA],\ [kBB]\} \end{aligned} \tag{25}$$

will have identical representations

$$\begin{bmatrix} p(A) = .5 \\ p(B) = .5 \end{bmatrix} \quad k \quad \begin{bmatrix} p(A) = .5 \\ p(B) = .5 \end{bmatrix} \tag{26}$$

Even with this deleterious loss of correlation, this sort of representation has worked fairly well for a few word-based classifications ((Schütze, 1993),(Hughs, 1994),(Finch & Chater, 1992), (Futrelle & Gauch, 1993)).

All of these include studies of $(2, 2)$-contexts represented as 0^{th}-order conditioned models. In each study, one set of contexts was used for all the occurrences of a given word. The conditioned models for every set were compared, and classes of contexts were formed which corresponded to categories of words, all having similar contexts.

These studies used various similarity measures, such as manhattan metric ((Hughs, 1994), vector cosine measures ((Hughs, 1994), (Futrelle & Gauch, 1993)), cosine measures over a reduced space ((Schütze, 1993)), or entropy measures ((Chan & Wang, 1991). Various clustering methods were used to form categories based on these measures.

Chan and Wang (Chan & Wang, 1991) have used 0^{th}-order simply conditioned models to characterize genetic sequences, with some success. They refer to these models as *syntheses*.

3.4.1 Higher order conditioning

The information loss of this simple vector model can be reduced through conditioning the probabilities. Since I have argued that the most local interactions are the most important, it makes sense to condition the left and right sequences on the positions closest to the key word in the center.

In general, we can call a sequence conditioned on the first u positions to the left and v positions to the right a (u, v)-*simple conditioned model.* Of course, positions closer to center than u or v would be conditioned only on those positions closer to center.

Thus in representing a position, β_n, we would need to give its probabilities conditioned on the $j_{n,v} = \min(v, n-1)$ previous positions:

$$Pr(\beta_n|\beta_1\beta_2\ldots\beta_j = \mathrm{w}_1\mathrm{w}_2\ldots\mathrm{w}_j),\ j = \min(v, n-1) \tag{27}$$

(Note that this is not the standard notation used for in n-gram analyses or Markov modeling. In this notation, $Pr(\beta_2|\beta_1 = \mathrm{w}_1)$ is the distribution of the second position, β_2, only for those cases in which $\beta_1 = \mathrm{w}_1$. Similarly, $Pr(\beta_3|\beta_1 = \mathrm{w}_1)$ is the distribution of the third position, β_3, when the *first* position is $\beta_1 = \mathrm{w}_1$.)

This representation requires $V^{j_{n,v}+1}$ parameters for each position: a probability for each of V terms, each conditioned against $V^{j_{n,v}}$ possible sequences of preceding terms.

It is obvious that a set of $(u+1, v+1)$-contexts is required to build a (u, v) model.

As an example, the $(0, 1)$ and $(0, 2)$-simply conditioned models on $(0, 3)$-contexts would look as follows:

$$k \begin{bmatrix} p(\beta_1 = \mathrm{w}_1) & [Pr(\beta_2|\beta_1 = \mathrm{w}_1)\ Pr(\beta_3|\beta_1 = \mathrm{w}_1)] \\ p(\beta_1 = \mathrm{w}_2) & [Pr(\beta_2|\beta_1 = \mathrm{w}_2)\ Pr(\beta_3|\beta_1 = \mathrm{w}_2)] \\ \vdots & \end{bmatrix} \tag{28}$$

$$k \begin{bmatrix} p(\beta_1 = \mathrm{w}_1) & \begin{bmatrix} p(\beta_2 = \mathrm{w}_1|\beta_1 = \mathrm{w}_1) & Pr(\beta_3|\beta_1\beta_2) \\ p(\beta_2 = \mathrm{w}_2|\beta_1 = \mathrm{w}_1) & Pr(\beta_3|\beta_1\beta_2) \\ p(\beta_2 = \mathrm{w}_3|\beta_1 = \mathrm{w}_1) & Pr(\beta_3|\beta_1\beta_2) \\ \vdots & \end{bmatrix} \\ p(\beta_1 = w_2) & \begin{bmatrix} p(\beta_2 = \mathrm{w}_1|\beta_1 = \mathrm{w}_2) & Pr(\beta_3|\beta_1\beta_2) \\ p(\beta_2 = \mathrm{w}_2|\beta_1 = \mathrm{w}_2) & Pr(\beta_3|\beta_1\beta_2) \\ p(\beta_2 = \mathrm{w}_3|\beta_1 = \mathrm{w}_2) & Pr(\beta_3|\beta_1\beta_2) \\ \vdots & \end{bmatrix} \\ \vdots & \end{bmatrix} \tag{29}$$

3.5 Lattice conditioned models

4 Similarity measures

4.1 String Metrics

1. manhattan

2. cosine

3. entropy increase

4. edit distance

All of these can be modified to highlight the effects of locality by incorporating a weight decreasing with sequence length.

