

Chapter 5

Quantifying similarity

In the previous chapter, I outlined a method of *model-based, local distributional smoothing* that I argued better matched the problem of word class formation and occurrence estimation than standard smoothing methods, due to the severity of the sparse data effects. As mentioned there, this sort of smoothing operation requires a measure of *locality* within the sample space, which in this case is a space of discrete context sequences. Such a similarity measure defines an implicit structure to the space of possible data, allowing one to represent the frequency of various phenomena as distributions throughout the structured space, rather than as point-valued phenomena defined only on the observed data.

The first part of this chapter revolves around the issues involved in defining a measure which can be used to structure the data space. Section §5.1 discusses various sequence measures that can be used to structure the space of possible contexts, and defines a local region within the data space that is required for the local smoothing. Section §5.3 gives some quantitative values of these measures over real language data, comparing the various choices of string measure.

Similarity results from real data are also compared to those from a data set generated from the first-order word frequency distribution. This comparison confirms the effectiveness of the similarity measures at identifying the high-order

linguistic structure present in the real data, but absent in the approximated data set.

The following chapters discuss using these sequence similarity measures as the basis for the construction of classes and the class-based associative model.

5.1 Sequence measures

Central to the discussion of the last chapter was the notion of a similarity metric which compares two contexts. This metric captures the relevant linguistic structure of sequences of symbols, and tells us when two sequences have like structure. We then use this metric to categorize all like sequences into a class, in effect averaging their properties into the class description. By categorizing all possible sequences into some manageable set of context classes, we reduce the complexity of the representation and create a model based on these context classes.

A linguistically meaningful reduction of the space of possible contexts would have to accomplish this averaging while preserving the scope of linguistic inference we wish to derive from the model. Such a reduction would preserve our intuitions about paradigmatic substitution classes, namely, that they represent lexical items that are interchangeable under limited conditions. In order to collapse similar contexts into a local region, we must make some allowance for disagreements in context comparison. Replacing an element of the context with a close synonym, for example, should not greatly affect the comparison. Neither should replacing that one element with a synonymous multi-word phrase.

One could imagine many possible means of comparing two sequences of tokens. We are searching for a comparison, however, that preserves the specific sort of meaningful *linguistic* similarity discussed in ch. 2. Such a measure would incorporate information relating both the identity of the tokens, and their configuration.

A measure of paradigmatic similarity would provide a starting point for such

comparisons between contexts, providing the basic judgement used to measure the total disagreement between contexts. However, we start with no model of language,¹ and the data we have contain no measure of synonymy. Such a measure is, in fact, part of what we are hoping to derive from this study. The data as it is given is represented only as a sequence of tokens. Because we are not starting with a model of language, these tokens have *nominal values* only. That is, they can only be compared in the sense that any two tokens either have identical values, or not, giving us a match/not-match binary comparison as a starting point for measuring context similarity. By counting the numbers of matches/disagreements between the pairs of tokens in two contexts, we can begin to measure the total similarity of the contexts.

In order to incorporate a linguistic sense of similarity in such a comparison, we will need to organize the way in which tokens are compared between the two contexts under consideration. There are many ways of choosing the possible pairs to be compared.

5.1.1 Set measures

One trivial way of making this comparison is to ignore the sequential aspects of the contexts, and treat each as a simple set of nominal values. The most basic comparison that can be made of two sets with nominal features is a count of the features shared between the two. Symbolically, if we treat the two sequences A and B as *sets of features* drawn from some feature space $\{f_1, f_2, \dots\}$, this count is the cardinality of their intersection, $S(A, B) \stackrel{\text{def}}{=} \|A \cap B\|$. In using this measure for comparing two lexical contexts, we can treat lexical tokens as the features from which sets are drawn, and treat the contexts as simple collections of tokens. The measure S is then the number of unique tokens in common between the two

¹It might be thought that a simple model of morphology or part of speech could be used to some advantage here. That has its own difficulties, in judging the appropriateness of such a model for the task at hand, and further, of judging the numerical relationships of similarity between morphological variants or parts of speech.

contexts.

This measure has number of drawbacks in terms of its linguistic appropriateness. First, and perhaps most importantly, the measure S does not take into account any ordering or configurational information that may be present in the lexical environment. The contexts are treated simply as bags of words, and any lexical constraints represented by their ordered relation to one another is completely ignored. Secondly, S is a very discrete measure, taking on only $n + m + 1$ possible values in the comparison of two (n, m) -contexts (the intersection can have zero to $n + m$ elements). Given the large domain of possible contexts (see §??), this range of possible comparison values seems incredibly sparse. Lastly, the measure S also doesn't consider the effects of the multiple occurrences of elements with the two sets. In the extreme, if A and B were identical, but each composed only of a repeated occurrence of a single token, say, x , then the measure would be $S(A, B) = 1$ no matter what their length, the same value that would be assigned two contexts which had no repeating values, but had only one element in common. At another extreme, two identical sequences of length n with no repeated tokens $A = B = [x_1, x_2, \dots, x_n]$ would have a measure $S(A, B) = n$.

The Jaccard measure

Another basic measure for comparing items based on their nominal features, which avoids these drawbacks in the set measure S resulting from repetition is the Jaccard measure. This is the ratio between the number of elements shared by two objects and the total number of elements in their union [Anderson?, p. 89].² The measure is simply the ratio of overlapping elements to total elements:

$$J(A, B) \stackrel{\text{def}}{=} \frac{\|A \cap B\|}{\|A \cup B\|} \quad (5.1)$$

In using this measure for comparing two lexical contexts, we again treat tokens as the features from which sets are drawn, and the contexts, which are sequences

²or maybe it's anderberg. need to check. Plus look in everitt.

of tokens, as simple collections of features. Thus, the Jaccard measure between two contexts such as c_1 and c_2 , below

$$c_1 = [a\ b\ p\ q] \quad c_2 = [c\ b\ r\ q] \quad (5.2)$$

is computed as:

$$J(c_1, c_2) = \frac{\|\{a, b, p, q\} \cap \{b, c, q, r\}\|}{\|\{a, b, p, q\} \cup \{b, c, q, r\}\|} = \frac{\|\{b, q\}\|}{\|\{a, b, c, p, q, r\}\|} = \frac{2}{6} \quad (5.3)$$

By using the cardinality of the union as the denominator in the ratio, the Jaccard measure normalizes the size of the intersection set according to the intrinsic ‘size’ of the two sets combined, and avoids the variable results provided by S . This normalization also has the useful property of restricting the value of the measure to the range $[0, 1]$

While the Jaccard measure deals well with a variable number of distinct tokens within each context, it still suffers from the other problems which make the set measure unsuitable for linguistic analysis: it does not take configuration into account, and has a very limited range of possible values when used to compare short contexts.

The Jaccard measure and variants have been used to good effect in comparing co-occurrence distributions and in problems in information retrieval [Grefenstette, Sparck-Jones].³ In these applications, though, the sets being used are summaries of usage across many occurrences, or representations of entire documents, containing a much wider distribution of tokens than relatively short contexts. Because of the size of the token set in the document representation, the problem of few output values does not exist for these applications. Because the Jaccard measure has been used effectively in these applications, the experiments in §5.3 show Jaccard results as a baseline.

Distance v. similarity The Jaccard measure as given is a measure of *similarity*, with higher values indicating contexts that are more alike. In order to make the

³find cites fro greffenstette's new book, and KSJ's ir work.

values comparable to those of the other measures presented, the experiments use the *distance* measure $J' \stackrel{\text{def}}{=} 1 - J$, for which high value indicate dissimilar contexts, and a value of 0 is given to identical objects.

5.1.2 Aligned pairwise comparisons

One method of incorporating the order implicit in each sequence of token elements is through an *aligned pairwise* comparison, which only considers the similarity of elements occupying the same position in each sequence, much like the Hamming distance between two distributions. By making this limitation on the pairs which are evaluated, the comparison takes into account the critical aspect of the configuration of symbols within each context.

We define a pairwise aligned comparison measure for two contexts, $A = [a_1, \dots, a_n]$, $B = [b_1, \dots, b_n]$, as the sum of pairwise disagreements between tokens from identical positions. As with the Jaccard measure, we normalize by the length of the context sequence.

$$P(A, B) = \frac{1}{n} \sum_{i=1}^n d_i, \quad \text{where } d_i = \begin{cases} 0 & \text{if } a_i = b_i \\ 1 & \text{if } a_i \neq b_i \end{cases} \quad (5.4)$$

The normalizing factor $1/n$ serves to restrict the values of P to the range $[0, 1]$. This measure could easily have been defined in a complimentary fashion, so that P' would have $d_1 = 1$ for equal tokens and 0 otherwise. The given method was again chosen so that it was a measure of distance between contexts, with a value of 0 for identical contexts, rather than a similarity measure.

For the two contexts given above (eqn. 5.2), the pairwise comparison measure P is computed

i	1	2	3	4
c_1	a	b	p	q
c_2	c	b	r	q
d_i	1	0	1	0

$$P(c_1, c_2) = \frac{1}{4}(1 + 0 + 1 + 0) = 1/2 \quad (5.5)$$

The measure P provides a degree of improvement over set measures such as Jaccard. By limiting the comparisons only to aligned items, P manages to capture at least the linear ordered structure of the contexts under comparison, which is completely ignored in the measure J . However, it does this in the crudest possible way, which ignores many of the obvious properties of real linguistic data. For example, we might like to think that the simple noun phrases ‘the boy’ and ‘the tall boy’ are very similar, and should not be regarded very differently in an analysis of a larger expression in which they might be embedded. One has a modifier, the other does not, but our intuition is that they will behave quite similarly in use. However, because the pairwise metric P forces a strict alignment of tokens, it will indicate that these two phrases are extremely different.

Embedding the phrases as the subject of the VP ‘went to the store’, we can see explicitly why this is the case. Below (5.6) are listed the the $(0, 4)$ -contexts of the token t_0 = ‘the’ for three phrases, which we would think should be judged as fairly similar.

(0, 4)-context

c_1	the	boy	went	to	the	store	(5.6)
c_2	the	tall	boy	went	to	the store	
c_3	the	girl	went	to	the	store	

One notices immediately that the additional token ‘tall’ brings the second sequence out of alignment. The pairwise comparisons will in this case fail to capture the similarity of the the remainder of the sequence, failing to account for this shift in alignment. The measure between the first two is large, $P(c_1, c_2) = 1$, while $P(c_1, c_3) = 1/4$. The alignment forced by the simple pairwise comparisons is not in agreement with the linguistic intuition that the two variants, c_2 and c_3 , do not differ so wildly. Certainly not so much that the pair (c_1, c_2) should be considered maximally distance, while the pair (c_1, c_3) is measured as relatively similar.

Related use of pairwise comparisons

Despite its drawbacks, this measure has been used to reasonable effect in some studies of the syntactic relatedness of words. This aligned pairwise comparison is very close to the measures underlying work such as that represented by [Schütze, 1993a, Finch and Chater, 1992, Hughs, 1994, Futrelle and Gauch, 1993]. Each of these approaches aligns a set of contexts appearing with a single word, w_k , and tallies the occurrences of tokens within each aligned position. In the diagram 5.7, the $f(t_i)$ represent these tallied frequency distributions for each aligned position. These distributions represent the range of usage in the position t_i for the collection of appearances of w_k .

$$\begin{array}{cccc|c|cccc}
 \dots & t_{-3}^1 & t_{-2}^1 & t_{-1}^1 & w_k & t_1^1 & t_2^1 & t_3^1 & \dots \\
 \dots & t_{-3}^2 & t_{-2}^2 & t_{-1}^2 & w_k & t_1^2 & t_2^2 & t_3^2 & \dots \\
 \dots & t_{-3}^3 & t_{-2}^3 & t_{-1}^3 & w_k & t_1^3 & t_2^3 & t_3^3 & \dots \\
 \hline
 \dots & f(t_{-3}) & f(t_{-2}) & f(t_{-1}) & w_k & f(t_1) & f(t_2) & f(t_3) & \dots
 \end{array} \tag{5.7}$$

A set of $f(\cdot)$'s is composed for each word w_k in the study, and a comparison of these distributions is used to evaluate the syntactic similarity of the words. Since this representation of the use of a word is very large, requiring a large distribution for each of several context positions, a severe amount of data reduction is often performed [Schütze, 1993a] before the comparison between words is actually performed. However, the comparisons between words amount to a measure of the similarity of distributions in each position, which is essentially a weighted count of the occurrences of tokens in that position for the words.

