Language Processing, Statistical Methods

Chris Brew

February 25, 2005

Abstract

Statistical methods for language processing are common in language technology but less widespread in other fields of linguistics. The article is an introduction to some of the relevant techniques and their application to the study of language.

Keywords: statistics, probability, natural language processing (NLP)

1 Introduction

It is not a new idea to analyse language by counting things and uncovering statistical regularities in the results. Ancient applications include code-breaking and literary analysis, while the late nineteenth and early twentieth century brought statistical approaches to language-related design problems in telegraphy and stenography. Researchers in the fields of computational linguistics, language technology and natural language processing can expect to encounter statistical methods and probabilistic thinking on a daily basis. The key enabling factors for natural language processing were the arrival of the digital computer and the availability of automatic data processing. There is nevertheless a body of striking empirical work that predates the ready availability of computers, including Mosteller and Wallace [1984] on authorship attribution in the Federalist Papers and Kučera and Francis [1967], which pioneered the design and creation of large-scale data resources for language analysis.

The focus of this article is on the statistical methods which have been developed for processing language by computer, but it is worth beginning with a detour that addresses two well-known objections. One is Jonathan Swift’s satirical account of stochastic text generation in the academy at Lagado.

Swift’s intent is to present as self-evidently absurd the notion that anything of literary value could be produced by such a process. This is satire, and makes no scientific claim, but language technologists may nevertheless find more of scientific interest than Swift would have expected. Exactly how are the “Moods,
Tenses and Declensions” encoded? Why does the machine have 40 iron handles rather than just one? Are the words that are produced always drawn from the same probability distribution or does the machine have a number of distinct internal states, so that sometimes one word is more likely, sometimes another? In spite of the satirical intent, interesting scientific issues are raised.

A second objection is made explicit in Noam Chomsky’s discussion of statistics, which is more measured and subtle than Swift’s satire.

... 

(1) Colorless green ideas sleep furiously
(2) Furiously sleep green ideas colorless

... 

It is fair to assume that neither sentence (1) nor sentence (2) (nor indeed any part of these sentences) has ever occurred in English discourse. Hence, in any statistical model for grammaticalness, these sentences will be ruled out on identical grounds. Chomsky [1957] p. 16

Chomsky’s goal in making this argument was to contrast the statistical approach with his own preferred methodology, which relies primarily on introspection about the grammatical acceptability of sentences found or imagined by the linguist.

Chomsky presumably reached his conclusion because he was operating with a rather narrower notion of “possible statistical model” than do most statistical linguists. For example, it is perfectly possible to construct a statistical model that sees not individual words but broad word classes. Simplifying somewhat, such a model might assign probabilities to (1) and (2) by treating them as if they were instead (3) and (4) respectively.

(3) Adjective Adjective Noun Verb Adverb
(4) Adverb Verb Noun Adjective Adjective

Under this scenario it is very much to be expected that 4 will receive a lower probability than 3, because it is so rare to see pairs of adjectives occurring at the end of a sentence. Chomsky is not wholly averse to statistics, and explicitly acknowledges that statistical methods can and should be used to model the use of language, but he does deny their usefulness as tools for explicating syntactic structure. The cited passage, and similar passages elsewhere in Chomsky’s work are a large part of the reason why most theoretical linguists have tended to avoid statistics. In spite of the scientific intent, much of the impact of Chomsky’s discussion comes from the satirical bite of its rhetoric, rather than the technical correctness of the arguments presented.

More detail on the relationship between statistics and linguistics can be found in Abney [1996], Pereira [2000], Bod et al. [2004]

For purposes of language technology, it is moot whether theoretical linguists accept statistical methods, since it is very clear that these methods can contribute to the creation of useful and instructive linguistic artefacts that carry out such tasks as part-of-speech assignment and word-sense disambiguation. The balance of this article gives an introduction to some of the concepts and techniques used.

2 Probability

Probability is the standard means for rational reasoning about uncertainty. In probability theory a key concept is the event.

Examples of events include these:

• A coin is tossed and comes down heads.
• The result of the spin of a roulette wheel is 19.
• Two dice are rolled, and the sum of the results is 7.

• Two contiguous words are selected from Conan Doyle’s ‘A Case of Identity’ and they turn out to be 
  Sherlock and Holmes respectively.

We can describe the situations above in terms of an event \( x \) and its probability \( P(x) \).