4.2 Edit Distance

One method for judging the similarity between strings of lexical items (tokens) is the *edit distance* formulated by Levenshtein ((Levenshtein,)). This is a similarity measure based on the minimum number of token insertions, deletions, and substitutions (mutations) required to transform one string into another. A generalization of this edit distance can be made by assigning differing weights to insertions of particular tokens or classes of tokens, and by also assigning weights to token substitution pairs. Straightforward computational methods for finding the edit distance between two strings ((Sellers, 1974; Wagner & Fisher, 1974)) have been used on a variety of problems in biology, genetics, speech and handwriting analysis ((Sankoff & Kruskal, 1983)), as well as in syntactic analysis of formal languages ((Lu & Fu, 1977)). (For a good introduction with applications to many domains, see (Sankoff & Kruskal, 1983).)

To demonstrate the generalized edit distance, consider the two strings:

the path that is the path
the way that is not the way

The first string can be transformed into the second by a number of insertion, deletion, and substitution operations. Substitutions are commonly counted as two operations, since they give the same effect as a deletion-insertion combination. In this example, 'not' could be inserted; 'path' could be substituted by 'way', then the second 'path' deleted at the end, then 'way' inserted; 'that' could be deleted then reinserted, and then 'not' inserted; etc. Many different sequences lead to the same result, but there will be a minimum number of operations required for the transformation.

After a short inspection, we could expect a minimum of 5 operations in this case – two for each change from 'path' to 'way', and one for the insertion of 'not'.

This distance measure can be generalized to compensate for different similarities between types of tokens. For instance, if one decides that 'way' and 'path' are more similar to each other than either is, say, to 'is' or 'the', then it would be good to have the substitution of 'path'–'way' amount to less than the possible substitution 'path'–'is'. To accomplish this, a *cost* can be associated with each operation, perhaps even a different cost for each sort of insertion or substitution. Then a transformation of *minimum cost*, rather than minimum operations, can be defined. If one makes the simple assumption that a substitution costs no more than the corresponding deletion-insertion pair, then this minimum cost can be shown to obey metric properties, and defines the *generalized edit distance* between the two strings, with larger distances corresponding to less similar strings.

There is a straightforward method for computing edit distance. In a prime example of dynamic programming, the edit distance is computed for every pair of initial substrings of the two strings under study, with results for shorter substrings combining to give results for longer substrings.

More explicitly, let our two strings be $A = (a_0, a_1, \ldots, a_m)$ and $B = (b_0, b_1, \ldots, b_n)$, where a_i is the i^{th} token in string A, starting with token 1. We let the first component of the the string, a_0, be a *null token*, representing an empty position into which we can insert.

Define also the initial substring $A_i = (a_0, a_1, \ldots, a_i)$ of a string to be the first i tokens, including the null token at the beginning.

The computation starts by assigning $D(A_0, B_0)) = 0$, the cost of transforming a_0 to b_0, the null token to itself. Each subsequent step in the computation proceeds with the simple rule:

$A \longrightarrow$

$B \downarrow$

	–	the	path	that	is	the	path
–	0	1	2	3	4	5	6
the	1	0	1	2	3	4	5
way	2	1	2	3	4	5	6
that	3	2	3	2	3	4	5
is	4	3	4	3	2	3	4
not	5	4	5	4	3	4	4
the	6	5	6	5	4	3	4
way	7	6	7	6	5	4	5

Figure 1: Dynamic programming for edit distance ('–' is the null token)

$$D(A_i, B_j) = \min \begin{cases} D(A_i, B_{j-1}) + D_{\text{insert}}(b_j) \\ D(A_{i-1}, B_j) + D_{\text{insert}}(a_i) \\ D(A_{i-1}, B_{j-1}) + D_{\text{substitute}}(a_i, b_j) \end{cases}$$

where $D_{\text{insert}}(x)$ is the cost for inserting x, and $D_{\text{substitute}}(x, y)$ is the cost of substituting x for y.