This type of analysis first groups all contexts occurring with a given key word w_k , and seeks to characterize the similarities among the averaged formal properties of the contexts. The central word is treated as an *independent* variable, and the syntactic environment is studied as an aggregate of all environments that occur with that word. While this does create a robust estimate of the total usage of the word, attacking the problem this way, loses the rich, polymorphic behavior of

individual words, conflating the constraints and dependencies that may operate different on separate modes of use, resulting in words treated as atomic entities with monolithic behavior. Because the lexical environment, containing many words, is more complex than the single central element, studying the relationship in this way will necessarily lose information.

In the present work, I am attempting to discover and model the structure of the syntactic environment, which is more variable than the choice of central word, and use that to infer properties and features of the central element, treating the word as *dependent* on the structure of the environment. By comparing contexts individually, rather than as an aggregate, we preserve information about the varied uses of any single word. Of course, the data reduction in approaches such as these ([Schütze, 1993a] etc.) makes the problem tractible, by aggregating all occurrences of a word based on the implied relationship of licensing that word. In our approach, we will be using the context similarity measures to aggregate contexts, based on the observable relationship between them. This lets us study the issues of license and dependency without confusing them with issues of polysemy and variable syntactic structure.

5.1.3 Edit Distance

To overcome the difficulties of fixed alignments in pairwise comparison of sequences we need to allow for a flexible alignment of tokens between the two contexts. This is often accomplished using the *edit distance* [Levenshtein,], a measure based on relating pairs of tokens between the two sequences in a structured way, but not necessarily in strict sequential alignment. Insertions and deletions of intervening, uncomparing tokens are also allowed. Straightforward computational methods for finding the edit distance between two strings have been used on a variety of problems in biology, genetics, speech and handwriting analysis ([Sankoff and Kruskal, 1983]), as well as in syntactic analysis of formal languages

([Lu and Fu, 1977]). (For a good introduction with applications to many domains, see [Sankoff and Kruskal, 1983].)

Number of Operations.

The edit distance can be defined as the minimum number of token insertions, deletions, and substitutions required to transform one sequence to another [Wagner and Fisher, 1974]. Letting A, B be arbitrary sequences and x, y be tokens, the insertion, deletion, and transformation operators are defined as

$$\begin{aligned} \text{insertion:} & \quad AB \xrightarrow{o_i} AxB \\ \text{deletion:} & \quad AxB \xrightarrow{o_d} AB \\ \text{substitution:} & \quad AxB \xrightarrow{o_s} AyB \end{aligned} \tag{5.8}$$

The *edit distance* is the length n of the shortest sequence of operators $A \xrightarrow{o_1} A_1 \xrightarrow{o_2} \dots \xrightarrow{o_n} A_n = B$ which transforms token sequence A into B .

For example, consider the following somewhat similar sequences:

$$\begin{aligned} A &= [\text{the boy went to the store}] \\ B &= [\text{the tall boy drove to the store}] \end{aligned} \tag{5.9}$$

In this case, A can be transformed to B through one insertion

$$[\text{the boy}] \xrightarrow{o_i} [\text{the tall boy}]$$

and one substitution

$$[\text{went}] \xrightarrow{o_s} [\text{drove}]$$

This is a total of 2 operations.

Many different sequences lead to the same transformation. For instance, multiple tokens could have been added to any position in the sequence, and then later removed. One could construct a transformation of an arbitrarily large number of operations in this way. The edit distance is defined as the *minimum* number of operations necessary. The minimum transformation for this example is readily apparent, but in general, finding the minimum sequence of operators is not always this obvious.

Generalizations for the edit distance.

Another transformation sequence, in which the substitution operations “[went] $\xrightarrow{o_s}$ [drove]” is replaced by the two operations of deleting ‘went’ and inserting ‘drove’ seems just as basic, but would involve one extra operation. The substitution operator can be seen as a generalization over the complex operation of ‘delete then insert’.

One generalization of the basic edit distance which solves this discrepancy, as well as increases the power of the measure, is formed by assigning a cost to each operation. One could assign a cost of, say, 1, to each of the insert and delete operators, and a cost of 2 to the substitution operator. Then the two sequences in (??) have an identical cost of 2.

$$\begin{array}{l} [\text{went}] \xrightarrow{o_d} \xrightarrow{o_i} [\text{drove}] \\ [\text{went}] \xrightarrow{o_s} [\text{drove}] \end{array} \quad (5.10)$$

This can be generalized still further, by assigning costs based not only on the operation, but also on the particular symbol involved. This is especially useful in a linguistic context. For instance, we may not wish to assign a great cost to the insertion of a modifier, such as an adjective, because we know that the structural properties of the NP it modifies will not be unduly affected. For the most part, most contexts that allow a bare N will also allow one or more modifiers. However, we may wish to assign a greater cost to the insertion of a verb, because we know that most constructions do not allow extra verbs to be inserted. Auxilliaris and modals would be assigned a lower insertion cost, because we know they may or may not appear in a VP construction.

Substitutions may be likewise dependent on the words involved. Replacing a word with a similar term should not incur a large penalty. However, replacing a word with an unlike word may destroy the linguistic properties of the sequence,

and should carry a greater cost. In our example, replacing the general term ‘went’ with the more specific ‘drove’ preserves much of the meaning of the sentence. However, replacing ‘went’ with something completely dissimilar, say ‘mantlepiece’, would completely destroy the linguistic structure of the sentence, as well as its meaning, and should carry a high cost.

Initially in our study, we have no information about word similarity. But, as the modelling stages progress, we will be able to find word similarity values based on distributional similarity across context clusters, formed using the context similarity measures. We can then, of course, feed these derived word similarities back into the process by using them to assign values to these inter-word substitution costs. In this way we can be sure that the sequence similarities computed have a better correspondence with the linguistic properties of their component words.

The *generalized edit distance* can now be defined in terms of the *minimum total cost* of a transformation sequence, rather than simply the minimum number of operations. If one makes the simple assumption that a substitution costs no more than the equivalent deletion-insertion pair, then this minimum cost distance can be shown to obey metric properties (including the triangle inequality). We formalize this measure in the following section.

Computing edit distance.

There is a straightforward method for computing edit distance ([Sellers, 1974, Wagner and Fisher, 1974]). As a prime example of dynamic programming, one can compute the edit distance for every pair of initial subsequences of the two strings under study, combining results for shorter substrings to give results for longer subsequences.

Explicitly, let our two sequences be $A = (a_1, \dots, a_m)$ and $B = (b_1, \dots, b_n)$, where a_i is the i^{th} token in string A , starting with token 1. We add a *null token* to the head of the each string, tokens a_0 and b_0 , representing an empty position that serves

$A \longrightarrow$

	-	the	boy	went	to	the	store	
$B \downarrow$	-	0	1	2	3	4	5	6
	the	1	0	1	2	3	4	5
	tall	2	1	2	3	4	5	6
	boy	3	2	1	2	3	4	5
	drove	4	3	2	3	4	5	6
	to	5	4	3	4	3	4	5
	the	6	5	4	5	4	3	4
	store	7	6	5	6	5	4	3

Figure 5.1: Dynamic programming for edit distance ('-' is the null token)

as a placeholder. Let us also define the initial substring of a string to be the first i tokens, $A_i \stackrel{\text{def}}{=} (a_0, a_1, \dots, a_i)$, including the null token inserted at the beginning.

Let $D(A, B)$ by the edit distance between the sequences A and B . The computation starts by defining $D(A_0, B_0) = 0$, 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:

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

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 . We assume that the substitution is symmetrical, $D_{\text{substitute}}(x, y) = D_{\text{substitute}}(y, x)$

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 5.1 gives this table for the example strings from (5.9).

5.1.4 Linguistic plausibility of comparisons

One can see that all of these measures fulfill a number of our criteria as a measure suitable for measuring some linguistic factors relating two sequences. The Jaccard and pairwise measures, though, fail in a number of respects to fully capture the kinds of configurational similarity we seek to measure. Jaccard offers the flexibility of using implicit comparisons between any pair of tokens from the two sequences, but does this at the expense of ignoring all configurational information. The pairwise measure allows only those comparisons which are aligned in the strictest sense, thus preserving the most obvious configurational information, but, as we saw with example 5.6, it ignores a large class of interesting relations involving mis-alignments.

The edit distance, on the other hand, can allow both structural and token identity disagreements while not being brittle in either. Like the pairwise measure, it will give higher similarity to structurally similar sequences. However, it will degrade gradually, but not break in comparisons which are misaligned. And while it does not go the full way toward incorporating comparisons of *any* pair of tokens as Jaccard does, it uses the comparison between all pairs to decide the best minimal set of transformations. The edit distance certainly does not encode all the linguistic similarity present between two sequences, but it is closer than the others presented here to comparing sequence features related to the syntactic configuration of words.

5.2 Richer measures

These measures do capture much of what we mean by linguistic structural similarity, even though they do not incorporate anything that we might regard as a model of syntactic structure. They capture configurational information such as

word order, phrasal similarity under modifier insertion, and the similarity of substitution by synonymous terms. We can improve upon them, however, by adding other important linguistic qualities. Most importantly, the idea of locality of function, that a word's contribution will be expressed most stringly in the configuration immediately surrounding it, is lacking in the current measures. The following section elaborates some methods for adding this information.

5.2.1 Locality in the metrics

There is a strong degree to which formal configurational (syntactic) considerations are local in nature. As we discussed for the choice of local context as a descriptor for the formal constraints operating on a token in the language, syntactic well-formedness and syntactic interactions are behaviors that operate primarily in a local region. Discussions of long-distance dependencies as deviation from the norm only serve to highlight the degree to which local interactions are assumed as the norm.

In order to highlight the locality of these configurational dependencies, we can modify the importance associated with each of the token comparisons contributing to the distance measure. One would like to weight near tokens more heavily, but without ignoring the contributions of distant ones. The window of our short context sequences is already a simple version of such a choice, as it includes near tokens up to some distance, and completely ignores further ones.

A linear weight decreasing with distance from t_0 would also include contributions from all tokens up to a point. We could, for instance, simply count the nearest token in full, and decrease the weight given to token t_i linearly with distance. If the maximum weight is $w = 1$, the weight for each token t_i under this approach is given by $w_i = 1 - iK$, for some constant $0 < K < 1$. One has to be sure that this decrease is not too rapid, however, since the point at which the weight becomes zero is the effective width of the context description. It would be

counterproductive to include more distant tokens, as they would have negative weights.

Alternatively, one could decrease the weight geometrically with each token, such that the weight given to a token is a fixed fraction of the previous weight. In this geometric approach, the weight is given as $w_i = K^i$, again for $0 < K < 1$. This approach has the advantage that the weights are positive for all i .

Half-weight length. For both the linear and the geometric approach, we can specify the rate of drop off in terms of a *half weight length*, P_h , which is the length (token position) at which the contribution of a token drops to $1/2$. This correspondence between length and weight drop off allows us to connect the weighting constants with our observations about the range of interaction included in the measure.