• \( x \): A coin is tossed and comes down heads.
  \( P(x) = 0.5 \) for a fair coin that comes down heads half the time.

• \( x \): The result of the spin of a roulette wheel is 19.
  \( P(x) = 1/38 \) for a fair roulette wheel with 38 slots.

• \( x \): Two dice are rolled, and the sum of the results is 7.
  \( P(x) = 6/36 = 1/6 \) for fair dice.

• \( x \): Two contiguous words are selected from Conan Doyle’s ‘A Case of Identity’ and they turn out to be 
  Sherlock and Holmes respectively.
  \( P(x) = 7/7051 \) because there are 7051 pairs of two contiguous words in the story, and exactly 7 of 
  them are Sherlock Holmes.

These probabilities are obtained either by counting up possibilities (for the coin or the dice) or by 
counting occurrences of events (as in the Conan Doyle example). By definition a fair coin is one that makes 
the two possible outcomes equally likely. A fair die is one that gives a probability of 1/6 to each of the six 
possible outcomes. Therefore there are 36 equally likely outcomes to a roll of two dice, of which exactly 
six \((1\text{and}6; 2\text{and}5; 3\text{and}4; 4\text{and}3; 5\text{and}2\text{and}6\text{and}1)\) add up to the required number. The probability calculus 
supports this kind of reasoning by explaining how the probability of the compound event of rolling 7 is 
related to those of the simpler components of that event. Just as logic is a method for reasoning about the 
truth or falsity of propositions, so the probability calculus is a method for reasoning about uncertainty. The 
rules of logical inference remain the same regardless of the nature of the premises. In the same way, the rules 
of probabilistic inference remain the same regardless of the origin of the initial probability estimates.

2.1 Random Variables and Bayes’s Rule

We have already presented an informal argument that links the probability of the event ‘two dice are thrown 
and the total is 7’ to the probabilities of the different possible outcomes of rolling the dice. A more formal 
presentation would be the following. Define two random variables \( R, G \) (corresponding to a red and green 
die respectively). Both \( R, G \) can take on any of the values 1, 2, 3, 4, 5, 6. We notate the probability of rolling 
a one with the green die as \( P(1) \) and the probability of rolling a three with the green die and a four with 
the red die as \( P(3,4) \). The latter is referred to as a joint probability because it consists of two component 
events.

Under the assumption that the two random variables involved in a joint event are independent, the 
probability of the joint event is obtained by multiplying the probability of the components. Thus

\[
P(3,4) = P(3)P(4)
\]

In assuming that the dice are fair, we are stipulating that all of the \( 6 \times 6 = 36 \) outcomes are equally likely. 
Since there are exactly 6 ways in which 7 can be rolled, it follows that \( P(7) = 1/6 \).

Thus far we have dealt with joint probabilities and their simple components, but many of the most 
important probabilities in natural language processing are conditional. A conditional probability is one 
where attention is focussed on a particular subset of the space of possible events. In the Conan Doyle 
story, as in general English, it turns out that some words are extremely predictable from preceding context. 
The conditional probability focuses only on those cases where the first word is Sherlock and asks about the
proportion of these cases where the next is *Holmes*. In fact, every time that the word *Sherlock* appears the next word is *Holmes*. We notate this as:

\[ P(\text{Holmes}_{n+1} | \text{Sherlock}_n) = 1 \]

using \(\text{Sherlock}_n\) to denote the event that the \(n\)th word is *Sherlock* and \(\text{Holmes}_{n+1}\) to denote the event that the \(n + 1\)th word is *Holmes*. The vertical bar signals the conditioning of the probability, and is read as “given”. Notice in table 1 that \(P(\text{Sherlock}_n | \text{Holmes}_{n+1})\) is not the same thing. *Sherlock* occurs 7 times only, always followed by *Holmes*, whereas *Holmes* occurs 46 times, as shown in the table. Therefore \(P(\text{Sherlock}_n | \text{Holmes}_{n+1}) = 7/46\).

An important property of conditional probability is that it participates in the *chain rule*, which states that

\[ P(A, B) = P(A) \times P(B|A) = P(B) \times P(A|B) \]

For the running example:

\[ P(\text{Holmes}_{n+1}, \text{Sherlock}_n) = P(\text{Sherlock}_n) \times P(\text{Holmes}_{n+1} | \text{Sherlock}_n) \]

\[ = P(\text{Holmes}_{n+1}) \times P(\text{Sherlock}_n | \text{Holmes}_{n+1}) \]

Given that \(P(\text{Holmes}) = 46/7051\) and \(P(\text{Sherlock}) = 7/7051\), the reader will be able to verify that the two apparently different decompositions of the joint event lead to \(P(\text{Sherlock}, \text{Holmes}) = 7/7051\).

