Some Statistical Opportunities in Speech and Language

Kenneth Ward Church

Abstract

Text analysis is a hot topic, and for good reason. Text is more available than ever before. Just ten years ago, the one-million word Brown Corpus (Francis and Kucera, 1982) was still considered large, but even then, there were much larger corpora in use such as the 18 million word Birmingham Corpus (Sinclair 1987a, 1987b). These days, there are many places that regularly use samples of text running into the hundreds of millions of words. And it is very likely that billions of words will be available very soon.

All of this data provides a great research opportunity; it easier these days to corpus data much more effectively than it was in the 1950s, the last time that empiricism was in fashion. Text analysis focuses on broad (though possibly superficial) coverage of unrestricted text, rather than a deep analysis of a restricted domain. This pragmatic view toward coverage and performance distinguishes text analysis from so-called ‘‘intelligent’’ approaches such as natural language understanding. This approach has produced a number of tools such as spelling correctors and part of speech taggers that work on unrestricted text, with reasonable accuracy and efficiency.

1. Recognition Applications

Recognition applications are perhaps the most obvious applications for large bodies of text. Three examples of recognition applications will be mentioned here: (1) Speech Recognition, (2) Optical Character Recognition (OCR), and (3) Spelling Correction.

Imagine a noisy channel, such as a speech recognition machine that almost hears, an optical character recognition (OCR) machine that almost reads, or a typist that almost types. Good text (W_i) goes into the channel, and corrupted text (W_o) comes out the other end.

$$W_i \rightarrow Noisy\ Channel \rightarrow W_o$$

How can an automatic procedure recover the good input text, W_i, from the corrupted output, W_o? In principle, one can recover the most likely input by hypothesizing all possible input texts, W_i, and selecting the input text with the highest score. Using a classic Bayesian argument, the score is computed by taking the product of the prior probability, $Pr(W_i)$, and the channel probability, $Pr(W_o | W_i)$. This procedure can be written as:

$$\underset{W_i}{ARGMAX}\ Pr(W_i)\ Pr(W_o | W_i)$$

where ARGMAX finds the argument with the maximum score.

The prior probability, also known as the language model, is the probability that the W_i would be input to the channel. For example, in the speech recognition application, it is the probability that someone would utter W_i, whereas in the spelling correction application, it is the probability that someone would type W_i. In practice, the prior is approximated by computing various statistics over a large sample of text.

The channel probability is the probability that the channel would transform the word sequence W_i into the sequence W_o. This is relatively high if W_i is the same as or very ‘‘similar’’ to W_o, where the definition of ‘‘similar’’ depends on the application. The channel for speech recognition, for example, will have a high probability of mapping words that sound similar (e.g., ‘‘writer’’ and ‘‘rider’’ in many American dialects) into the same output representation. However, in other applications such as optical character recognition, ‘‘writer’’ and ‘‘rider’’ are unlikely to be confused by the channel because these words are optically quite distinct. Thus, the channel model clearly depends on the application as illustrated in the Table 1.

Table 1: Examples of Channel Confusions in Different Applications

Application	Input	Output
Speech Recognition	writer	rider
	here	hear
Optical Character Recognition	all	a1l (*A-one-L*)
	of	o{
	form	farm
Spelling Correction	government	goverment
	occurred	occured
	commercial	commerical

It is convenient to partition the prior and the channel in this way, so that the same prior can be used for a variety of recognition applications including speech recognition, optical character recognition and spelling correction. The channel, of course, generally cannot be ported from one application to another.

2. Spelling Correction

I have found that spelling correction is a good application to look at because it is analogous to many important recognition applications based on a noisy channel model (such as speech recognition), though somewhat simpler and therefore possibly more amenable to detailed statistical analysis. In (Kernighan, Church, and Gale, 1990), we described a program called *correct* which inputs a misspelled word such as *absurb*, and outputs a list of candidate corrections sorted by probability: *absorb* (56%), *absurd* (44%). The probability scores are the novel contribution; there are have been many programs in the past that generated a (long) list of candidate corrections, but few have attempted to score the candidates by a stochastic model of the prior probability of observing the candidate correction $Pr(c)$ and a channel probability of observing a particular typo given the candidate correction $Pr(t|c)$. Both of these probabilities were estimated from about 50 million words of Associated Press newswire (which includes about 15,000 typos which are used to train the channel model).