Starting with $D(0,0)$, one can fill each $D(i,j)$ in a table, ending at $D(m,n)$, the edit distance between the two strings. The table is filled from upper left to lower right, as each entry is computed from its upper, leftward, and diagonal neighbors using the minimum rule above. Figure 1 gives this table for the example strings.

4.3 String alignments

As a by-product of the edit distance computation, one can create an *alignment* of the two strings. This alignment matches the elements of the two sequences in linear order and shows the correspondence between tokens and substrings of the two matched strings. An alignment can be generated directly from the table created in the edit distance computation. (Multiple minimum paths may result, giving alternate but equivalent alignments.)

The alignment created from our two example strings (figure 2) gives the correspondence between the tokens of the two initial strings. From the figure, it is easy to see the structural similarities of the two strings.

–	the	path	that	is	–	the	path
–	the	way	that	is	not	the	way

Figure 2: A string alignment table

Alignments can be created for sets of more than two strings. These alignment tables easily be abstracted to probabilistic descriptions of the sequences, such as the simply conditioned models discussed above.

4.4 Locality in the edit distance

There is a degree to which purely configurational (syntactic) considerations are local in nature. Syntactic well-formedness and syntactic interactions are properties and behaviors that seem to have a high locality of effect. The presence of phrasal constituents in almost every syntactic theory is evidence of the degree to which this belief is held — phrasal boundaries mark the limits of local syntactic interactions for most word classes. Only some word classes, such as verbs and event-denoting nouns, seem to affect the placement and configuration of more distant constituents. Most word types seem to affect (and, conversely, are affected by) primarily the configuration in their immediate vicinity.

In order to highlight the locality of these configurational effects, the edit distance used in the experiments was modified so as to decrease the importance of token distance from the keyword. One would like to weight near tokens more heavily, but without ignoring the contributions of distant ones. A window function (sometimes called a step function) would be simplest, but would only count near tokens and completely discount far ones. A linear drop-off function would be able to include contributions of all tokens, but because some strings are very long, it would necessitate a slow drop-off if even the very distant tokens were to contribute to the measure.

In the end, a geometrically decreasing weight function was chosen, due to its useful properties:

- Near tokens are weighted more heavily than far tokens.
- All tokens in the string still contribute to the distance measure.
- A *half power* distance can be defined, which helps in the understanding and analysis of the results.

The half power distance is the distance for which the tokens on one side (those near the keyword) account for half of the total possible edit distance, while those on the other side (farther from the keyword) account for the remainder. This helps give a more intuitive reading for the resulting distance, with an effective *window* around the keyword which can be treated equally with the remainder of the string.

The implementation of this geometric drop-off requires only a small change to the original dynamic programming algorithm for edit distance. The table-filling rule becomes:

$$D(A_i, B_j) = \begin{cases} D(A_i, B_{j-1}) + L^{i+j} \times D_{\text{insert}}(b_j) \\ D(A_{i-1}, B_j) + L^{i+j} \times D_{\text{insert}}(a_i) \\ D(A_{i-1}, B_{j-1}) + L^{i+j} \times D_{\text{substitute}}(a_i, b_j) \end{cases}$$

where L is the *locality factor*, which is defined in terms of the half power distance, P_h:
$L = \frac{1}{2}^{1/P_h}$, so that $L^{P_h} = \frac{1}{2}$.

5 Classification techniques

The previous chapters have discussed methods of characterizing sets of contexts, and measures for comparing them, but the question of how one finds the sets has been shrugged off until now. This chapter discusses methods for grouping the raw lexical data into classes which have distinct and interesting properties.

5.1 Background in Clustering Methods

Clustering techniques are methods for grouping data into distinct categories. Of course, the notion of *distinct* is dependent on what you care to pay attention to, and perhaps what you are able to pay attention to. In general, clustering techniques fall into several families, among them parametric, hierarchical, and density-driven methods.

5.2 Parametric

Parametric clustering can be seen as an optimization process, in which clusters are found which maximize some fitness criterion over the data. Often, this criterion takes the form of a combination of within and between group similarity.

algorithms operate by positing a model for form of the groups or clusters to be found,

5.3 fixed

Fixed models (k-means) as a middle ground

5.4 Hierarchical

5.4.1 divisive

Top-down methods, necessity of discriminants, computational complexity.

