Advanced Review

Statistical methods in language processing

Steven Abney*

The term *statistical methods* here refers to a methodology that has been dominant in computational linguistics since about 1990. It is characterized by the use of stochastic models, substantial data sets, machine learning, and rigorous experimental evaluation. The shift to statistical methods in computational linguistics parallels a movement in artificial intelligence more broadly. Statistical methods have so thoroughly permeated computational linguistics that almost all work in the field draws on them in some way. There has, however, been little penetration of the methods into general linguistics. The methods themselves are largely borrowed from machine learning and information theory. We limit attention to that which has direct applicability to language processing, though the methods are quite general and have many nonlinguistic applications.

Not every use of statistics in language processing falls under *statistical methods* as we use the term. Standard hypothesis testing and experimental design, for example, are not covered in this article. *WIREs Cogn Sci* 2011 2 315–322 DOI: 10.1002/wcs.111

INTRODUCTION

History

Statistical methods entered computational linguistics from speech recognition, and machine learning became a second major tributary soon thereafter. Major topics adopted from speech recognition include information theory, Hidden Markov Models (HMMs), and maximum entropy modeling. Machine learning provided a wealth of learning methods, with a particular focus on classification. (In the other direction, computational linguistics has subsequently made contributions to machine learning as well, particularly in the area of semisupervised learning.) From both tributaries came a methodology of experimentation and rigorous evaluation.

Once introduced, statistical methods quickly became dominant in computational linguistics. At the time, natural language processing systems were typically driven by large sets of grammar rules supplemented with preference rules for resolving ambiguities and error-correction rules for handling unexpected inputs, whether erroneous or out of domain. All three sorts of rules were manually constructed, and developers became overwhelmed by interactions among them once systems reached a certain size. Adapting a system to a new domain was nearly as hard as starting over again.

The critical problems, then, were the need for mathematically well-founded ambiguity resolution; the need for learning methods, in particular to adapt systems automatically to new domains; and the need for robustness in the face of noise and incomplete knowledge. Statistical methods addressed all three needs, leading to their rapid adoption.

For convenience, we can organize statistical methods into four areas: (1) the noisy channel model, which we use as a rubric for methods deriving from information theory and coding theory via speech recognition; (2) general machine learning methods; (3) distributional learning methods; and (4) stochastic grammars. We discuss each in turn, but first we touch on some issues common to all.

Corpora and Evaluation

Perhaps the simplest and yet most profound element of the methodology is the use of shared data sets, often consisting of or derived from *corpora* of naturally occurring language. Corpora provide raw material for learning, they allow one to quantify the effectiveness

*Correspondence to: abney@umich.edu

Department of Linguistics, University of Michigan, Ann Arbor, MI, USA

DOI: 10.1002/wcs.111

and generality of processing algorithms, and perhaps most importantly, they enable one to replicate and build on previous results.

The approach is fundamentally experimental. One typically formulates an experiment by defining the objects of interest, or *instances*, and the property that they have that one would like to predict, which constitutes the *label* for the instance. Each instance is represented as a set of *features*. For example, in part of speech tagging, the instances are word occurrences, the label is the correct part of speech in context, and the features may include the word (as a character string), its suffixes, and the preceding and following words.

Labeled data consist of instances that have been manually annotated with the correct label. One typically uses one set of labeled data (the *training set*) to formulate or estimate a model, and a second set (the *test set*) to evaluate the model. In evaluation, one wishes to determine the predictive accuracy of the model on the population as a whole, and the test set is used to obtain an estimate. It is important that the test set be representative of the population, and that it be independent of the training set and any other information used to formulate the model. Test error is an unbiased estimator of true error, but training error generally underestimates true error, because a model generally performs better on the training set used to construct it than it does on the population as a whole.

In formulating a model, one seeks a balance between *fit* to the data (measured by training error) and the *simplicity* of the model. Putting too much emphasis on reducing training error can actually increase test error, because the model becomes adapted to quirks of the training set rather than population patterns; this is known as *overfitting*. The complementary error, putting too much emphasis on simplicity, is *underfitting*.

The basic measure of predictive accuracy is *error*, which is the proportion of instances where the model prediction differs from the human annotation. In some tasks, instances are assigned lists of labels rather than single labels. For example, a task of interest in bioinformatics is the identification of all names of genes occurring in a given sentence. In that case, there are two kinds of error: those of omission and those of commission. Define the *true* labels to be the labels given in the human annotation. The proportion of true labels that are correctly predicted is *sensitivity* (also called *recall*), and the proportion of false candidates that are correctly rejected is *specificity*. An alternative to specificity is *precision*, the proportion of predictions that are correct.