This equivalence is of more than academic interest in natural language processing, because it establishes a relationship between \(P(A|B)\) and \(P(B|A)\).

\[ P(B|A) = \frac{P(B) \times P(A|B)}{P(A)} \]

This relationship is known as *Bayes’s Rule*. In natural language processing it is very common for \(A\) to be an *observation* such as a word, whereas \(B\) is a *hidden variable*, such as the part-of-speech tag for that word. The usefulness of the rule for natural language processing comes from the following facts:

- In many applications it suffices to find the value of \(B\) that maximizes \(P(B|A)\), and the probability itself is just a means to this end. For purposes of maximization we can simply drop \(P(A)\) because it is a constant. When determining a part-of-speech given a word we do not care how probable the word is.

- It is often easier to collect the data and justify the assumptions that would be needed to form a sensible estimate of \(P(B)\) and \(P(A|B)\) than it would be for \(P(B|A)\).

In beginning to talk of estimates, we are moving from probability to statistics, which is the topic of the next subsection.
2.2 Statistics

Imagine repeatedly tossing a coin, keeping track of heads and tails. It is reasonable to suppose that the successive trials are unaffected by the outcome of previous trials. It is also reasonable to expect that over time an unbiased coin will give approximately equal numbers of heads and tails. It is also true that for a biased coin the ratio between heads and tails will eventually settle down to some stable value that depends on how much material has been shaved off the edge of the coin by the person who biased it. In other words, given enough data we stand a good chance of obtaining a stable estimate the true underlying value of the coin’s bias. The same is true of language: if we see a large enough training corpus we will be in a position to estimate the underlying parameters of various different statistical models that we might assume to have generated that corpus.

For the Conan Doyle story, sensible probabilities can be obtained by counting the number of times the relevant words occur in sequence. Although this process gives complete information about the specifics of ‘A Case of Identity’, corpus researchers are typically interested in generalizing from the specific to more general hypotheses. They may for example wish to make claims about the patterns of word occurrence in all Sherlock Holmes stories, in the whole of Conan Doyle, in Victorian and Edwardian literature, or in written English. Such claims rely not only on correct use of probability but also on scientific judgments about the extent to which small samples from some process can be regarded as representative of the process as a whole. The mathematics and science behind these judgments are at the heart of the field of statistics. Taken together, probability and statistics provide recipes for collecting data that one hopes will be representative, reducing that data to informative summaries and reasoning about the results. A good introduction to probability and statistics is DeGroot [1986]

2.2.1 Information theory

Information theory, the mathematics of telegraphy, is central to statistical language modeling. It makes the idea of quantity of information precise. This enables meaningful comparisons between different natural language processing systems.

Imagine watching a sequence of symbols go past on a ticker-tape. You have seen the symbols \( w_1 \ldots w_{i-1} \) and you are waiting for \( w_i \) to arrive. You ask yourself the following question:

How much information will I gain when I see \( w_i \)?

Another way to express the same thing is:

How predictable is \( w_i \) from its context?

Each time we see a symbol we are more surprised or less surprised, depending on which symbol turns up. Large information gain goes with extreme surprise. If you can reliably predict the next symbol from context, you will not be surprised, and the information gain will be low. The average information gain is called entropy. The entropy will be lowest when you can usually predict the next symbol with good reliability, and highest when you often cannot. The amount of information gained when we see a \( w \) that has probability \( P(w) \) turns out to be the logarithm of the probability (notated \( \log P(w) \)). If logarithms are taken to base 2, information is denominated in bits, but if natural logarithms are used it is denominated in nats.

A good statistical model of language is one which provides reliable predictions. It therefore tends to minimize entropy. The model that we have just described is a word-level language model. Such a model would be useful to a speech recognition system that wishes to constrain its search for the next word by focusing attention on those that are likely to follow the words it has already recognized.