5.4.2 agglomerative

Bottom-up methods. Requires metric, or at least a partial ordering function. computational complexity.

5.5 Context Clustering

By choosing the appropriate similarity metrics and models of context, we can use well-known clustering and/or numerical taxonomy techniques to investigate the structure of the language.

6 Proposed Experiments & Schedule

I plan to create a variety of classifications differing in the size and style of context representation as well as the similarity metrics used, with the intent of evaluating and comparing the results of each.

Both the small and large models can be built in a two phase process. The first phase entails creating classifications for the contexts of each word separately. Because of the data sparseness issue, one cannot expect to create a robust context classification at this stage.

Using a number of context classes perhaps 2 to 5 times the number of words (decided perhaps by an entropy cut-off), one can label the corpus with category membership, and reclassify the contexts of the study words, including category labels within the similarity measures. By adding this label information to the corpus, especially for the low frequency words, much more of the context data will become comparable. This will *smooth* the sparse data, especially the adverse effects of singleton words.

6.1 Initial models

An initial field of classification will be created and evaluated (subjectively and with respect to part-of-speech labeling, see below). Those methods which have the best results will be used to build larger scale models to be evaluated with the remaining tests.

Some of these experiments will be single-phase context classifications, using $\approx 10,000$ contexts, about 50-100 for 100-200 words.

Some will incorporate the two step smoothing.

6.1.1 Context representations

- $(0,0),(1,1)$ and $(2,2)$-simply conditioned models of context
- widths (2,2) (3,3) (5,5) [also (7,7) time/resources permitting]

6.1.2 Similarity measures

Each to be tested with and without locality weighting:

- manhattan
- cosine
- entropy
- edit distance

6.1.3 Large scale classification

Initial classification of $\approx 2,000$ words, to derive category set.

Second classification of all terms found, based on first category set.

6.2 Evaluating various models

6.2.1 Comparison to Part-of-Speech tagging

6.2.2 Predictive accuracy: perplexity

6.2.3 Word sense identification

Muc-6 word sense ID.

6.3 Schedule

existing work	I already have code (LISP and C) to implement the basic categorization scheme, and minimal display and evaluation tools. Some initial studies (Waterman, 1994) have already been done.
June–September	employed at Apple, Cupertino to develop semantic categorization for reference material. Further development of code, especially evaluation tools. Some of the small initial models will be built here.
October-December	completion of initial models and evaluation. Will need to port code to MasPar and Onyx for the large scale classification.
January–February	subsequent (re)analysis, finish writing, defend.

7 Innovative Claims

1. A de novo technique for separating *modes of use* of words. These modes are identified inductively on the basis of syntactic and semantic context cues.

2. A de novo technique for acquiring word classes derived from empirical evidence of use. Word classes are based on a classification of possible contexts in which words are used.

3. An explicit general representation of the context classes, in terms of probabilistic models of word appearance and configuration, in an accessible form. Most lexical classification schemes do not induce such representations, which offer explanatory evidence for the method. Those that do (n-gram techniques) aren't linguistically supportive descriptions.

   These descriptions can be used to support lexically-indexed grammar mechanisms such as LTAGS, slot grammars, link grammars, and word-expert parsing.

4. Individual words have partial membership in each of the context classes in proportion to the degree to which they appear in that type of context. This is in opposition to schemes in words have crisp membership in word classes based on similarity to other words (Grishman & Sterling, Brown, Hughs, Finch, etc.).

5. A word's membership in various context classes define it's relation to other words, from which word classes can be derived. (much like Grishman&Sterling, although along completely different parameters).

6. This representation, modes of use, context classes, and word classes, provide a means of expressing the constraints between lexical appearance constraints and semantic constraints. As such, it will be a good empirical platform for testing predictions of lexical semantic theories such as GL.

7. The derived classification gives empirical evidence for deriving such constraints.

8 Dissertation Outline

References

(1988). Proc. of the 12^{th} International Conference on Computational Linguistics, Budapest.

(1988). The wall street journal.

(1991). *Proceedings of the 29^{th} Annual Meeting of the ACL*. Association for Computational Lingistics.

(1992). *Proceedings of the 14^{th} International Conference on Computational Linguistics*, Nantes, France.