THE NOISY CHANNEL MODEL

Definition of the Noisy Channel

The noisy channel model provides the framework for the dominant approach to speech recognition, and it has found application to many other linguistic problems, including part of speech tagging and machine translation. The model comes from Shannon's original paper on information theory.[1] A message is encoded for transport through a communication channel and decoded at the receiving end. Transmission introduces noise: bits are flipped randomly with a certain probability. The error rate in the decoded message can be reduced by introducing redundancy into the encoding, but as redundancy is introduced, the effective transmission rate drops. It was once thought that zero error rate was achievable only by reducing the effective transmission rate to zero, but Shannon showed that error-free transmission was in fact possible at a positive transmission rate known as the *channel capacity*, which is the number of bits remaining after subtracting the *entropy* introduced by the noise.

The problem of speech recognition is to infer the sequence of words that the speaker had in mind when producing an utterance that arrives at the hearer's ears (or a microphone) as an audio signal. A *language model* defines the probability $p(w)$ of the speaker choosing word sequence w, and an *acoustic model*—representing the properties of the noisy channel—defines the probability $p(s|w)$ that w will reach the hearer in the form of the signal s. Bayes' Rule provides the inverse probability that an observed signal s came from word sequence w:

$$p(w|s) = \frac{p(w)p(s|w)}{\sum_{w'} p(w')p(s|w')}. \quad (1)$$

The goal is to identify the value for w that maximizes (1). Since the denominator is the same for all values of w, it may be ignored, and one seeks simply to maximize the product of the language model probability $p(w)$ and the acoustic model probability $p(s|w)$.

Information-Theoretic Quantities

The negative logarithms of the probabilities $p(w)$ and $p(s|w)$ constitute a measure of information called *entropy*, which corresponds to message length under an optimal encoding. Before the fact, entropy measures the uncertainty regarding the outcome, and after the fact, it measures the amount of information one acquired by learning the outcome. The entropy of the channel noise, for example, is

the amount of information needed to encode it or counteract it. Likewise, there is an intrinsic amount of information in word sequences of English (or any human language). The *entropy of English* has been estimated at no more than 1.75 bits per character of text.[2] This can be interpreted as the difficulty of guessing the next character in a text, knowing all the previous text up to that point. It is equivalent to a choice among $2^{1.75} = 3.36$ equally likely alternatives.

A language model can be viewed as predicting the text a character or word at a time. The difficulty a language model has in guessing the text is called the *cross-entropy* of the language model with the text. It can be shown that cross-entropy is an upper bound for the true entropy of the text. The difference between them is the *divergence* between the language model and the true distribution over word sequences. This provides a measure of quality of a language model: the lower the divergence, the better the model.

Another useful information-theoretic quantity is the *mutual information* between two message sources X and Y. It is the reduction in uncertainty of X, knowing Y. Equivalently, it is the divergence between the distributions $p(x)$ and $p(x|y)$, averaged over values of y.

Markov Models

A simple form of language model is a *Markov chain*, which is a stochastic finite-state machine that generates text as follows. It chooses a state x_0 at random according to an initial-state distribution. Then at each time $t \geq 0$, it outputs a word and chooses a next state x_{t+1} at random according to the *transition probability* $p(x_{t+1}|x_t)$. The output word is uniquely determined by the state. In a *bigram model*, the word and state are identical. In a *trigram model*, a state corresponds to a pair of words (previous and current), with the effect that the choice of output word is conditioned on the previous two words of text.

An *HMM* is a generalization of a Markov chain in which the output is not uniquely determined by the state, but rather by an *emission probability* $p(y_t|x_t)$ defining an output distribution for each state. Oversimplifying greatly, a typical speech recognizer incorporates an HMM whose states correspond to letters (or phones) in the text, and whose outputs are 'codewords' representing spectral features of the signal.

In *recognition*, one is given a sequence $y = (y_1, \ldots, y_n)$ of spectral codewords, and the task is to find the state sequence (letter sequence) $x = (x_1, \ldots, x_n)$ that maximizes $p(x, y)$. The *Viterbi algorithm* is an efficient method for recognition with an HMM. It is a special case of *dynamic programming*.