In evaluating the program, we restricted our attention to 564 typos that had exactly two candidate corrections. A panel of three judges were given the typo (e.g., *absurb*), the two candidate corrections (e.g., *absorb* and *absurd*) and a concordance line (e.g., ‘‘it is *absurb* and probably obscene for...’’), and were asked to select one of the two corrections (or none-of-the-above). The judges found this task more difficult than they had anticipated, and very time consuming (it took each judge about four hours to grade the 564 examples). In addition, the judges felt that the task would have been much harder without the concordance line, suggesting that context should be incorporated into the program.

Table 2 shows that *correct* agrees with the majority of the judges in 87% of the 332 cases of interest.[1] In order to help calibrate this result, we compared *correct* to three inferior methods: *channel-only*, *prior-only* and *chance*. Table 2 shows that both the channel-only and the prior-only models provide a significant contribution over chance, and that *correct*, which is a combination of the two, is significantly better than either in isolation. Table 2 also shows that the judges are significantly better than all of the programs, indicating that there is room for improvement.

1. We restricted our attention to those cases where at least two judges selected one of the two candidate corrections, and they agreed with each other.

Table 2: Evaluation of *Correct*

Method	Discrimination	%
correct	286/ 329	87 ± 1.9
Judge 1	271/ 273	99 ± 0.5
Judge 2	271/ 275	99 ± 0.7
Judge 3	271/ 281	96 ± 1.1
channel-only	263/ 329	80 ± 2.2
prior-only	247/ 329	75 ± 2.4
chance	172/ 329	52 ± 2.8

The program, of course, is not making use of context whereas the human judges did have access to a concordance line. The following examples show that the task is extremely difficult without context.

Table 3: Hard without Context

Typo	Choice 1	Choice 2
actuall	actual	actually
constuming	consuming	costuming
conviced	convicted	convinced
confusin	confusing	confusion

Of course, the task becomes much easier if the context is provided as demonstrated by the following four concordance lines.

in determining whether the defendant *actuall* will die. In the 1985 decision, the...
on Friday night, a show as lavish in *constuming* and lighting as those the late Liberace used to...
of the area. ‘‘When we’re *conviced* and the Peruvians are convinced (the base camp)...
The political situation grew more *confusin* today, with an official media report indicating...

Both (Mays *et al.*, 1990) and (Church and Gale, 1991a) have found that statistical n-gram models of context can help considerably, although performance is still far below that of the human judges. A quick look at the concordance lines above shows (a) that the relevant contextual clues are often fairly close to the typo, and (b) that there are relatively few cases that make use of long-distance syntactic dependencies. (a) suggests that simple n-gram methods might work fairly well in many cases, and (b) suggests that more complicated ‘‘intelligent’’ parsing methods might not be worth the trouble.

3. The Trigram Model

One of the simpler and more popular priors is the n-gram model. This model makes the simplifying assumption that word probabilities depend on only the previous $n-1$ words, and that long-distance dependences which extend beyond this limited window can be ignored. Jelinek (1985) uses the example shown in Table 4 to illustrate the power of the trigram model. In the sentence, *We need to resolve all the important issues within the next two days*, most of the words are extremely predictable from the trigram context (the current word plus the previous two). Note that *we* is the 9^{th} most likely word to begin a sentence in his model; the words *the, this, one, ..., in* are more likely to begin a sentence than *we*. The word *need* is found to be the 7^{th} most likely word to follow *we*; the words *are, will, ..., do* are more likely than *need*. And so on. Jelinek uses this example to argue that the rank is usually very small in comparison to the vocabulary size, which was 20,000 words in this example.

Table 4: Example of Trigrams (Jelinek, 1985)

The This One Two A Three Please In *We*	9
are will the would also do *need*	7
to	1
know have understand ... *resolve*	98
the this these problems ... *all*	9
issues problems *the*	3
necessary data information ... *important*	641
role thing that ... *issues*	9
and from in to are with ... *within*	66
the	1
next	1
be *two*	2
meeting months years ... *days*	7

Note that function words (e.g., *to, the*) are generally more predictable than content words (e.g., *resolve*, *important*). This turns out to be important in speech recognition because the shorter function words are more easily confused by the channel model and so it is fortunate that they are more predictable from context.