(1992). *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, San Mateo. Morgan Kaufman.

Anderberg, M. R. (1973). *Cluster Analysis for Applications*. Academic Press, New York. originally Anerbergs thesis.

Ayuso, D., Boisen, S., Fox, H., Ingria, R., & Weischedel, R. (1992). Bbn: Description of the plum system as used for muc-4. In *Proceedings of the Fourth Message Understanding Conference (MUC-4)*.

Baker, J. K. (1975). Stochastic modeling for automatic speech understanding. In Reddy, D. R. (Ed.), *Speech Recognition*. Academic Press.

Basili, R., Pazienza, M. T., & Velardi, P. (1993). Hierarchical clustering of verbs. In *Acquisition of Lexical Knowledge from Text, Proceedings of the SIGLEX Workshop*.

Black, E., Jelinek, F., Lafferty, J., Magerman, D., Mercer, R., & Roukos, S. (1992). Towards history-bases grammars: Using richer models of probabilistic parsing. In *Proceedings of the Fifth DARPA Workshop on Speech & Natural Language*.

Boguraev, B. & Briscoe, T. (Eds.) (1989). *Computational Lexicography for Natural Language Processing*. Longman, London. gidlines to using Longman's LDOCE for lexical acquisition.

Brent, M. R. (1991). Automatic acquisition of subcategorization frames from untagged text. In *Proceedings of the 29^{th} Annual Meeting of the ACL*.

Brown, P., Della Pietra, V. J., DeSouza, P. V., Lai, J. C., & Mercer, R. L. (1992). Class-bassed n-gram models of natural language. *Computational Linguistics*, 18(4).

Chan, S. & Wang, A. (1991). Synthesis and recognition of sequences. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(12):1245–1255.

Cowie, J., Guthrie, L., Pustejovsky, J., Wakao, T., Wang, J., & Waterman, S. (1993). The diderot information extraction system. In *Proceedings of the First Conference of the Pacific Association for Computational Linguistics*, Vancouver.

Dagan, I., Marcus, S., & Markovitch, S. (1993). Contextual word similarity and estimation from sparse data. In *Proceedings of the 31^{st} Annual Meeting of the ACL*.

Dagan, I., Periera, F., & Lee, L. (1994). Similarity-based estimation of word coocurrence probablities. In *Proceedings of the 32^{nd} Annual Meeting of the ACL*.

DeMarcken, C. (1990). Parsing the lob corpus. In *Proceedings of the 28^{th} Annual Meeting of the ACL*.

Finch, S. & Chater, N. (1992). Bootstrapping syntactic caegories using statsitical methods. In Daelemans, W. & Powers, D. (Eds.), *Background and Experiments in Machine Learning of Natural Langague*.

Fu, K.-S. & Booth, T. L. (1975). Grammatical inference: Introduction and survey. *IEEE Transactions on Sytems, Man, and Cybernetics*, 5(1,4).

Fukunaga, K. (1990). *Introduction to Statistical Pattern Recognition*. Academic Press, San Diego, second edition.

Futrelle, R. & Gauch, S. (1993). Experiments in syntactic and semantic classifications and disambiguations using bootstrapping. In *Acquisition of Lexical Knowledge from Text, Proceedings of the SIGLEX Workshop*.

Gale, W. A. & Church, K. (1990). Poor estimates of context are worse than none. In *Proceedings of the 28^{th} Annual Meeting of the ACL*.

Goodman, M., Waterman, S., & Alterman, R. (1991). Interactive reasoning about spatial components. In *Proceedings 13th Conference of the Cognitive Science Society*, Hillsdale, NJ. Earlbaum.

Grishman, R., Macleod, C., & Sterling, J. (1992). New york university: Description of the proteus system as used for muc-4. In *Proceedings of the Fourth Message Understanding Conference (MUC-4)*.

Grishman, R. & Sterling, J. (1992). Acquisition of selectional patterns. In *Proceedings of the 14th International Conference on Computational Linguistics.*

Hindle, D. & Rooth, M. (1991). Structual ambigity and lexical relations. In *Proceedings of the 29th Annual Meeting of the ACL.*

Hobbs, J. & Appelt, D. (1992). Sri international: Description of the fastus system used for muc-4. In *Proceedings of the Fourth Message Understanding Conference (MUC-4)*.