2.2.2 Cross entropy

In the previous section we developed the idea that entropy is measure of the average information gain from seeing the next symbol of a ticker-tape. Now we imagine that we are still watching a ticker tape, whose behaviour is still controlled by some probability distribution, but that we have imperfect knowledge of the
true probabilities. Call the true probability model $P$ and our imperfect knowledge of it $P_M$. When we see $w$ we assess our information gain as $\log P_M(w)$, not as the correct $\log P(w)$. The average information gain that we assess in this situation is called the cross-entropy of the signal. It is defined in terms of the imperfect model $P_M$ as well as the true model $P$. It is a remarkable and important fact that the cross-entropy with respect to any incorrect probabilistic model is greater than the entropy with respect to the correct model.

The reason that this fact is important is that it provides us with a justification for using cross-entropy as a tool for evaluating models. The search for a good model can be organized in the following way:

- Initialize your model with random (or nearly random) parameters.
- Measure the cross-entropy.
- Alter the model slightly (maybe improve it).
- Measure again, accepting the new model if the cross-entropy has improved.
- Repeatedly alter the model until it is good enough.

If we can guarantee that the alterations to the model will improve cross-entropy, so much the better, but even if some of the changes worsen the cross-entropy the search may eventually yield good models.

3 Statistical modelling

Linguistic phenomena are complex and involve much fine detail. In principle, given sufficient data, it would be possible to build accurate statistical models of such phenomena. Unfortunately, we never have sufficient data, a difficulty which is referred to as the sparse data problem. It is counter-intuitive but true that even millions of words of training data can be insufficient. Therefore we are driven to compromise, choosing to build relatively simple models that capture some but not all of the detail. The advantage of such models is that they can be trained on feasibly available data, the corresponding disadvantage being that they miss some of the potentially useful and interesting regularities that we would like to find.

Engineers and statisticians have a standard approach to this problem. The mathematics of this approach is straightforward, but its application to any particular modeling problem requires considerable insight and good judgment if it is to lead to effective solutions.

The easy part is the following: we begin by describing the situation that we wish to model as a collection of random variables. In the case of a word-level language model we might note that $i$th word $w_i$ can often be predicted from its predecessors $w_1 \ldots w_{i-1}$. Using the chain rule we obtain the following exact expression for the probability of a string as a product of simpler terms

$$P(w_1 \ldots w_n) = P(w_1) \times P(w_2|w_1) \times P(w_3|w_1, w_2) \times \ldots \times P(w_n|w_1 \ldots w_{n-1})$$

or, using product notation

$$P(w_1 \ldots w_n) = P(w_1) \prod_{i=2}^{n} P(w_i|w_1 \ldots w_{i-1})$$

Unfortunately, while the first few terms of this series are simple, the later terms involve probabilities that are conditioned on large amounts of prior context. If a sentence is 20 words long, the prediction of the 20th word will be conditioned on the preceding 19 words. It is very unlikely that this precise sequence of words will have occurred in the training data, so the model will have no basis for making a prediction. Even if the sequence has occurred, it is unlikely to have occurred often enough and with broad enough a range of different continuations for the predictions to be reliable. For example, if the sequence of 19 words has been seen exactly once in the training data, the model will have seen only one word after it, and will predict that word with absolute certainty, excluding all others. Clearly the model has not seen enough data to warrant such certainty.
The general prescription for dealing with such situations is to adopt *independence assumptions*. This is done by ignoring some of the dependencies between the conditioning context and the variable whose value we are attempting to predict. Of course, we wish to keep the most important dependencies and drop the unimportant ones. For a word-level language model it is plausible that the most important influence comes from the immediately preceding word, so we can adjust the model to neglect all dependencies that occur at a longer range than this. That is, we write:

\[
P(w_1 \ldots w_n) \approx P(w_1) \times P(w_2|w_1) \times P(w_3|w_2) \cdots \times P(w_n|w_{n-1})
\]

or, using product notation:

\[
P(w_1 \ldots w_n) \approx P(w_1) \prod_{i=2}^{n} P(w_i|w_{i-1})
\]

This greatly reduces the amount of data that must be stored. Where we previously needed to keep track of the successors of every contiguous subsequence of the training data, we can now get away with recording only the statistics about single words and their possible successors. This alone makes the difference between feasibility and infeasibility. In addition, the chances of having enough training data to collect reliable statistics are greatly increased. The general strategy of simplifying probabilistic models by dropping the less important conditioning information is frequently very effective, but judgment is needed in order to decide how to identify that less important information and how to formulate a model that strikes the appropriate balance between feasibility and detailed predictive power. For the word-level language model it turns out that the model outlined above (termed the *bigram* model, because it needs only statistics over two-word sequences) is an effective compromise. A trigram model (one that uses three-word sequences) is even better, and is widely used in speech recognition.