For the linear scheme, the dependence of the weighting factor, K , on this half weight length is given in (5.11).

$$\begin{aligned} w_i &= 1 - iK \\ 1/2 = w_{P_h} &= 1 - P_h K \\ K &= \frac{1}{2P_h} \end{aligned} \tag{5.11}$$

For the geometric scheme, the dependency between K and the half length is given by (??).

$$\begin{aligned} w_i &= K^i \\ 1/2 = w_{P_h} &= K^{P_h} \\ K &= (1/2)^{\frac{1}{P_h}} \end{aligned} \tag{5.12}$$

The figure (5.2) illustrates the properties of the local weighting for both the linear and geometric approaches, with $P_h = 6$. The geometric scheme has weights

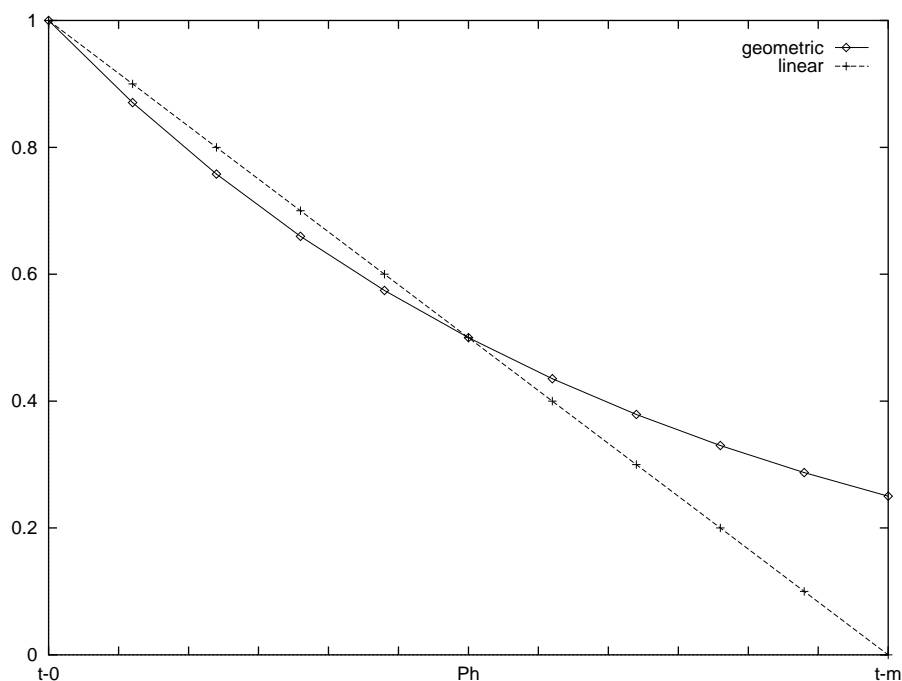


Figure 5.2: Linear and geometric comparison weighting functions, with $P_h = 6$.

approximately 5-10% lower for the first few tokens, but then flattens out while the linear weighting approaches zero after P_h .

By using decreasing weights such as these, we are shifting the importance of the correspondence in the sequences being compared, such that items near the key word in the context account for a greater degree of the comparison. Both the linear and the geometric model accomplish this shift in weighting. The linear model, though, is restrictive in that it allows the influence of farther items to be discounted entirely. The geometric decrease allows *all* later words to participate in the comparison. Also, the decrease in weights is not nearly as dramatic toward the distant end of the comparison. This is especially important in cases where, for instance, the difference between two sequences lies in that one has had a modifier inserted, but are otherwise alike (e.g. the sequences of (5.6)). If the later tokens are seriously discounted because of this, the inherent similarity of the sequences will be undervalued. For this reason, the experiments presented later all use the

geometric version of the locality weights.

Locality for the set measure

This kind of local preference weighting can be incorporated into the set measures described above. Recall the simple set measure $S = \|A \cap B\|$. We can weight each element of the intersection by the sum of the weights on each occurrence of that element as a token of A and B . Thus if $x \in A \cap B$, and the occurrences of x are $a_2 = x$ and $a_3 = x$, then the weight given to x in the intersection would be the sum $w_2 + w_3$.

This also has the property of reducing some of the odd behavior of the set measure with respect to repetition. With the weighting added, the measure now counts additional weight for additional matching tokens of the same type.

Formally, we can define the weighted set measure S' as:

$$\begin{aligned}
 S' &\stackrel{\text{def}}{=} \sum_{t \in A \cap B} s'(t) \\
 s'(t) &\stackrel{\text{def}}{=} s'_a(t) + s'_b(t) \\
 s'_a(t) &= \sum_i \begin{cases} w_i & \text{if } a_i = t \\ 0 & \text{otherwise} \end{cases} \tag{5.13}
 \end{aligned}$$

Alternately, we could define a weighted set measure to include only the weight for one of each token type in the intersection.

$$\begin{aligned}
 S'' &\stackrel{\text{def}}{=} \sum_{t \in A \cap B} s''(t) \\
 s''(t) &= \max(w_i \mid (a_i = t \text{ or } b_i = t)) \tag{5.14}
 \end{aligned}$$

This second formulation reduces to the cardinality of the intersection ($\|A \cap B\|$) when $w_i = 1$ for all i .

Locality for Jaccard

This weighted set measure can be extended to create a weighted Jaccard measure defined in eqn. 5.1. The weighted extension for the union $A \cup B$ is defined by weighting the occurrence of each token by the weight associated with its position in each respective sequence. This can be defined, using the definition of $s'_a(t)$ given above in eqn. 5.13 by the formula:

$$\begin{aligned} J'(A, B) &\stackrel{\text{def}}{=} \frac{I'(A, B)}{U'(A, B)} \\ I'(A, B) &\stackrel{\text{def}}{=} S' = \sum_{t \in A \cap B} s'(t) \\ U'(A, B) &\stackrel{\text{def}}{=} \sum_{t \in A \cup B} s'(t) \end{aligned} \quad (5.15)$$

Thus the locality weighted Jaccard measure is normalized by the weighted sum of all elements of both sequences.

Locality for pairwise comparisons

The pairwise measure has perhaps the easiest modification for locality weighting of the measures given here. In this case, each comparison pair is simply scaled by the weight associated with its position. The weighted pairwise comparison measure, P' , is then the sum of the weights of each matching pair:

$$P'(A, B) = \frac{1}{n} \sum_{i=1}^n d_i, \quad \text{where } d_i = \begin{cases} 0 & \text{if } a_i = b_i \\ w_i & \text{if } a_i \neq b_i \end{cases} \quad (5.16)$$

Locality in the edit distance

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

$$D(A_i, B_j) = \begin{cases} D(A_i, B_{j-1}) + w_{i+j} D_{\text{insert}}(b_j) \\ D(A_{i-1}, B_j) + w_{i+j} D_{\text{insert}}(a_i) \\ D(A_{i-1}, B_{j-1}) + w_{i+j} D_{\text{substitute}}(a_i, b_j) \end{cases} \quad (5.17)$$

where w_i is the weight for token distance i . Because the weights must be indexed off the sum of the indices, $i + j$, the weights decrease twice as fast as might be expected. A pair of contexts with a single insertion needed at token k would differ by $w_{2k}D_{\text{insert}}(a_k)$. However, a substitution required at the same location would be given a weight $2w_{2k}$ (if $D_{\text{sub}} = 2D_{\text{insert}}$, as is usual).

Token weights

All of these distance measures can also be generalized to compensate for different similarities between tokens. For instance, if one decides that ‘went’ and ‘drove’ are more similar to each other than either is, say, to ‘boy’ or ‘to’, then it would be reasonable to have the substitution of ‘went’ \rightarrow ‘drove’ have a lower cost than the other possible substitutions.

As discussed above, we initially have no evidence for assigning this kind of word similarity value, and we simply must assume that all unequal tokens are identically distinct. Later, however, as these sequence measures are applied in order to create context classes, we will accumulate evidence that can be used to form a model of distributional similarity among the tokens. In this way, we are able to use the hints offered by initial structural similarity to build more refined models of structural and lexical behavior.

5.2.2 String alignments

As a by-product of the edit distance computation, one can create an *alignment* of the two strings, which gives the elementwise pairing that is associated with the minimal sequence of edit operations. 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. Such an alignment can be generated directly from the table created in the edit distance computation by following the path of minima chosen during the computation from the upper left corner to the

lower right. Rightward travel along this path corresponds to insertion of a token from string *A*, downward travel to tokens from string *B*, and diagonal paths to substitutions. Multiple minimum paths can occasionally result, giving alternate but equivalent alignments. In these cases, I have chosen to favor the shortest path, which gives precedence to diagonal substitution paths.

The alignment created from our two example strings (figure 5.3) 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	-	boy	went	to	the	store
-	the	tall	boy	drove	to	the	store
-	the	-	girls	drove	to	the	store

Figure 5.3: A string alignment table

Alignments can be created for sets of multiple strings. These alignment tables can further be abstracted to probabilistic descriptions of the sequences, either by representing each column of the table as a word distribution, or by using a FSA or Markov chain description of the transitions from one column to the next. Chan and Wang ([Chan and Wang, 1991]) have used the column-distribution descriptions of alignment tables, which they call *syntheses*, in order to generalize the edit distance and capture the notion of distance between two sets of RNA base-pair sequences. Techniques such as this may prove useful in later linguistic work, as well.

5.3 String measure experiments

In order to evaluate the potential usefulness of these metrics for structuring the data space, several experiments were conducted using the various string measures. Each of the metrics was used to evaluate a large sample of real language data taken from a collection of articles from the *Wall Street journal* [wsj:corpus]. Section ?? briefly describes the data and its preparation. See appendix ?? for the

full details.

The principle evaluation consisted of computing the value each of the measures on all pairs taken from a set of approx. $N = 10^4$ context sequences chosen randomly from the WSJ corpus. This results in $N(N - 1)/2$ pairs for which the distance was computed using each measure.

Comparison with random data

In addition to the corpus of real English sequences, a baseline comparison corpus was created using a first-order distribution approximation. The comparison between this baseline and the actual English data reveals how the structural characteristics of actual language data differ from randomly constructed sequences.

What we expect to see is that the real language data will contain many more similar contexts than the randomly generated data. If this is the finding, then it will support the case for the structural properties of language sequences laid out in chapter 2. If linguistic function is represented systematically through configuration, and those configurations pattern after a relatively small number of classes, then the measures we have proposed should show a large number of similar contexts. By comparison, the randomly generated data has no such inter-word configurational dependencies, even though it contains the same word frequencies. Our sequence measures should show a relative lack of similar contexts in this random data.

By comparing the English data to the generated set, we also provide a baseline for the performance of the similarity measures. Assuming that the real language data does contain a greater number of similar contexts, we also need to establish, for each type of measure, at just what numerical value the similarity can no longer be attributed to real linguistic patterns. Beyond some degree, we expect that the random data will have as many pairs of 'similar' sequences, due purely to chance, as does the real language data. By comparing the results, we also establish this

point beyond which our measures no longer indicate linguistic significance.

5.3.1 English sequence data

A short description of the data and its treatment is given here. For a full description of the data, see appendix ??.

The sequences used in the experiments presented in this chapter were drawn from a corpus of *Wall Street Journal* articles, originally published in 1989, and distributed in both an unfiltered and a parsed, part-of-speech tagged form by the Linguistic Data Consortium [?wsj:corpus?].

Tokenization

The text has been tokenized to distinguish words from whitespace, punctuation, etc. This processing is somewhat different from that commonly seen in corpus processing literature in that whitespace, punctuation, and other symbols *have been retained as tokens*. It was considered that, as a study of the formal properties of written language, these characteristic demarcations, which are unique to written language, and which form a substantial part of its structure, should be explicitly retained and analyzed along with the words which have explicit counterparts in spoken language. Some properties of the written form, such as the upper/lower case distinction, were not retained in the tokenization process.