Some of the content words also have relatively small ranks. Consider, the content word *issues*, for example. It turns out that there are relatively few words that follow the word *important* (at least, in the subdomain of IBM office correspondences). This kind of *collocational* (or co-occurrence)[2] constraint between words are often not captured very well with a syntactic parser. Perhaps this is the reason why trigram models have tended to out-perform so-called ‘‘intelligent’’ approaches, when performance is measured in terms of entropy.

4. Word Frequencies and Word Association Norms

The trigram model does a good job of modeling word frequencies which are very important, as any psycholinguist knows. Generally speaking, subjects respond more quickly and more accurately to a high frequency word (e.g., a word that appears relatively often in a sample of text such as the Brown Corpus) than to an unusual low frequency word. The word association effect is similar except that it involves pairs of words. In general, subjects respond more quickly and more accurately to a word like *doctor* if it follows a highly associated word such as *nurse* (Meyer, Schvaneveldt and Ruddy, 1975, p. 98).

Word frequencies are fairly easy to estimate from a sample of text such as the Brown Corpus. Hanks and I have argued that word associations should also be estimated by computing various statistics over large corpora (Church and Hanks, 1990). It is more common in the psycholinguistic literature to find a study like (Palermo and Jenkins, 1964); they estimated word association norms for 200 words by asking a few thousand subjects (psychology undergraduates) to write down a word after each of the words to be measured. Results were reported in tabular form, indicating which words were written down, and by how many subjects, factored by grade level and sex. The word *doctor*, for example, is reported on pp. 98-100, to be most often associated with *nurse*, followed by *sick, health, medicine, hospital, man, sickness, lawyer*, and about 70 more words.

2. Halliday (1966, p. 150) was very interested in the difference between *strong* and *powerful*. Although both words have very similar syntax and semantics, there do seem to be some contexts where one word is much more appropriate than the other, e.g., *strong tea* vs. *powerful drugs*. The terms *collocation*, *co-occurrence* and *lexis* have been used to describe these kinds of constraints on pairs of words.

5. Strengths and Weaknesses

The main advantage of the trigram model is that it has very low entropy, 1.76 bits per character (Brown *et al.*, 1991). Parsers generally don't do as well because they tend to ignore word frequencies. The trigram model is also able to capture some collocations and word associations.

The most obvious weakness with the trigram model is the lack of syntax; the model makes no attempt to capture long-distance dependencies such as syntactic agreement, conjunction and wh-movement. In fact, the lack is syntax is probably not the most serious problem with the model. The sparse-data problem is extremely serious since many trigrams do not appear very often in the training corpus, if at all. In addition, the trigram model assumes that trigrams have a binomial distribution, an assumption which is often violated in practice.

6. Parsers May Not Help Very Much

It has been common practice, especially during the first Darpa Speech Understanding Project (Klatt, 1977), to try to use a syntactic parser to take advantage of contextual constraints. Unfortunately, there has not been very much success. If I tell you that the next word is going to be a noun, then I really haven't told you very much. The following example illustrates the problem.

In the Optical Character Recognition (OCR) application, it is likely that the words *form* and *farm* might be confused by the channel model. Imagine, for example, that they were found in one of the following two contexts:

$$\textit{federal} \left[\begin{array}{c} \textit{farm} \\ \textit{form} \end{array} \right] \textit{credit}$$

$$\textit{some} \left[\begin{array}{c} \textit{farm} \\ \textit{form} \end{array} \right] \textit{of}$$

Most people would have little trouble deciding that *farm* is much more likely in the first context and that *form* is much more likely in the second context. In fact, trigram models also have little difficulty with this example. However, a syntactic parser wouldn't help very much. The parser might tell us that the missing word is a noun, but that wouldn't help distinguish between *form* and *farm* because they are both nouns. In general, if one were to compare the relative importance of local context versus long-distance dependencies, one would almost certainly find that the local context is much more important, at least in terms of predicting the next word.