As we saw in the previous paragraph, restrictive independence assumptions can reduce the impact of the sparse data problem by reducing the number of bins into which the limited amount of training data needs to be sorted. However, it can still happen that the finiteness of the training data cause problems. Specifically, in the case of the word-level language model, when we attempt to use it to predict new sentences, it may encounter a bigram that the model has never seen. Unadjusted, the model would give any sentence containing such a bigram a probability of zero, regardless of what else is going on in the sentence. This is usually not what is required: one wants to be able to make some kind of discrimination between such sentences, so realistic models involve *smoothing*. The idea is to slightly *discount* the evidence from the training corpus, in order that unexpected events can be given some minimal probability. One simple way to do this is to form a smoothed model by averaging together the bigram language model with an extremely non-committal model, such as the one that says that all words are equally likely to occur. This latter is the *uniform model*. Since one does trust the bigram language model to some extent, and one wants the result to be informative, one will give the bigram model greater weight in the averaging process than the uniform model. Although sentences with zero-frequency bigrams will still get low probabilities, these will not be zero. Other, more complex, smoothing schemes are described in chapter 6 of Manning and Schütze [1999] Another approach to the sparse data problem is to use a model of language based on parts-of-speech, as suggested in the discussion of Chomsky above. Once again, an option is to average together the more robust part-of-speech language model with the more delicate word-based language model to yield a combined model that is both reasonably robust and reasonably delicate.

## 4 Machine Learning for natural language processing

Given the availability of large-scale language data, it becomes possible to apply standard algorithms from artificial intelligence and data mining. The most relevant sub-field of artificial intelligence is called machine learning Since this activity is now so widespread in natural language processing it is useful to review a few standard concepts from the machine learning literature that are taken as read in the application of machine learning to natural language processing.
Machine learning algorithms come in two main flavours. The first type is supervised. These algorithms learn from pairs of a predictive context $x$ and a known correct answer $y$. For example, the algorithm designed to disambiguate the senses of words might be shown a large number of contexts in which the word *line* occurs with the different senses listed below, which are selected from the WordNet electronic database [Fellbaum, 1998].

1. line – (a mark that is long relative to its width)
2. line – (text consisting of a row of words written across a page or computer)
3. line – (a conductor for transmitting electrical or optical signals or electric power)

It is likely that the dominant sense in geometry textbooks is the first, that the second sense is frequent in editorial instructions to authors, and that the third is well-represented in telephone repair manuals. In any event, the supervised learning paradigm assumes that some expert has marked up instances of *line* with the appropriate sense. The business of the supervised learner is to learn the relationship between visible clues in the surrounding text and the correct answers. Methods for doing this are referred to as classifiers. Many classifiers are used in natural language processing, including the Naive Bayes classifier, which often performs well in spite of making extremely bold independence assumptions, and decision-tree classifiers, which work by designing a structured series of questions to ask about clues that might or might not be present in the predictive context. For example, a decision tree trying to disambiguate *line* might begin by asking whether the word *angle* appears in the predictive context. If the answer is *yes*, it might conclude that we are probably dealing with the first sense. But if not it might proceed to ask whether the word *transmission* is present, and so on. Algorithms exist for constructing decision trees directly from annotated corpora, without recourse to human intervention.

The second type of machine-learning algorithm used in natural language processing is the unsupervised algorithm. In this setting we do not assume that correct answers are available, and instead try to solve the problem of grouping the predictive contexts $x$ into coherent groups. Methods for doing this are usually referred to as clustering algorithms. An effective clustering algorithm might group together all the samples collected from geometry textbooks, but further processing would be needed in order to recognize that the instances of *line* in these samples are predominantly of the first of the WordNet senses listed. Such algorithms are useful when we are working with unannotated corpora, but the groups that they form are sometimes difficult to interpret and criteria for success are not as obvious as in the case of supervised algorithms. Manning and Schütze [1999] cover both supervised and unsupervised word-sense disambiguation in chapter 8 of their textbook.

5 Modeling of sequences and structures

Our discussion of word-level language models has touched on one of the key distinguishing features of natural language processing as an application area for machine learning. The crucial aspect of these models was that the desired output was not a single word but a sequence of interrelated words. This section describes statistical approaches problems that have this character: the assignment of parts-of-speech and the assignment of syntactic structure. The discussion is necessarily brief, but is intended to indicate the nature of the particular challenge that is posed by the task of predicting structured outputs.