The following sequences of characters are tokenized as discrete symbols:

1. any continuous sequence of the alphabetic characters [a-zA-Z]
2. any continuous sequence of whitespace characters [newline space]
3. individual numerals [0-9]
4. individual punctuation characters and other symbols [, . " \$ % etc.]

Table 5.1 gives some examples of how the English data and the tokenized sequences correspond.

Defining sequences

Chapter ?? defined the local (n, m) -context of a word in corpus as the sequence:

$$[t_{-n} - t_{-1}] \quad t_0 \quad [t_{+1} - t_{+m}] \quad (5.18)$$

The sequences compared in the experiments presented here are $(7, 7)$ -contexts taken from the tokenized WSJ corpus. Each sequence is further broken down into its left and right segments. Each side of the comparison contains approx. 3 or 4 words and the separating whitespace.

Each sequence distance was computed separately on the left and right subsequences of the context. For sequence measures that apply a locality weighting, distance is taken extending out from the central key word. Thus the tokens at t_{-n} and t_{+m} receive the least weight. The final measure between two contexts A and B is taken as the sum of these separate left and right measures. The measure $M(A, B)$ between two contexts is computed as

$$M(A, B) = M(A_L, B_L) + M(A_R, B_R)$$

5.3.2 Generated sequence data

A comparison set was created, in order to study the difference between the sampled English data discussed above, and a generated sample with the same first-order distributional properties. The behavior of the various measures with respect to the real and first-order model data will show the extent to which the measures

English	Tokens
Sales of medium-sized cars, which Chrysler Corp.'s Chrysler division mirrors the "closed-end dropped nearly 7% Oct. 13,	sales _ of _ medium - sized _ cars , _ which chrysler _ corp . ' s _ chrysler _ division . mirrors _ the _ " closed - end dropped _ nearly _ 7 % _ oct . 1 3 ,

Table 5.1: English data v. tokenized form. Tokens are separated by '|', while '|_|' indicates an explicit whitespace token.

are able to detect the patterns of linguistic configuration in the English data.

The sample was created according to an ideal Zipf distribution, $p(w) = K1/r(w)$. The constant K was chosen to create a data set with a total size and word frequencies as close as possible to those of the WSJ data.

The log-log plot of token frequency versus token rank is shown in figure 5.4.

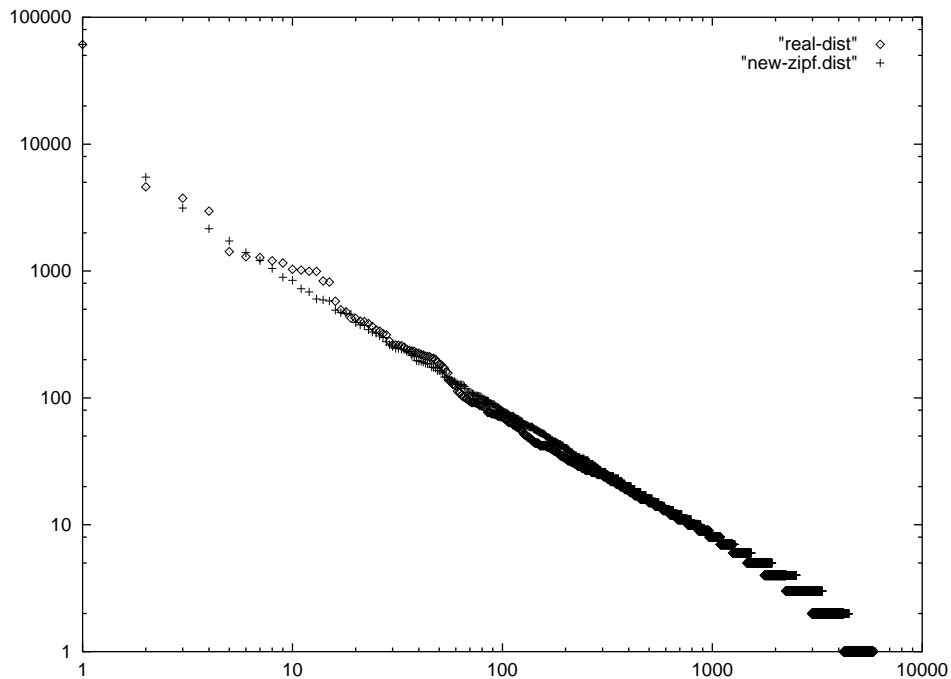


Figure 5.4: log-log plot of English data set and generated Zipf-distributed data.

5.3.3 Distance measure results

Histograms of the Jaccard, pairwise, and edit distance measures are shown in figures 5.5, 5.6, and 5.7. Each point on the plots expresses the number of sequence pairs having a given distance measured between them. The total number of pairs in each plot is ≈ 42 million (this is $N(N-1)/2$ for $N = 9238$). As mentioned above, the distance between a pair of sequences is the sum of the left and right side measures taken independently, $M(A, B) = M(A_L, B_L) + M(A_R, B_R)$.

Both the pairwise measure (5.6) and Jaccard measure (5.5) are normalized,

with values in the range [0-1], and so the combined left+right measures were binned in the range [0-2], with a bin size of 0.01. Edit distance (5.7) was binned in the range [0-30], also with a bin width of 0.01.

The figures on the ordinate in all the plots are given in the number of bins, and thus are either $10\times$ or $100\times$ the measured distance. The response variable (number of pairs) are plotted on a log scale, so that the enormous number of large distance pairs does not overwhelm the display of small distance results.

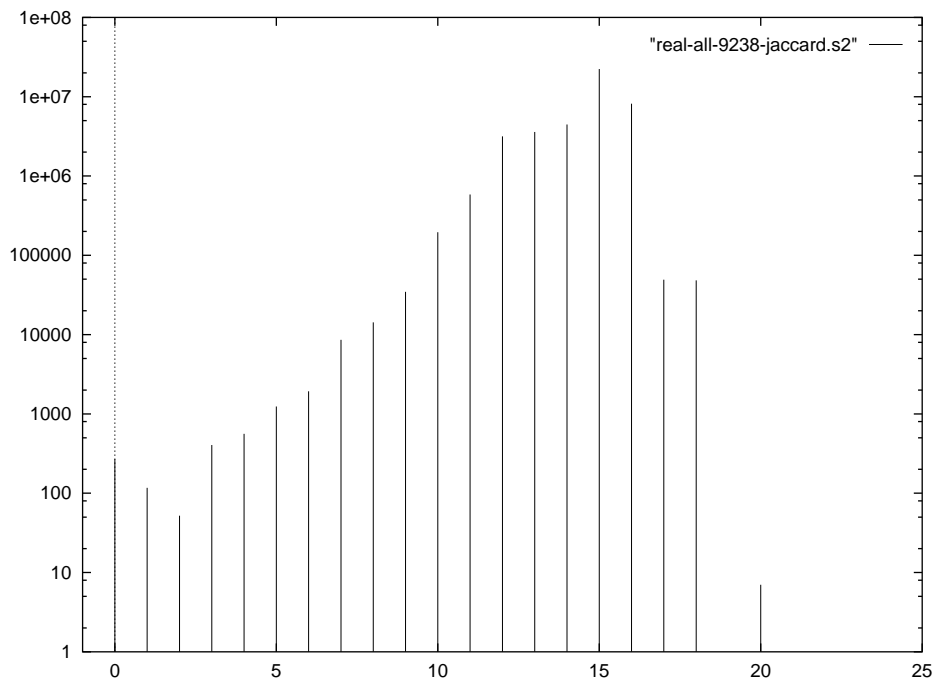


Figure 5.5: Distribution of Jaccard distance over WSJ sample sequences.

Two properties are distinctly evident in all of these plots. Perhaps the most obvious property is the ‘spikiness’ of the edit distance and pairwise comparison histograms, indicating that these measures have a sparse set of possible values. These measures are quite discrete, and cannot represent the kinds of fine distinctions that a linguistically realistic measure should have. Certainly, in order to be useful, any measure of linguistic similarity will have to have more than a handful of possible values.

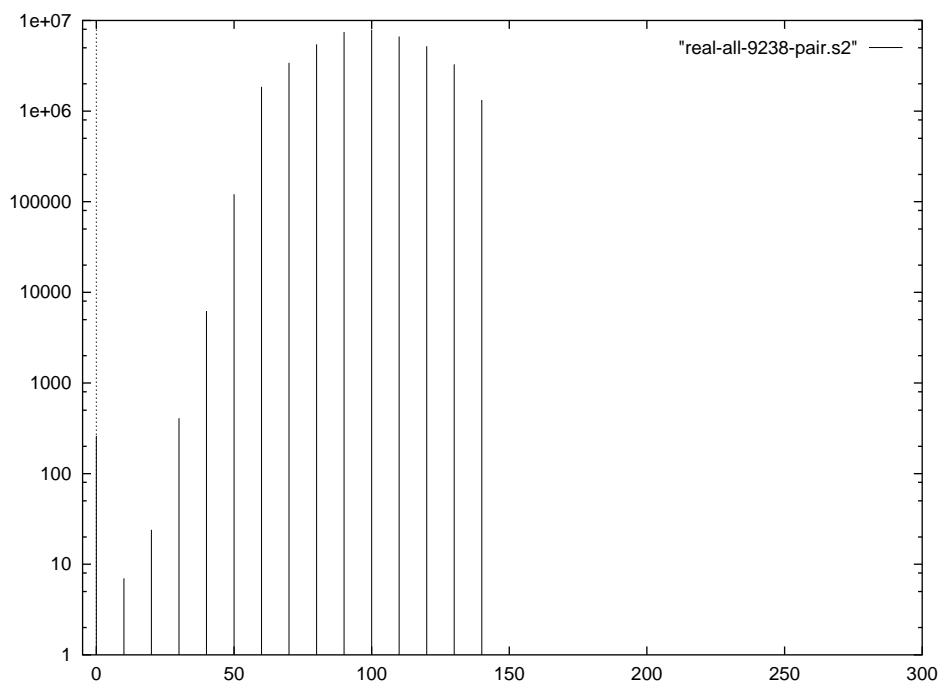


Figure 5.6: Distribution of pair-wise distance $P(a, b)$ over WSJ sample sequences.

The Jaccard measure suffers from this problem to a lesser degree. however, as discussed above, the Jaccard measure suffers from its insensitivity to positional information of any sort.

Real v. Zipf sequences

It is interesting to compare the plots of the similarity measures takes over the real English data (5.6, 5.5, 5.7) with those taken over the Zipf-distributed generated sequences. Plots 5.8, 5.9, and 5.10 show measures for the real data along with the generated data. In all three plots, it is evident that the real English data contains many more pairs with low distance than the generated data does.

This observation adds evidence to the assertion that the occurrences of words in real human languages are structured and interdependent. Therefore, these interactions and cannot and should not be represented by first order, independent models.