The linguistic notion of syntax (constraints on nouns, verbs, subjects, objects, phrases, etc.) was not intended to be used in a noisy channel model. Chomsky has always been more interested in linguistic *competence* (an idealization of syntax) than *performance* (deviations that are found in the real world including: word frequencies, word association norms, collocations, statistical preferences, memory and computational limitations, etc). It should not be surprising that performance issues are important in recognition applications, and consequently, models that are based too closely on idealized notions of syntactic competence are likely to run into trouble when they are tested on real data.

7. Entropy

It is common practice to evaluate a language model on the basis of its entropy. The standard ascii code uses 8 bits to represent a character. Obviously, many of these bits are unnecessary since some letters are much more common than others. If one were to take advantage of letter frequencies using a Huffman code to encode each letter one at a time, then it would take about 5 bits to code each character. This very simple code does almost as well as the Unix(TM) *compress* program, which uses the Lempel-Ziv algorithm (Welch, 1984).

In general, models based on words achieve much better compression than models based on characters. A unigram model (a Huffman code based on word probabilities) requires about 2.1 bits per character (Brown, personal communication). Note that the unigram model out-performs Lempel-Ziv by a considerable margin, indicating that the standard Unix(TM) compress program could be improved significantly.

The trigram model achieves even better compression, 1.76 bits per character (Brown *et al.*, 1991). This last model is remarkably close to Shannon's estimate for the entropy of English. However, it isn't exactly fair to compare these estimates since Shannon's estimate was based on a 27 character alphabet whereas these other estimates are based on a 256 character alphabet. Nevertheless there does seem to be some reason to believe that the trigram model is doing quite well, and that it might be almost as good as native speakers in predicting the next letter.

Table 5: Entropy of Various Language Models

Model	Bits / char
Ascii	8
Huffman code each char	5
Lempel-Ziv (Unix(TM) compress)	4.43
Unigram	2.1
Trigram	1.76
Shannon's Estimate	1.25

8. Sparse Data ''Fixes''

As mentioned above, the sparse data problem is probably the most serious weakness with the trigram model. In fact, there are usually many more parameters than data points. Let V be the number of types in the vocabulary and N be the number of tokens in the corpus. Then there are V^3 parameters, which is generally much much larger than N, the size of the training set. For example, in the Brown Corpus, there are $V^3 \approx 1.25 \times 10^{14}$ trigrams, and only $N \approx 10^6$ tokens to train from. Obviously, most of the possible trigrams will not be observed in the training corpus.

One might think that one could fix the sparse data problem by collecting more data, but ironically V^3 generally grows much faster than N. That is, if you collect a larger corpus (more tokens), then you will also find more types (vocabulary items). It isn't exactly clear how these two function grow, but I believe that the vocabulary grows almost linearly with corpus size. In any case, V^3 grows much much faster than N, so collecting more data is not a solution to the sparse data problem.

Something has to be done about the sparse data. Katz (1987) suggests ''backing-off'' from the trigram estimates when there isn't enough data. Basically, the idea is to replace trigram estimates with a combination of unigram, bigram and trigram estimates. This is obviously a good idea.

One can also try to reduce the number of parameters by grouping words into classes (e.g., parts of speech, synonym sets, etc.) Brown *et al.* (1990b) suggest building classes with a self-organizing procedure which joins words based on a mutual information criterion. The criterion has the effect of joining together words that have similar distributions (e.g., days of the week, months of the year, etc). Although this particular suggestion is very intriguing, it probably won't help too much with the sparse data problem because it isn't possible to determine that two words have a similar distribution unless you have a fair number of examples of both words. The real problem is what to do with words that you haven't seen very often in the training set. Worse, what do you do with words that you haven't seen at all. The criterion for joining words cannot depend on data that is unavailable.

9. MLE, ADD1, GT and HO

Finally, one can ''adjust'' frequency counts, especially when they are small. In principle, n-gram probabilities can be estimated from a large sample of text by counting the number of occurrences of each n-gram of interest and dividing by the size of the training sample. This method, which is known as the ''Maximum Likelihood Estimator,'' (MLE) is very simple. However, it is unsuitable because n-grams which do not occur in the training sample are assigned zero probability. This is qualitatively wrong for use as a prior model, because it would never allow the n-gram, while clearly some of the unseen n-grams will occur in other texts. For non-zero frequencies, the MLE is quantitatively wrong.