In *training*, the task is to estimate the model parameters (transition and emission probabilities) from known examples. In *supervised training*, one is given samples of recorded speech y paired with their transcripts x, and in *unsupervised training* one is given only the speech. The *forward–backward algorithm* is an efficient method for unsupervised training of an HMM. It is a special case of the *Expectation-Maximization (EM) algorithm*.[3] The sparse data problem is an important issue in language model estimation, and a large variety of *smoothing* methods have been developed to address it.[4]

Statistical Machine Translation

One of the earliest proposed methods for automatic machine translation was Weaver's idea of treating foreign text as English text that has been encrypted,[5] or that has (in our terms) passed through a noisy channel. That is quite literally the approach taken in statistical machine translation. A stochastic model is assumed for generating a 'source' sentence and 'encoding' it as a sentence in the foreign language, and the translation problem is to reverse the model: given the foreign sentence, determine which sentence was most likely the source.

The general structure is much as in speech recognition. A language model defines $p(e)$, the probability of choosing sentence e as the source; the language model is typically a Markov chain. A *translation model* defines $p(f|e)$, the probability of encoding e as the foreign sentence f. The translation model is analogous to the acoustic model in speech recognition.

One widely used translation model is known as 'IBM Model 3'.[6] After choosing a source sentence e, the generative process chooses a *fertility* for each source word e_i, indicating how many foreign words will be generated from e_i. The fertility is 1, for example, when English *dog* generates French *chien,* but 2 when *breakfast* generates *petit déjeuner*. A fertility is also chosen for a special null word—this makes allowance for inserting foreign words that are not aligned with any source word. Next, the actual foreign words are chosen. In a case like *petit déjeuner,* the model makes two independent choices: *petit* is chosen as one of the words translating *breakfast,* and *déjeuner* is chosen as a second one. Finally, the chosen foreign words are ordered by choosing a position for each, conditioned on the position of the source word.

Each of the abovementioned choices is a stochastic choice. The translation model comprises parameters defining a probability distribution for each choice.

As in speech recognition, the model is used 'in reverse' to translate. The machine translation system is given a foreign sentence, and must reconstruct the most likely source sentence. Unfortunately, no efficient exact algorithm is known—instead, heuristic search is used to find and evaluate candidate source sentences.

MACHINE LEARNING

Parameter estimation for the generative models used in speech recognition and machine translation represents a special case of machine learning. In this section, we turn to the more general theory of learning and its linguistic applications.

Machine learning and natural language processing are two branches of artificial intelligence, and they have become deeply intertwined in recent years. Machine learning techniques are traditionally divided into three categories: supervised learning (classification and regression), unsupervised learning (clustering), and reinforcement learning. A newer fourth category is semisupervised learning. We treat classification and semisupervised learning here, and grammatical inference, a variety of unsupervised learning, arises in the following sections.

Classification

Classifiers are applied to a broad range of tasks in computational linguistics, including sentence segmentation, word-sense disambiguation, text classification, and even parsing. To apply a classifier, a task is represented as assigning *labels* to *instances*, where an instance is represented by its relevant *features*. Features can be reduced to numeric attributes (in the worst case, 0 for absent and 1 for present), so that an instance becomes a *feature vector*, representing a point in *feature space*, and the learner's task is to divide the space into regions associated with labels.

The *nearest neighbor* algorithm is a particularly simple learner: for any given point, it chooses the label of the nearest labeled instance. The regions are neighborhoods surrounding labeled instances. Although algorithmically simple, the nearest neighbor algorithm produces a complex classifier, in the sense that it has a complicated *decision boundary* or dividing line between labeled regions.

Another simple learner is the *Naive Bayes* algorithm. Whereas the nearest neighbor algorithm is *discriminative*, meaning that it tackles the classification task directly, the Naive Bayes algorithm is *generative*. It postulates that labeled instances are generated by stochastically choosing a label, then stochastically choosing a value for each feature given the label. Classification is accomplished derivatively by determining which label was most likely used to generate a given instance whose label is unknown. The Naive Bayes algorithm also contrasts with the nearest neighbor algorithm in that its decision boundary is linear (the higher dimensional generalization of a straight line).

The 'naiveté' of the Naive Bayes algorithm lies in the assumption that features are generated independently, conditioned only on the label. Care is required when using models that make independence assumptions—data that violate the assumptions may cause erratic performance. For that reason, generative methods that do not make independence assumptions, but explicitly model dependencies, are attractive. Such methods include *random fields* and *maximum entropy* models. The maximum entropy criterion is equivalent to maximum likelihood in the cases of interest; what is distinctive about 'maxent' models is the *iterative scaling* algorithms that they use for estimation.[7] The general idea is that the expected frequency of occurrence of a feature, according to the model, should equal its actual frequency of occurrence in a training sample. Iterative scaling methods begin with an initial model, compare its predicted frequencies to the sample frequencies, and adjust the parameters of the model to improve the fit in such a way as to assure that the process ultimately terminates.