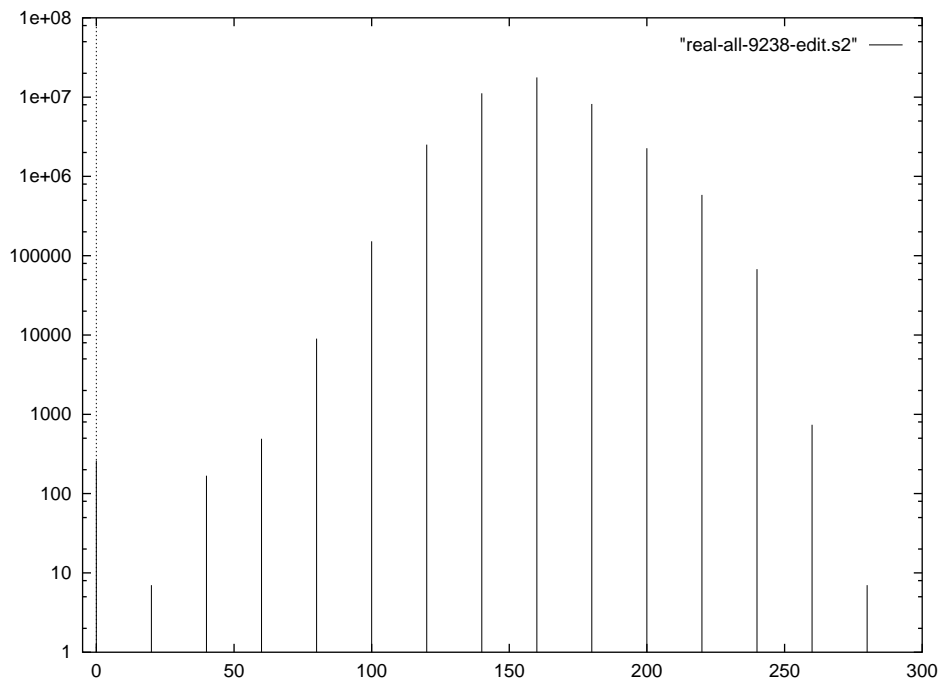


Figure 5.7: Distribution of edit distance over WSJ sample sequences.

Figure 5.8: Jaccard measure for real English sequences and Zipf-distributed generated sequences.

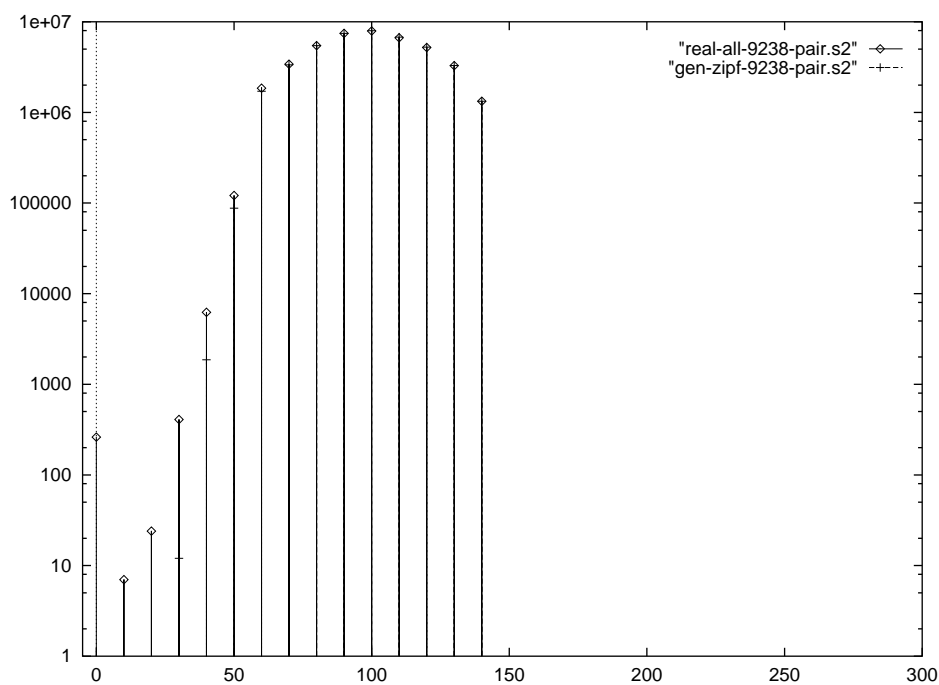


Figure 5.9: Pairwise distance distributions for WSJ sample compared with Zipf-distributed generate data. Again note the greater number of low-distance samples in the real language sample. (The generated data has a value of 0, plotted as $\log(0) = 1$, on the first three points.)

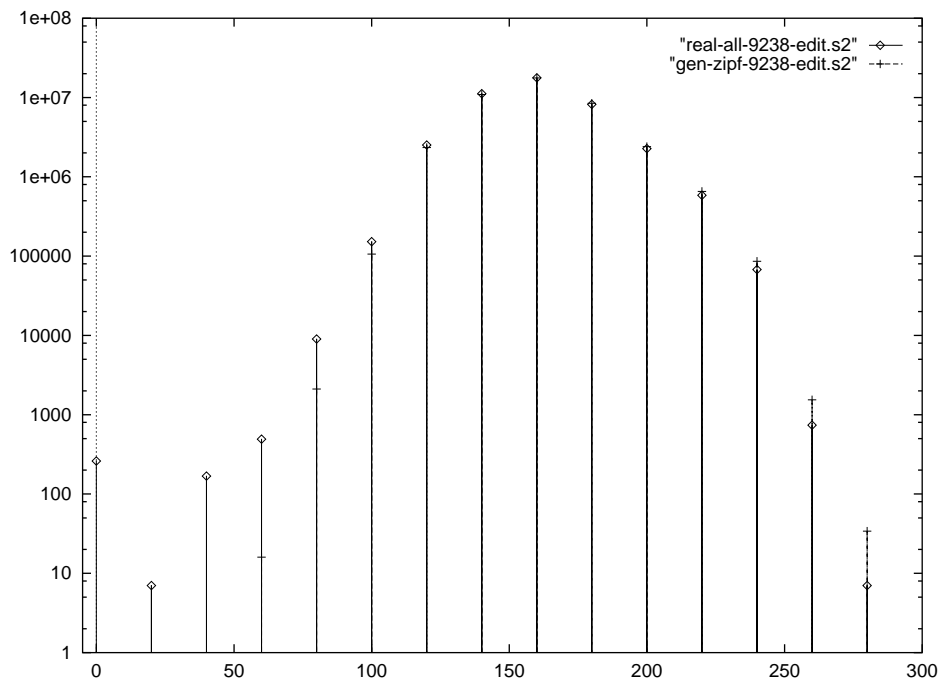


Figure 5.10: Edit distance distributions for real language sample and Zipf-distributed generated data. Note the prevalence of low-distance relations in the real data.

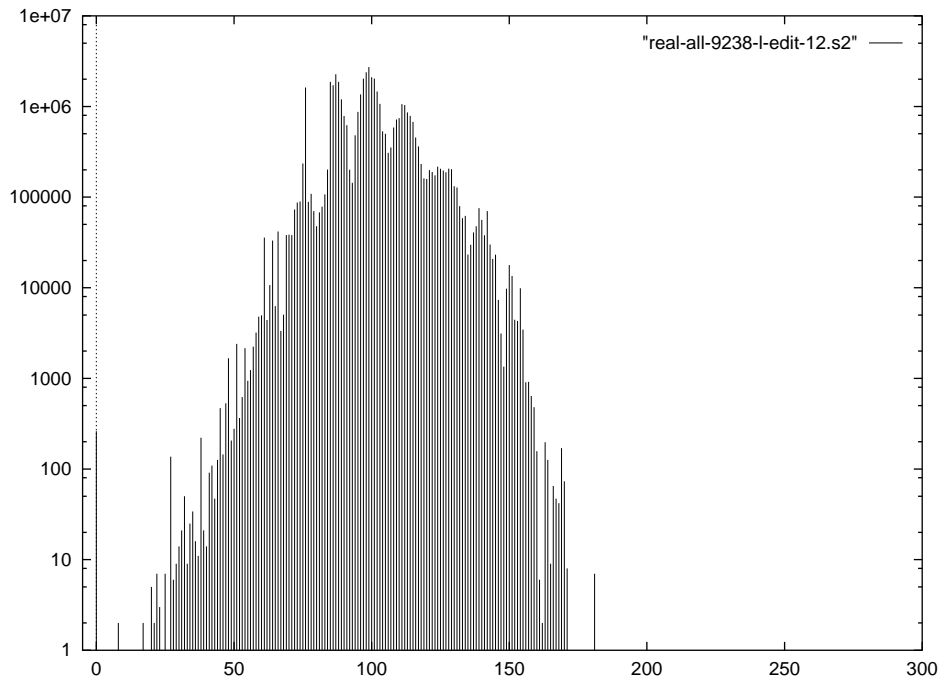


Figure 5.11: Distribution of localized edit distance over WSJ sample. ($P_h = 12$)

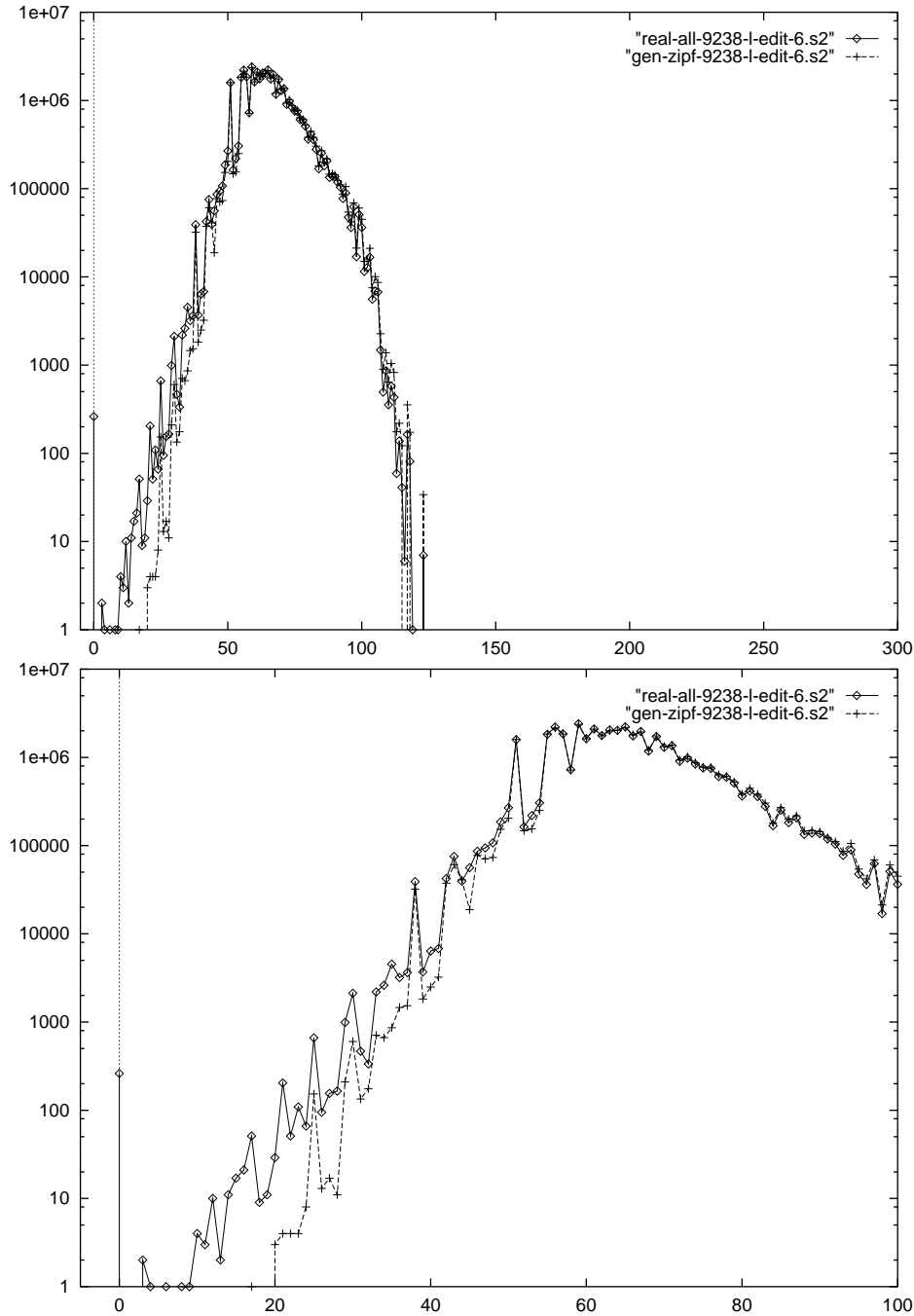


Figure 5.12: Edit distance ($P_h = 6$) distributions for WSJ and Zipf generated data.