Three alternatives will be mentioned here. These methods all take the observed counts (r) and produce an adjusted count (r^*). The last two methods also make use of N_r, the number of types that occur exactly r times.

$$r^* = r \qquad \text{MLE}$$

$$r^* = (r + 1) \frac{N}{N + S} \qquad \text{ADD1}$$

$$r^* = (r + 1) \frac{N_{r+1}}{N_r} \qquad \text{GT}$$

$$r^* = C_r / N_r \qquad \text{HO}$$

The first method, ADD1 (Jeffreys, 1948), simply adds one to all of the observed counts and then adjusts the total appropriately by multiplying by $N/(N+S)$ where S is the number of types (e.g., V^3). This method is generally a disaster, especially when S is much larger than N, which is most of the time. In a spelling correction application, Gale and I have found that this method produced very misleading estimates and concluded that estimating the context badly can be worse than not estimating the context at all (Church and Gale, 1990).

The second method, GT (Good, 1953), depends only on the modest assumption that ngrams have binomial distributions. Unfortunately, even this modest assumption turns out to be highly problematic. Words and ngrams are like busses in New York City; they are social animals and like to travel in packs. The word *earthquake*, for example, has a very bursty distribution in the Associated Press (AP) Newswire, depending on whether or not there has recently been an earthquake. The word *turkey* also has a bursty distribution in the AP, with a burst appearing once a year in late November. In fact, one can show that the binomial assumption is often seriously off depending on what happens to be in the news, among other things.

The last method, HO held-out estimate (Jelinek and Mercer, 1985), assumes the least, merely that the training and test corpora are generated by the same process. This method splits the text into two halves and uses the first half to determine N_r, the number of types that occur r times, and the second half to determine their total mass C_r. r^* is then simply set to C_r / N_r. For example, to determine 0*, the adjusted count for ngrams that did not occur in the first half, one would compute C_0, the total count in the second half for ngrams that did not appear in the first half, and divide by N_0, the number of ngram types that did not appear in the first half.

In (Church and Gale, 1991b), we compared the GT and HO methods for estimating bigram frequencies in 22 million words of Associated Press Newswire and found that the GT method was slightly better when the binomial assumption was appropriate. Tables 6 and 7 show that both methods produce remarkably similar estimates for r^*.

Table 6: Good-Turing (GT) Estimate

r	Nr	r*
0	74,671,100,000	0.0000270
1	2,018,046	0.446
2	449,721	1.26
3	188,933	2.24
4	105,668	3.24

Table 7: Held-Out (HO) Estimate

r	Nr	Cr	r*
0	74,671,100,000	2,019,187	0.0000270
1	2,018,046	903,206	.448
2	449,721	564,153	1.25
3	188,933	424,015	2.24
4	105,668	341,099	3.23

The agreement of the two methods, though, is partly due to the fact that we took extraordinary measures to control for the New York City bus effect. That is, we spit the text into two samples by randomly assigning each bigram to one of the two samples. This effectly destroyed any time structure that might have existed in the two samples. If we had split the text into two halves sequently by assigning the first six months of the newswire to the first half and the second six months to the second half, then we would have observed significant differences due to the non-binomial nature of the news.

Table 8 shows that there is considerable agreement when the text is split randomly. The *t*-scores are possibly somewhat larger than we would like, but they are really not too bad considering that we are dealing with extremely infrequent events. The *t*-scores are computed using an estimate of variances which is described in (Church and Gale, 1991b). Table 9 shows that there is considerable disagreement if the texts are split sequentially.