Hobbs, J., Appelt, D., & Pal, M. (1990). Interpretation as abduction. Technical Report Technical Note 449, SRI International AI Center, Palo Alto.

Hogeweg, P. & Hesper, B. (1984). The alignment of sets of sequences and the construction of phyletic trees: An integrated method. *J. Molecular Evolution*, 20:175–186.

Hughs, J. (1994). *Automatically acquiring a classification of words.* PhD thesis, University of Leeds.

Jelinek, F., Lafferty, J. D., & Mercer, R. L. (1992). Basic methods of probabilistic context free grammars. Technical report, Continuous speech recognition group, IBM T.J. Wason Reserch Center.

Johansson, S. & Stenström, A.-B. (Eds.) (1991). *English Computer Corpora: Selected Papers and Research Guide.* Mouton de Gruyter, Berlin.

Lehnert, W., Cardie, C., Fisher, D., McCarthy, J., Riloff, E., & Soderland, S. (1992). University of massachusetts: Muc-4 test results and analysis. In *Proceedings of the Fourth Message Understanding Conference (MUC-4)*.

Lehnert, W., Cardie, C., Fisher, D., Riloff, E., & Williams, R. (1991). In *Third Message Understanding Conference (MUC-3)*, pp. 223–233.

Levenshtein, V. Binary codes capable of correcting deletions, insertions, and reversals. *Cybernetics and control Theory*, 10(8):707–710. originally published as (Levenshtein, 1965).

Levenshtein, V. (1965). Binary codes capable of correcting deletions, insertions, and reversals. *Doklady Akademii Nauk SSR*, 163(4):845–848.

Lu, S. & Fu, K. S. (1977). A clustering procedure for syntactic patterns. *IEEE Trans. on Systems, Man, and Cybernetics.*

Meteer, M., Schartz, R., & Weischedel, R. (1991). Empirical studies in part of speech labelling. In *Proc. of the 4th DARPA Workshop on Speech and Natural Language*, pp. 331–336.

Periera, F., Tishby, N., & Lee, L. (1993). Distribution clustering of english words. In *Proceedings of the 31st Annual Meeting of the ACL.*

Pustejovsky, J. (1991). The generative lexicon. *Computational Linguistics*, 17(4).

Pustejovsky, J. (1992). The acquisition of lexical semantic knowledge from large corpora. In *Proceedings of the Fifth DARPA Workshop on Speech & Natural Language.*

Pustejovsky, J. (forthcoming). *The generative Lexicon: A Theory of Computational Lexical Semantics.* MIT Press.

Pustejovsky, J. & Anick, P. (1988). The semantic interpretation of nominals. In Proc. of the 12^{th} International Conference on Computational Linguistics.

Pustejovsky, J. & Boguraev, B. (1993). Lexical knowledge representation and natural language processing. *Artificial Intelligence.*

Sankoff, D. & Kruskal, J. (Eds.) (1983). *Time warps, string edits, and macromolecules.* Addison Wesley, Reading, MA.

Schütze, H. (1993). Part of speech induction from scratch. In *Proceedings of the 31^{st} Annual Meeting of the ACL.*

Sellers, P. (1974). An algorithm for the distance between two finite sequences. *J. Comb. Thy,* A16:253–258.

Skakakibara, Y., MichaelBrown, Underwood, R. C., Mian, I. S., & Haussler, D. (1993). Stochastic context-free grammars for modeling rna. Technical report, U.C. Santa Cruz.

Smadja, F. (1989). Macrocoding the lexicon with co-occurrence knowledge. In *First Int'l Language Acquisition Workshop,* IJCAI 89.

Wagner, R. & Fisher, M. (1974). *J. ACM,* 21:168–173.

Waterman, S. A. (1993). Structural methods for lexical/semantic patterns. In *Acquisition of Lexical Knowledge from Text, Proceedings of the SIGLEX Workshop.*

Waterman, S. A. (1994). Distinguished usage. In Boguraev, B. & Pustejovsky, J. (Eds.), *Corpus Processing for Lexical Acquisition.* MIT Press.

Wilks, Y., Fass, D., Gou, C., McDonald, J., Plate, T., & Slator, B. M. (1990). Providing machine tractable dictionary tools. *Machine Translation,* 5.

Zernik, U. (1990). Lexical acquisition: Where is the semantics? *Machine Translation,* 5:155–174.