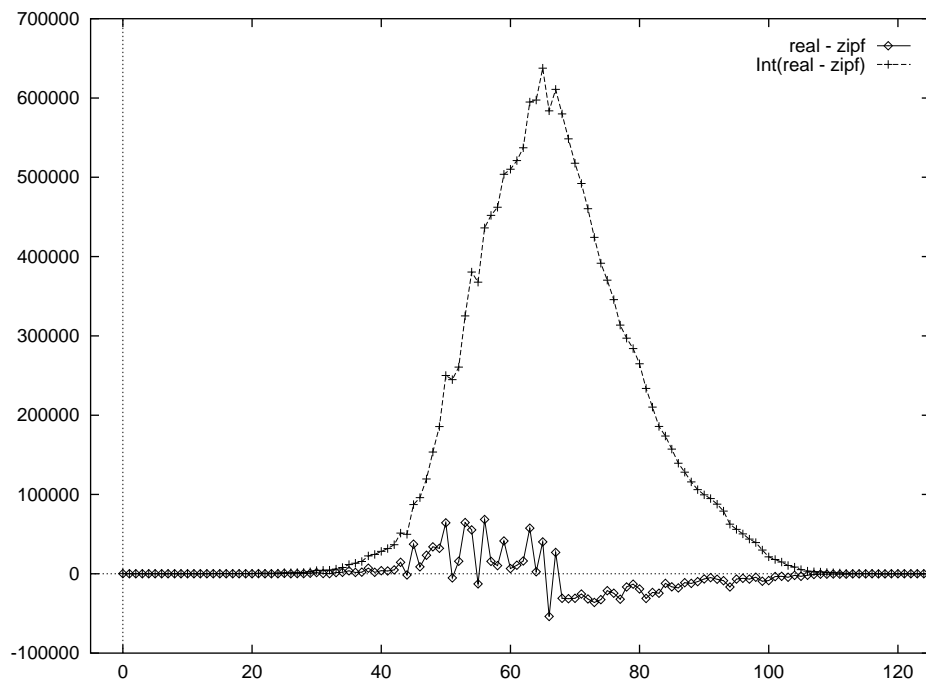


Figure 5.13: Difference $(R_n - Z_n)$ and integral difference $\sum_0^n (R_n - Z_n)$ of the counts of pairs at each distance n for actual English text and Zipf-distributed random data. ($P_h = 6$)

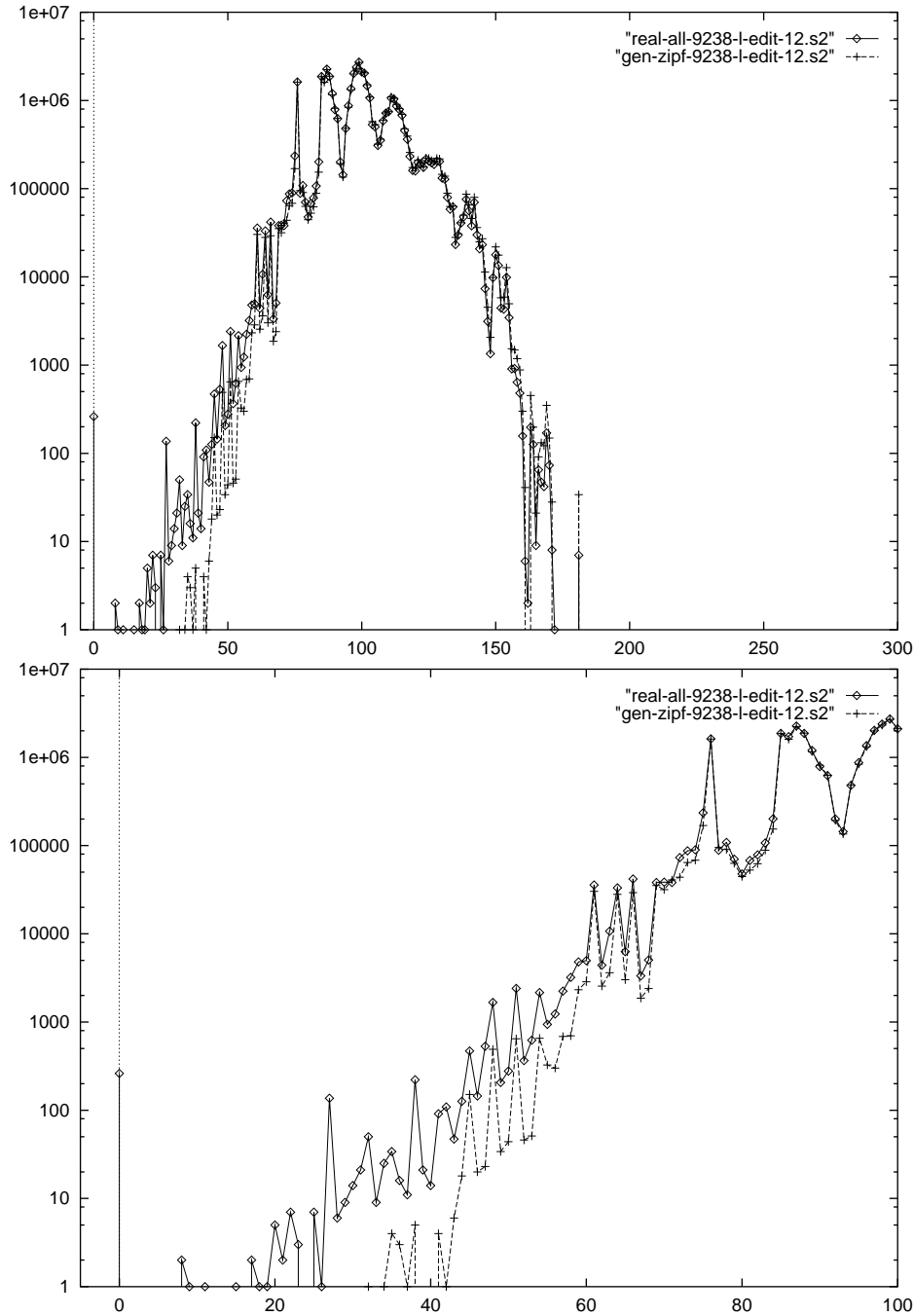


Figure 5.14: Edit distance ($P_h = 12$) distributions for WSJ and Zipf generated data.

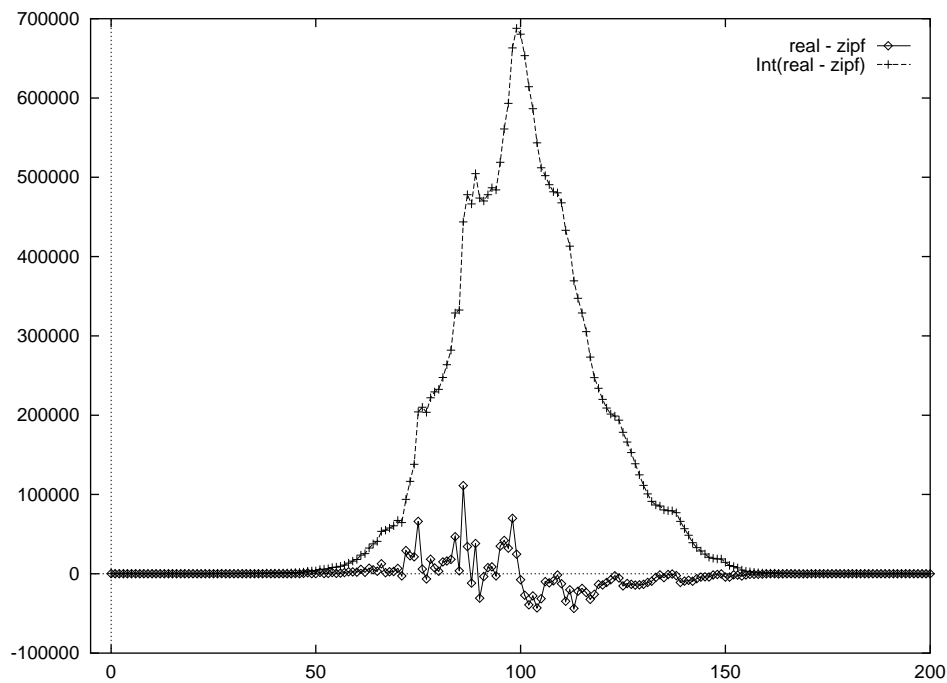
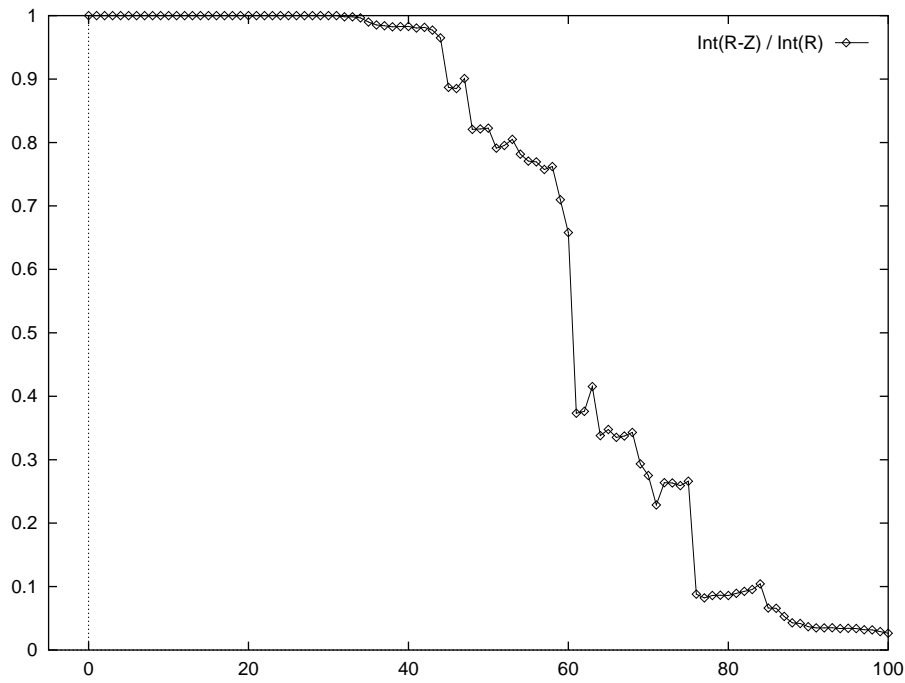


Figure 5.15: Difference $(R_n - Z_n)$ and integral difference $\sum_0^n (R_i - Z_i)$ of the counts of pairs at each distance n for actual English text and Zipf-distributed random data. ($P_h = 12$)



$$\frac{\sum_{i=0}^n (R_i - Z_i)}{\sum_{i=0}^n R_i}$$

Figure 5.16: Fraction of pairs not accounted for by the random data as a function of distance. (localized edit distance $P_h = 12$)

Chapter 6

Classification and Generalization

[why are we doing clustering?] Why generalize? reiterate motivation, purpose. i.e. – the need to abstract to an understandable level, which will explicate the understanding of the structure underlying the data.

The previous chapters have discussed methods of characterizing sets of contexts, and measures for comparing them, ¹ 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.