Table 8: Split Text Randomly

r	HO	GT	t
0	.000027041	.000027026	–.7
1	.4476	.4457	–2.9
2	1.254	1.260	2.5
3	2.244	2.237	–1.5
4	3.228	3.236	1.0
5	4.21	4.23	1.8
6	5.23	5.19	–2.8

Table 9: Split Text Sequentially

r	HO	GT	t
0	0.00001684	0.0001132	479.4
1	0.4076	0.5259	113.
2	1.0721	1.2378	47.0
3	1.9742	2.2685	37.8
4	2.8632	3.1868	26.4
5	3.7982	4.2180	25.8
6	4.7822	5.2221	15.4

In summary, there are quite a number of very powerful techniques such as GT and HO for estimating the probability of an n-gram that did not appear very many times in the training corpus, if at all. These methods appear to work remarkably well when the assumptions are met, but unfortunately, there are serious problems with the assumptions. There has recently been some interest in adaptive models, models that can take advantage of recency effects and forgetting effects. If words were binomially distributed, then the probability of a word should be independent of how long it has been since it was last mentioned. In the AP wire, it appears that the probability increases dramatically when a word has been mentioned recently, and drops fairly consistently with the length of time since the last mention.

10. Translation Applications

Section 1 discussed the use of noisy channel methods in recognition applications. This section will show how the same methods can be used to address translation applications such as Machine Translation (MT). The approach was first suggested by Weaver in 1949 and is currently being revived by Brown *et al.* (1990a). If you would like to translate words in a source language, W_s (e.g., French) into words in a target language, W_t (e.g., English), you imagine that the source words W_s were the output of a noisy channel. The translation task is to find the most likely input to the noisy channel given the observed outputs.

$$W_t \rightarrow \mathit{Noisy\ Channel} \rightarrow W_s$$

Viewed in this way, translation is very similar to recognition. In principle, one can recover the most likely input by hypothesizing all possible target language texts, W_t, and selecting the target text with the highest score, where scores are computed by basically the same formula as above:

$$\underset{W_t}{ARGMAX}\ Pr(W_t)\ Pr(W_s | W_t)$$

This information theoretic approach to machine translation is extremely controversial among researchers in machine translation because it questions many of the basic assumptions that have dominated the field since the 1950s when Chomsky (1957) and others pointed out that statistical n-gram methods are incapable of modeling certain syntactic constraints such as agreement over long distances. Brown *et al.* (1990a) argue that the statistical approach is more tractable than it was in the 1950s. Computers are certainly faster than they were then. In addition, and probably much more importantly, it is now possible to find large amounts of *parallel text*, text such as the Canadian parliamentary debates which are available in multiple languages. Brown *et al.* estimate $Pr(W_t)$ and $Pr(W_s | W_t)$ by computing various statistics over these parallel texts. Although the approach may be deeply flawed for many of the reasons that were discussed in the 1950s, there is, nevertheless, a growing community of researchers in corpus-based linguistics such as (Klavans and Tzoukermann, 1990) who are becoming convinced that the approach is worth pursuing because there is a very good chance that it will produce a number of lexical resources that could be of great value to their research.

11. Part of Speech Tagging

This description of the machine translation problem is fairly general and can be applied to quite a number of transduction problems. Consider, part of speech tagging, for example. A part of speech tagger takes an input sequences of words such as *The table is ready*. and outputs a sequence of parts of speech such as: *Article Noun Verb Adjective*. The problem is non-trivial because it is well-known that part of speech depends on context. The word ‘‘table,’’ for example, is usually a noun, but it can also be a verb in some contexts such as: *The chairman will table the motion*.

The tagging problem can be viewed as a translation problem, not unlike machine translation. Imagine that we have a sequence of parts of speech *P* that go into the channel and produce a sequence of words *W*. Our job is to try to determine the hidden parts of speech *P* given the observed words *W*.

$$P \rightarrow \mathit{Noisy\ Channel} \rightarrow W$$

As before, in principle, one can hypothesize all possible inputs to the channel and score them by:

$$\underset{P}{ARGMAX}\ Pr(P)\ Pr(W | P)$$

Again, the parameters in this model are generally estimated by computing various statistics over large bodies of text. Both Church (1988) and DeRose (1988) have used the Tagged Brown Corpus (Francis and Kucera, 1982) for this purpose, which is particularly convenient because it comes with parts of speech that were check by hand. deMarcken (1990) used the Tagged Lancaster/ Oslo-Bergen Corpus (LOB) which also comes with parts of speech. Others such as Jelinek (1985) have used the Baum-Welch Algorithm (Baum, 1972) to estimate the parameters from raw untagged text.