The first structured prediction task we consider is part-of-speech tagging (see PART OF SPEECH TAGGING). In this task the goal is to assign part-of-speech labels to words of a running text. The basis for one solution to this problem is the Hidden Markov Model (HMM). This is an enrichment of the word-based language model we saw earlier. In this model the sequence of observable words is thought of as associated with a parallel sequence of unobserved parts-of-speech. Our task, since we see only the words, is to recover a best guess at the likely sequence of parts-of-speech. HMMs can be dramatized via a well-known urns and balls metaphor. In the urn and ball model, we assume that there are $N$ large urns in a room. Each urn contains a number of coloured balls. There are $M$ different colours, and the relative proportion of each
colour may differ from urn to urn. A genie randomly jumps from urn to urn, randomly choosing a ball from each of the urns that it visits. It shouts out the colour of the ball, then replaces it in the urn. The observer hears the shouts but cannot see where the genie is, so does not know which urn the genie is currently in. If one knows the contents of the urns there are good algorithms that allow a listener aware only of the genie’s shouts to form a rational best guess about the sequence of urns that the genie is most likely to have visited.

This metaphor can be adapted to the part-of-speech assignment problem. Each urn is identified with a single part-of-speech. There may be one urn for nouns, another for verbs, another for prepositions. Instead of colored balls, the urns contain words. Among the contents of the urn for verbs will be words like devour, react and solve, which are unambiguously verbs. They will be present in the urn in proportion to the frequency with which they occur in the language. Similarly, unambiguous nouns such as door and lining will be present only in the noun urn. A listener who hears any of the aforementioned words can be sure from which urn the genie drew that word.

But words with part-of-speech ambiguities afford no such certainty. If the genie shouts out line, which is ambiguous between noun and verb, the best that the listener can do is to take account of the conditional probabilities \( P(\text{line}|\text{noun}) \) and \( P(\text{line}|\text{verb}) \) to form a best guess. These probabilities are called \textit{emission} probabilities and are relevant to the competition between noun and verb as ways for explaining the presence of the word line.

With luck the listener may also have noticed that the previous word was the and that the urn for articles is very far from the urn for verbs, but quite close to the urn for nouns, making it unlikely that the genie will have just jumped all the way from article to verb, but relatively likely that it has jumped from article to noun. This fact about the layout of the room could be encoded in the form of \textit{transition probabilities} \( P(\text{noun}|\text{article}) \) and \( P(\text{verb}|\text{article}) \) that tell us which part-of-speech is likely to follow which. The algorithms that allow a best guess about the sequence of jumps taken by the genie also produce a coherent best guess at the likely sequence of parts-of-speech that go with a sequence of words. Remarkably, it also turns out that statistical models for the part-of-speech assignment problem can be learned in unsupervised fashion from unannotated text, by using a variant of the scheme for reducing cross-entropy that was outlined above.

More details of the training and use of Hidden Markov Models are contained in Huang et al. [1990]. This monograph focuses mainly on speech recognition, but the same algorithms apply to the part-of-speech assignment problem. The treatment of HMMs (sequential structure) has a parallel for probabilistic context-free grammars (tree structure). This cannot be covered here but is dealt with in detail in chapter 12 of Jurafsky and Martin [2000]. The same theoretical possibilities in terms of training from data and recovery of a best guess are available even for these more complex structures.

It is tempting to suppose that Swift’s literary engine was a primitive part-of-speech tagger, that the forty iron handles controlled the choice of appropriate urns, that the ‘Tenses, Moods and Declensions’ were encoded into the parts-of-speech corresponding to the urns and that the chance of producing semi-coherent English text from it was greater than Gulliver was able to appreciate. Literary merit is another matter.

6 Conclusion

Statistical processing of language is a well-established subfield of language science, drawing on linguistics, computer science and engineering. For extended introductions to the field see Manning and Schütze [1999] and Jurafsky and Martin [2000], two textbooks that do a good job of representing the breadth of the field. The increasing availability of language samples in electronic form opens many possibilities for both language technology and theoretical investigation. These samples include carefully curated collections that have been annotated by expert linguists, larger but more lightly curated collections that have been annotated by semi-automatic means and huge bodies of uncurated text culled from the World-Wide Web. It is clear that all these are potentially useful sources of linguistic knowledge, less clear how to exploit these sources to maximum scientific and technological benefit.
References