Outline:

The goal: associative class model

restate form of desired class model

Road map for building it:

Restate basic argument:

no direct similarity for words, must look at similarity of word use across contexts. Contexts themselves are too sparse, so we need methods of smoothing them, in order to make occurrences across different contexts somewhat comparable. Ideally, we could create a measure between the occurrence distributions

¹not entirely true! much of that discussion moved to this chapter!

of two words that took into account the context similarities between all of their occurrences, and do this for every pair of words, in order to judge word similarity and create word classes. This is simply too much work.

Instead, the plan is to form a set of *context classes*, Γ , which summarize a group of possible contexts, and then judge word similarity by the distribution of word occurrences across these context classes, γ_i . In the previous chapter, we looked at a number of context similarity measures that could be used for achieving this. In this chapter, we look at how to use that context similarity to construct context classes.

Once we have these, we can measure the occurrences of words across the γ_i . We then need to find a measures of distributional similarity, with which we can construct classes of words based on their distribution of usage across the γ_i . Sections §?? discusses some of the measures of distributional similarity that might be appropriate to this task. Most of the applicable measures, however, perform well in the limit of dense sample data, but have serious drawbacks when used over sparse data such as linguistic co-occurrences. In section §??, I introduce a method of re-estimating these measures based on a Monte Carlo simulation of their performance over plausible data.

The sections following that explore an iterative method for fitting a class-based associative model to the raw occurrence data, using sequence-similarity clustering of context data to form the context classes Γ , and distributional similarity of word occurrences within Γ to identify the lexical classes, L .

6.1 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.

6.1.1 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,

6.1.2 fixed

Fixed models (k -means) as a middle ground

6.1.3 Hierarchical

divisive

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

agglomerative

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

6.2 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.

6.3 Classification stability and convergence

a discussion of how an iterated classification behaves.

take a sample chunk of text, say 10-20k words of wsj, with POS tags.

find full distance measure between elements of the text, $D_{ij} = d(t_i, t_j)$ using word identity ($d(a, b) \stackrel{\text{def}}{=} [a \stackrel{?}{=} b]$) as a measure of interword distance

cluster into some pre-set number of classes.

Iteration step:

find full distance set between elements of the text, $D_{ij} = d'(t_i, t_j)$ using *class* identity as a measure of interword distance. $d'(a, b) \stackrel{\text{def}}{=} [\omega(a) \stackrel{?}{=} \omega(b)]$

cluster into same pre-set number of classes.

measure similarity of classification to previous classification

Repeat classification based on class membership.

- 1) How does classification change over time?
- 2) Does classification ever completely stabilize?

6.4 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:

$$\begin{aligned}
 \alpha_n^1 \dots \alpha_2^1 \alpha_1^1 k^1 \beta_1^1 \beta_2^1 \dots \beta_m^1 &= c_k^1 \\
 \alpha_n^2 \dots \alpha_2^2 \alpha_1^2 k^2 \beta_1^2 \beta_2^2 \dots \beta_m^2 &= c_k^2 \\
 &\vdots \\
 \alpha_n^l \dots \alpha_2^l \alpha_1^l k^l \beta_1^l \beta_2^l \dots \beta_m^l &= c_k^l
 \end{aligned} \tag{6.1}$$

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. Analysing the data by any means would involve looking at each element separately, and fails to capture the abstraction desired in forming such a category in the first place.

6.4.1 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 0th-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:

$$\dots \begin{bmatrix} p(\alpha_2 = w_1) \\ p(\alpha_2 = w_2) \\ p(\alpha_2 = w_3) \\ \vdots \end{bmatrix} \begin{bmatrix} p(\alpha_1 = w_1) \\ p(\alpha_1 = w_2) \\ p(\alpha_1 = w_3) \\ \vdots \end{bmatrix} k \begin{bmatrix} p(\beta_1 = w_1) \\ p(\beta_1 = w_2) \\ p(\beta_1 = w_3) \\ \vdots \end{bmatrix} \begin{bmatrix} p(\beta_2 = w_1) \\ p(\beta_2 = w_2) \\ p(\beta_2 = w_3) \\ \vdots \end{bmatrix} \dots \quad (6.2)$$

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

$$\dots Pr(\alpha_2) \quad Pr(\alpha_1) \quad k \quad Pr(\beta_1) \quad Pr(\beta_2) \quad \dots \quad (6.3)$$

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, 1993a] 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

$$s_1 = \{kAB, kBA\} \quad s_2 = \{kAA, kBB\} \quad (6.4)$$

have equivalent representations

$$\begin{bmatrix} p(A) = .5 \\ p(B) = .5 \end{bmatrix} \quad k \quad \begin{bmatrix} p(A) = .5 \\ p(B) = .5 \end{bmatrix} \quad (6.5)$$

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

[Maybe just change this to a ref. to a related work section]

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 and Gauch, 1993]), cosine measures over a reduced space ([Schütze, 1993a]), or entropy measures ([Chan and Wang, 1991]). Various clustering methods were used to form categories based on these measures.

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:

$$Pr(\beta_n | \beta_1 \beta_2 \dots \beta_j = w_1 w_2 \dots w_j), \quad j = \min(v, n - 1) \quad (6.6)$$

(Note that this is not the standard notation used for in n -gram analyses or Markov modelling. In this notation, $Pr(\beta_2 | \beta_1 = w_1)$ is the distribution of the second position, β_2 , only for those cases in which $\beta_1 = w_1$. Similarly, $Pr(\beta_3 | \beta_1 = w_1)$ is the distribution of the third position, β_3 , when the *first* position is $\beta_1 = w_1$.)

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:

$$k \begin{bmatrix} p(\beta_1 = w_1) & [Pr(\beta_2 | \beta_1 = w_1) \quad Pr(\beta_3 | \beta_1 = w_1)] \\ p(\beta_1 = w_2) & [Pr(\beta_2 | \beta_1 = w_2) \quad Pr(\beta_3 | \beta_1 = w_2)] \\ \vdots & \end{bmatrix} \quad (6.7)$$

Chapter 7

Evaluating various models

7.1 Predictive accuracy: perplexity

7.2 Comparison to Part-of-Speech tagging

7.3 Word sense identification

stuff

Bibliography

[,]

[Anderberg, 1973] Michael R. Anderberg. *Cluster Analysis for Applications*. Academic Press, Inc., London, 1973. originally presented as Anderberg's thesis, University of Texas, Austin, 1971.

[Ass, 1991] Association for Computational Linguistics. *Proceedings of the 29th Annual Meeting of the ACL*, 1991.

[Ayuso *et al.*, 1992] Damaris Ayuso, Sean Boisen, Heidi Fox, Robert Ingria, and Ralph Weischedel. Bbn: Description of the plum system as used for muc-4. In *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, 1992.

[Baker, 1975] Janet K. Baker. Stochastic modeling for automatic speech understanding. In D. R. Reddy, editor, *Speech Recognition*. Academic Press, 1975.

[Basili *et al.*, 1993] Roberto Basili, Maria Teresa Pazienza, and Paola Velardi. Hierarchical clustering of verbs. In *Acquisition of Lexical Knowledge from Text, Proceedings of the SIGLEX Workshop*, 1993.

[Black *et al.*, 1992] Ezra Black, Fred Jelinek, John Lafferty, David Magerman, Robert Mercer, and Salim Roukos. Towards history-bases grammars: Using richer models of probabilistic parsing. In *Proceedings of the Fifth DARPA Workshop on Speech & Natural Language*, 1992.

- [Bloomfield, 1933] Leonard Bloomfield. *Language*. Holt, New York, 1933.
- [Bod, 1992] Rens Bod. A computational model of language performance: Data oriented parsing. In *Coling*, pages 855–859, 1992.
- [Boguraev and Briscoe, 1989] Branimir Boguraev and Ted Briscoe, editors. *Computational Lexicography for Natural Language Processing*. Longman, London, 1989. guidelines to using Longman’s LDOCE for lexical acquisition.
- [Breiman *et al.*, 1984] Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. *Classification and regression trees*. Wadsworth & Brooks, Monterey, California, 1984.
- [Brent, 1991] Michael R. Brent. Automatic acquisition of subcategorization frames from untagged text. In *Proceedings of the 29th Annual Meeting of the ACL*, 1991.
- [Brown *et al.*, 1992] Peter F. Brown, Vincent J. Della Pietra, Peter V. deSouza, Jennifer C. Lai, and Robert L. Mercer. Class-based n -gram models of natural language. *Computational Linguistics*, 18(4):467–479, 1992.
- [Cardie, 1993] Claire Cardie. A case-based approach to knowledge acquisition for domain specific sentence analysis. In *Proceedings of the 11th National Conference on Artificial Intelligence*, pages 798–803, Menlo Park, 1993. AAAI Press.
- [Chan and Wang, 1991] S.C. Chan and A.K.C. Wang. Synthesis and recognition of sequences. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(12):1245–1255, 1991.
- [Cheeseman *et al.*, 1988] Peter Cheeseman, James Kelly, Matthew Self, John Stutz, Will Taylor, and Don Freeman. Autoclass: A bayesian classification system. In John Laird, editor, *Proceedings of the Fifth International Machine Learning Conference*, pages 54–64, San Francisco, 1988. Morgan Kaufmann.

- [Chen and joshua T. Goodman, 1996] Sanley F. Chen and joshua T. Goodman. An empirical study of smoothing techniques for language modeling. pages 310–318, Morristown, New Jersery, 1996. ACL.
- [Chomsky, 1957] Noam Chomsky. *Syntactic Structures*. Mouton & Co., The Hague, 1957.
- [Chomsky, 1965] Noam Chomsky. *Aspects of the Theory of Syntax*. MIT Press, Cambridge, 1965.
- [Church and Gale, 1991] Kenneth W. Church and William A. Gale. A comparison of the enhanced good-turing and deleted estimation method for estimating probabilities of english bigrams. *Computer speech and Language*, 5, 1991.
- [Church and Hanks, 1990] Kenneth Ward Church and Patrick Hanks. Word association norms, mutual information and lexicography. *Computational Linguistics*, 16(1):22–29, 1990.
- [col, 1988] Proc. of the 12th International Conference on Computational Linguistics, Budapest, 1988.
- [col, 1992] *Proceedings of the 14th International Conference on Computational Linguistics*, Nantes, France, 1992.
- [Cowie *et al.*, 1993] Jim Cowie, Louise Guthrie, James Pustejovsky, Takahiro Wakao, Jin Wang, and Scott Waterman. The diderot information extraction system. In *Proceedings of the First Conference of the Pacific Association for Computational Linguistics*, Vancouver, 1993.
- [Dagan *et al.*, 1993] Ido Dagan, Shaul Marcus, and Shal Markovitch. Contextual word similarity and estimation from sparse data. In *Proceedings of the 31st Annual Meeting of the ACL*, 1993.

- [Dagan *et al.*, 1994a] Ido Dagan, Fernando Periera, and Lillian Lee. Similarity-based estimation of word cooccurrence probabilities. In *Proceedings of the 32nd Annual Meeting of the ACL*, 1994.
- [Dagan *et al.*, 1994b] Ido Dagan, Fernando Pereira, and Lillian Lee. Similarity-based estimation of word cooccurrence probabilities. In *Proceedings of the 32nd Annual Meeting of the ACL*, pages 272–278, Morristown, New Jersey, 1994. ACL.
- [Dagan *et al.*, 1997] Ido Dagan, Lillian Lee, and Fernando Pereira. Similarity-based methods for word sense disambiguation. pages 56–63, Morristown, New Jersey, 1997. ACL.
- [Davison, 1983] Mark L. Davison. *Multidimensional Scaling*. Wiley Series in probability and mathematical statistics. John Wiley & Sons, 1983.
- [DeMarcken, 1990] Carl DeMarcken. Parsing the lob corpus. In *Proceedings of the 28th Annual Meeting of the ACL*, 1990.
- [Dempster *et al.*, 1977] Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, 39:1–38, 1977.
- [Dunn and Everitt, 1982] G. Dunn and Brian S. Everitt. *An introduction to numerical taxonomy*. Cambridge University Press, 1982.
- [Edwards, 1972] A. W. F. Edwards. *Likelihood*. Cambridge University Press, 1972.
- [Elliot *et al.*, 1995] Robert J. Elliot, Lakhdar Aggoun, and John B. Moore. *Hidden Markov Models*. Number 29 in Applications of Mathematics. Springer-Verlag, 1995.
- [Everitt, 1980] Brian Everitt. *Cluster Analysis*. Halsted Press, second edition, 1980.