I have always felt that hand-tagged text produces more reliable estimates, and recently Merialdo (1990) performed an experiment which seems to back-up my suspicion. He estimated the parameters using some hand-tagged data and then ran the re-estimation procedure and compared performance before and after re-estimation. One might have thought that re-estimation ought to improve performance, but he found just the opposite. He concludes that one should use as much tagged text as possible to estimate the parameters, and one should resort to re-estimation only when it is not possible to find a sufficient amount of tagged training material.

There are, of course, many other ‘‘translation’’ applications that are very analogous to machine translation and part of speech tagging where one wants to transduce one tape of symbols into another. In speech recognition, for example, it is common to use these noisy-channel methods to translate a sequence of acoustic labels (e.g., the output of a filter bank) into a sequence of phonetic labels (e.g., consonants and vowels).

12. Conclusions

Quite a number of applications have been mentioned in just a few pages: spelling correction, speech recognition, optical character recognition, text compression, machine translation and part of speech tagging. Of course, there are many other applications that should have been discussed, especially information retrieval (Salton, 1989) and author identification (Mosteller and Wallace, 1964), but there just wasn't enough space to say everything.

All of this work points very strongly to the fact that 1950-style empiricism is back in fashion. I have been asked to explain why, and I'm not sure that I have a good answer. Of course, it is possible that the current interest in empiricism is just a fad that will soon fade away. But, I would like to believe that there are good reasons for the revival. One can point to huge advances in computational power since the 1950s. But, even more importantly, the electronic culture has now permeated the publishing sector to such an extent that it is no longer difficult to find hundreds of millions of words of text in electronic form. And there is promise of billions of words in the very near future. The availability of data on such a massive scale has made it possible to carry out experiments that just weren't possible back in the 1950s. Indeed, many of the experiments discussed in this paper would not have been possible without the availability of very large corpora.

Kenneth Church

References

Baum, L. (1972) ‘‘An Inequality and Associated Maximization Technique in Statistical Estimation of Probabilistic Functions of a Markov Process,’’ *Inequalities*, vol. 3, pp. 1-8, 1972.

Brown, P., Cocke, J., Della Pietra, S., Della Pietra, V., Jelinek, F., Lafferty, J., Mercer, R., Rossin, P. (1990a), ‘‘A Statistical Approach to Machine Translation,’’ *Computational Linguistics*.

Brown, P., Della Pietra, V., deSouza, P., Lai J., Mercer, R. (1990b) ‘‘Class-based N-gram Models of Natural Language,’’ unpublished ms., IBM.

Brown, P., Della Pietra, S., Della Pietra, V., Lai J., Mercer, R. (1991) ‘‘An Estimate of an Upper Bound for the Entropy of English,’’ submitted to *Computational Linguistics*.

Chomsky, N. (1957) *Syntactic Structures*, The Hague: Mouton & Co.

Church, K. (1988) ‘‘A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text,’’ Second Conference on Applied Natural Language Processing, Association for Computational Linguistics, Austin, Texas.

Church, K., and Hanks, P. (1990) ‘Word Association Norms, Mutual Information, and Lexicography,’ *Computational Linguistics*, 16:1.

Church, K., and Gale, W. (1990) ‘‘Poor Estimates of Context are Worse than None,’’ Darpa Workshop, Hidden Valley, PA.

Church, K., and Gale, W. (1991a) ‘‘Probability Scoring for Spelling Correction,’’ *Statistics and Computing*.

Church, K., and Gale, W. (1991b) ‘‘A Comparison of the Enhanced Good-Turing and Deleted Estimation Methods for Estimating Probabilities of English Bigrams,’’ *Computer Speech and Language*, vol. 5, pp. 19-54.

DeRose, S. (1988) ‘‘Grammatical Category Disambiguation by Statistical Optimization,’’ *Computational Linguistics*, Vol. 14, No. 1.

deMarcken, C. (1990) ‘‘Parsing the LOB Corpus,’’ Association for Computational Linguistics.

Firth, J. (1957) ‘‘A Synopsis of Linguistic Theory 1930-1955’’ in *Studies in Linguistic Analysis*, Philological Society, Oxford; reprinted in Palmer, F. (ed. 1968) *Selected Papers of J. R. Firth*, Longman, Harlow.

Francis, W., and Kucera, H. (1982) *Frequency Analysis of English Usage*, Houghton Mifflin Company, Boston.

Good, I., (1953), ‘‘The population frequencies of species and the estimation of population parameters,’’ *Biometrika*, v. 40, pp. 237-264.

Halliday, M. (1966) ‘‘Lexis as a Linguistic Level,’’ in Bazell, C., Catford, J., Halliday, M., and Robins, R. (eds.), *In Memory of J. R. Firth*, Longman, London.

Jeffreys, H., (1948) *Theory of Probability*, second edition, section 3.23, Oxford: Clarendon Press.

Jelinek, F. (1985) ‘‘Self-organized Language Modeling for Speech Recognition,’’ IBM Report, also available in, Waibel, A. and Lee, K. (eds.) (1990) *Readings in Speech Recognition*, Morgan Kaufmann Publishers, San Mateo, California.

Jelinek, F., and Mercer, R. (1985) ‘‘Probability distribution estimation from sparse data,’’ *IBM Technical Disclosure Bulletin*, v. 28, pp. 2591-2594.

Katz, S. M., (1987), ‘‘Estimation of probabilities from sparse data for the language model component of a speech recognizer,’’ *IEEE Transactions on Acoustics, Speech, and Signal Processing*, v. ASSP-35, pp. 400-401.

Kernighan, M., Church, K., Gale, W. (1990) ‘‘A Spelling Correction Program Based on a Noisy Channel Model,’’ Coling, Helsinki, Finland (proceedings are available from the Association for Computational Linguistics).

Klavans, J., and E. Tzoukermann (1990) ‘‘The BICORD System,’’ Coling, Helsinki, Finland (proceedings are available from the Association for Computational Linguistics).

Klatt, D. (1977) ‘‘Review of the ARPA Speech Understanding Project,’’ *Journal of the Acoustical Society of America*, reprinted in Waibel, A. and Lee, K. (eds.) (1990) *Readings in Speech Recognition*, Morgan Kaufmann Publishers, Inc., San Mateo, California.

Mays M., F. Damerau and R. Mercer (1990) ‘‘Context Based Spelling Correction,’’ IBM internal memo, RC 15803 (# 730266).

Merialdo, B. (1990) ‘‘Tagging Text with a Probabilistic Model,’’ in *Proceedings of the IBM Natural Language ITL*, Paris, France, pp. 161-172.

Mosteller, F. and Wallace, D. (1964) *Inference & Disputed Authorship: The Federalist*, Addison-Wesley, Reading, Massachusetts.

Meyer, D., Schvaneveldt, R., and Ruddy, M., (1975), ‘‘Loci of Contextual Effects on Visual Word-Recognition,’’ in Rabbitt, P., and Dornic, S., (eds.), *Attention and Performance V*, Academic Press, London, New York, San Francisco.

Palermo, D., and Jenkins, J., (1964) ‘‘Word Association Norms,’’ University of Minnesota Press, Minneapolis.

Salton, G. (1989) *Automatic Text Processing*, Addison-Wesley Publishing Co.

Shannon, C. (1948) ‘‘The Mathematical Theory of Communication,’’ *Bell System Technical Journal*.

Shannon, C. (1951) ‘‘Prediction and Entropy of Printed English,’’ *Bell Systems Technical Journal*, vol. 30, pp. 50-64.

Sinclair, J., Hanks, P., Fox, G., Moon, R., Stock, P. (eds), (1987a), *Collins Cobuild English Language Dictionary*, Collins, London and Glasgow.

Sinclair, J., (1987b) (ed.), ‘‘Looking Up: an account of the COBUILD Project in lexical computing,’’ Collins, London and Glasgow.

Welch, T. (1984) ‘‘A Technique for High Performance Data Compression,’’ *IEEE Computer*, vol. 17, no. 6, pp. 8-19.

Weaver, W. (1949) ‘‘Translation,’’ reproduced in Locke, W. and Booth, A., (1955) (eds.), *Machine Translation of Languages*, MIT Press, Cambridge, Mass.