- [Fillmore, 1968] Charles Fillmore. The case for case. In Emmon Bach and R. T. Harms, editors, *Universals in Linguistic Theory*. Holt, Rinehart & Winston, New York, 1968.
- [Finch and Chater, 1992] Steven Finch and Nick Chater. Bootstrapping syntactic categories using statistical methods. In Walter Daelemans and David Powers, editors, *Background and Experiments in Machine Learning of Natural Language*, 1992.
- [Fu and Booth, 1975] King-Sun Fu and Taylor L. Booth. Grammatical inference: Introduction and survey. *IEEE Transactions on Systems, Man, and Cybernetics*, 5(1,4), 1975.
- [Fukunaga, 1990] Keinosuke Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press, Inc., San Diego, second edition, 1990.
- [Futrelle and Gauch, 1993] Robert Futrelle and Susan Gauch. Experiments in syntactic and semantic classifications and disambiguations using bootstrapping. In *Acquisition of Lexical Knowledge from Text, Proceedings of the SIGLEX Workshop*, 1993.
- [Gale and Church, 1990] William A. Gale and Kenneth W. Church. Poor estimates of context are worse than none. In *Proceedings of the 28th Annual Meeting of the ACL*, 1990.
- [Gale, 1986] William A. Gale, editor. *Artificial intelligence and statistics*. Addison-Wesley, 1986.
- [Gonzales, 1978] Rafael C. Gonzales. *Syntactic pattern recognition*. Addison-Wesley, 1978.
- [Good, 1953] Irving. J. Good. The population frequencies of species and the estimation of population parameters. *Biometrika*, 40(3,4):237–264, 1953.

- [Goodman *et al.*, 1991] Marc Goodman, Scott Waterman, and Rick Alterman. Interactive reasoning about spatial components. In *Proceedings 13th Conference of the Cognitive Science Society*, Hillsdale, NJ, 1991. Earlbaum.
- [Greenacre, 1984] Michael J. Greenacre. *Theory and Applications of Correspondence Analysis*. Academic Press, 1984.
- [Grefenstette, 1994] Gregory Grefenstette. *Explorations in automatic thesaurus discovery*. Kluwer, Boston, 1994.
- [Grishman and Sterling, 1992] Ralph Grishman and J. Sterling. Acquisition of selectional patterns. In *Proceedings of the 14th International Conference on Computational Linguistics*, 1992.
- [Grishman *et al.*, 1992] Ralph Grishman, C. Macleod, and J. Sterling. New york university: Description of the proteus system as used for muc-4. In *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, 1992.
- [Hindle and Rooth, 1991] Don Hindle and Mats Rooth. Structural ambiguity and lexical relations. In *Proceedings of the 29th Annual Meeting of the ACL*, 1991.
- [Hobbs and Appelt, 1992] Jerry Hobbs and Doug Appelt. Sri international: Description of the fastus system used for muc-4. In *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, 1992.
- [Hobbs *et al.*, 1990] Jerry Hobbs, Doug Appelt, and M. Pal. Interpretation as abduction. Technical Report Technical Note 449, SRI International AI Center, Palo Alto, 1990.
- [Hogeweg and Hesper, 1984] P. Hogeweg and B. Hesper. The alignment of sets of sequences and the construction of phyletic trees: An integrated method. *J. Molecular Evolution*, 20:175–186, 1984.
- [Hughes, 1994] John Hughes. *Automatically acquiring a classification of words*. PhD thesis, University of Leeds, 1994.

- [Jeffreys, 1939] Harold Jeffreys. *Theory of probability*. Oxford University Press, third edition, 1939.
- [Jelinek *et al.*, 1992] Fred Jelinek, John D. Lafferty, and Robert L. Mercer. Basic methods of probabilistic context free grammars. Technical report, Continuous speech recognition group, IBM T.J. Watson Research Center, 1992.
- [Jobson, 1992] J. D. Jobson. *Applied Multivariate Data Analysis*, volume 2. Springer-Verlag, 1992.
- [Johansson and Stenström, 1991] Stig Johansson and Anna-Brita Stenström, editors. *English Computer Corpora: Selected Papers and Research Guide*. Mouton de Gruyter, Berlin, 1991.
- [Kamp, 1981] Hans Kamp. A theory of truth and semantic representation. In J. A. G. Groenendijk, T. M. V. Janssen, and M. B. J. Stokhof, editors, *Formal Methods in the Study of Language*, Mathematical Centre Tracts. Amsterdam, 1981.
- [Katz, 1987] Slava Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE TASSP*, 35(3), 1987.
- [Langacker, 1987] R. W. Langacker. *Foundations of Cognitive Grammar, i*. Stanford University Press, Stanford, 1987.
- [Lehnert *et al.*, 1991] Wendy Lehnert, Claire Cardie, D. Fisher, Ellen Riloff, and R. Williams. In *Third Message Understanding Conference (MUC-3)*, pages 223–233, 1991.
- [Lehnert *et al.*, 1992] Wendy Lehnert, Claire Cardie, D. Fisher, John McCarthy, Ellen Riloff, and S. Soderland. University of massachusetts: Muc-4 test results and analysis. In *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, 1992.

- [Levenshtein,] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Cybernetics and control Theory*, 10(8):707–710. originally published as [Levenshtein, 1965].
- [Levenshtein, 1965] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Doklady Akademii Nauk SSR*, 163(4):845–848, 1965.
- [Lu and Fu, 1977] S. Y. Lu and King Sun Fu. A clustering procedure for syntactic patterns. *IEEE Trans. on Systems, Man, and Cybernetics*, October 1977.
- [Meter et al., 1991] Marie Meter, Richard Schartz, and Ralph Weischedel. Empirical studies in part of speech labelling. In *Proc. of the 4th DARPA Workshop on Speech and Natural Language*, pages 331–336, 1991.
- [muc, 1992] *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, San Mateo, 1992. Morgan Kaufman.
- [Novák, 1992] Vilém Novák. *The alternate mathematical model of linguistic semantics and pragmatics*. Number 8 in IFSR international series on systems science and engineering. Plenum Press, New York, 1992.
- [Periera et al., 1993] Fernando Periera, Naftali Tishby, and Lillian Lee. Distribution clustering of english words. In *Proceedings of the 31st Annual Meeting of the ACL*, 1993.
- [Press et al., 1986] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William Vetterling. *Numerical Recipes*. Cambridge University Press, 1986.
- [Pustejovsky and Anick, 1988] James Pustejovsky and Peter Anick. The semantic interpretation of nominals. In *Proc. of the 12th International Conference on Computational Linguistics*, 1988.

- [Pustejovsky and Boguraev, 1993] James Pustejovsky and Branimir Boguraev. Lexical knowledge representation and natural language processing. *Artificial Intelligence*, 1993.
- [Pustejovsky, 1991] James Pustejovsky. The generative lexicon. *Computational Linguistics*, 17(4), 1991.
- [Pustejovsky, 1992] James Pustejovsky. The acquisition of lexical semantic knowledge from large corpora. In *Proceedings of the Fifth DARPA Workshop on Speech & Natural Language*, 1992.
- [Pustejovsky, 1995] James Pustejovsky. *The Generative Lexicon*. MIT Press, 1995.
- [Salton and Schneider, 1983] Gerald Salton and Hans-Jochen Schneider, editors. *Research and Development in Information Retrieval*. Number 146 in Lecture Notes in Computer science. Springer-Verlag, Berlin, 1983.
- [Sankoff and Kruskal, 1983] D. Sankoff and J.B. Kruskal, editors. *Time warps, string edits, and macromolecules*. Addison Wesley, Reading, MA, 1983.
- [Schütze, 1992] Hinrich Schütze. Context space. In *Fall Symposium on Probabilistic Approaches to Natural Language*, pages 113–120, Menlo Park, 1992. AAAI Press.
- [Schütze, 1993a] Hinrich Schütze. Part of speech induction from scratch. In *Proceedings of the 31st Annual Meeting of the ACL*, 1993.
- [Schütze, 1993b] Hinrich Schütze. Word space. In S. J. Hanson, J. D. Cowan, and C. L. Giles, editors, *Advances in Neural Information Processing Systems 5*, pages 895–902. Morgan kaufmann, San Francisco, 1993.
- [Sellers, 1974] P.H. Sellers. An algorithm for the distance between two finite sequences. *J. Comb. Thy*, A16:253–258, 1974.

- [Skakakibara *et al.*, 1993] Yasabumi Skakakibara, Michael Brown, Rebecca C. Underwood, I. Saira Mian, and David Haussler. Stochastic context-free grammars for modeling rna. Technical report, U.C. Santa Cruz, 1993.
- [Smadja, 1989] Frank Smadja. Macrocoding the lexicon with co-occurrence knowledge. In *First Int'l Language Acquisition Workshop*, IJCAI 89, 1989.
- [Sowa, 1984] John F. Sowa. *Conceptual structures: Information processing in Mind and Machine*. Addison-Wesley, Reading, Mass., 1984.
- [Stanfill and Waltz, 1986] Craig Stanfill and David Waltz. Toward memory-based reasoning. *Communications of the ACM*, 29(12):1213–1228, 1986.
- [Taylor, 1995] John R. Taylor. *Linguistic Categorization: Prototypes in Linguistic Theory*. Oxford University Press, New York, 1995.
- [Ushioda, 1996] A. Ushioda. Hierarchical clustering of words and applications to nlp tasks. In E. Ejerhed and Ido Dagan, editors, *Fourth Workshop on Very Large Corpora*, pages 28–41. ACL, Somerset, New Jersey, 1996.
- [ute Essen and Steinbiss, 1992] ute Essen and Volker Steinbiss. Co-occurrence smoothing for stochastic language modeling. In *Proceedings of ICASSP*, pages 161–164, 1992.
- [Wagner and Fisher, 1974] R. A. Wagner and M. J. Fisher. The string-to-string correction problem. *J. ACM*, 21:168–173, 1974.
- [Waterman, 1993] Scott A. Waterman. Structural methods for lexical/semantic patterns. In *Acquisition of Lexical Knowledge from Text, Proceedings of the SIGLEX Workshop*, 1993.
- [Waterman, 1995] Scott A. Waterman. Distinguished usage. In Branimir Boguraev and James Pustejovsky, editors, *Corpus Processing for Lexical Acquisition*. MIT Press, 1995.

- [Wells, 1957] Rulon S. Wells. Immediate constituents. In Martin Joos, editor, *Readings in Linguistics I*. University of Chicago Press, Chicago, 1957.
- [Wilks *et al.*, 1990] Yorick Wilks, Dan Fass, C. M. Gou, J. E. McDonald, T. Plate, and Brian M. Slator. Providing machine tractable dictionary tools. *Machine Translation*, 5, 1990.
- [wsj, 1989] The wall street journal. dist.'d by the Linguistic Data Consortium, 1989.
- [Zernik, 1990] Uri Zernik. Lexical acquisition: Where is the semantics? *Machine Translation*, 5:155–174, 1990.
- [Zipf, 1972] George Kingsley Zipf. *Human behavior and the principle of least effort : an introduction to human ecology*. Hafner Pub. Co., 1972. facsimile of 1949 Addison Wesley edn.