We mentioned that the Naive Bayes algorithm produces a linear decision boundary. A number of methods are designed with the explicit aim of finding a good linear decision boundary, including the *perceptron* algorithm,[8] *boosting*,[9] and *support vector machines (SVMs)*.[10,11] A linear boundary implicitly assumes a binary classification task, but this is not a limitation in practice because methods are known for effectively representing multi-class problems as collections of binary tasks: for example, *error-correcting output codes*.[12] The limitation to linear decision boundaries can also be relaxed. A learner that finds a linear boundary can be used to construct a nonlinear classifier by means of the *kernel trick*.[13] The basic idea is that a nonlinear decision boundary, such as a quadratic boundary, can be represented as a linear boundary in a higher dimensional space. Importantly, the higher dimensional representation never needs to be computed explicitly. This idea has been used in computational linguistics to enrich the feature representation without loss of efficiency. In particular, *tree kernels* allow one to use all subtrees of a given tree as features, without needing to explicitly compute (exponentially many) subtrees.[14]

Of the algorithms mentioned, the most widely used in computational linguistics are probably Naive

Bayes and the perceptron, for their algorithmic simplicity, and maxent and SVMs, for their high performance across a wide variety of problems.

Semisupervised Learning

An area of machine learning that has been particularly spurred by problems in computational linguistics is semisupervised learning. In the semisupervised setting, the learner is given a small amount of labeled data (i.e., supervision) and a large amount of unlabeled data. One simple method is *self-training*, in which the labeled data are used to train a classifier, the classifier is applied to the unlabeled data, its high-confidence predictions are added to the labeled data, and the process repeats.[15]

A second method originally developed for language data is *co-training*, which requires two independent *views* of each instance: for example, the contents of a web page and the contents of links pointing to it. One classifier is trained on each view, and each classifier is used to label new instances for the other classifier. Then the classifiers are retrained and the process repeats.[16]

The most advanced semisupervised algorithms are *spectral methods*.[17] If we view the labeled data as providing a few 'fixed points' in the midst of an expanse of unlabeled data, the semisupervised learning task becomes one of interpolating between those fixed points. Spectral methods construct smooth wave-like interpolation functions.

DISTRIBUTIONAL STUDIES

We turn now to methods that are more specialized to language. There is a body of descriptive and unsupervised learning techniques that characterize words by their distributional properties. They come largely from traditions outside machine learning, such as corpus linguistics, information retrieval, and grammatical inference.

Word Distributions

Models of *word distributions* are related to language models. *Zipf's law* states that the frequency of a word is inversely related to its rank frequency: that is, $f(w_n) = Z/n^s$ where w_n is the nth most frequent word and s and Z are fixed constants.[18] This is an example of a *power law*, a kind of heavy-tailed distribution that has recently received attention in the description of random networks, such as are used to model the World Wide Web.[19] Zipf's law is often cited in connection with the sparse data problem: it implies that most things occur only once or twice, making it difficult to estimate their frequency accurately.

Raw word frequency can be deceptive because word distributions are often *bursty*. Models such as *mixtures of Poissons* have been proposed to model burstiness.[20] Function words tend to be much less bursty than content words, so burstiness itself can be useful for inferring syntactic category.

Generally, much of the syntax and semantics of a word can be inferred from its distribution. A word can be represented as a vector of *contextual features*, where the context may be as broad as the document in which the word occurs or as narrow as the words occurring immediately before and after. Clustering context vectors based on their *similarity* yields distributional word classes. Using narrow contexts yields classes that approximate syntactic categories, and using broader contexts tends to produce semantic classes. A common similarity measure is *cosine similarity*, which abstracts away from the gross frequency of words. An alternative information-theoretic measure is *divergence* of context distributions. In addition to methods based on statistics and information theory, the method of *latent semantic indexing* infers word classes by direct algebraic manipulation (specifically, singular value decomposition) of the word-document matrix.[21]

Co-membership in a class is a *paradigmatic relation* between words. Distributional methods are also used to infer *syntagmatic relations*, also called *collocational* or *selectional relations*, being the degree to which two words tend to occur together. Collocational measures compare the actual rate of co-occurrence to what would be expected by chance. There are many ways of doing the comparison. *Mutual information*, for example, is a simple measure provided by information theory; it is the logarithm of the ratio between the actual rate of co-occurrence $p(x, y)$ and the expected rate $p(x)p(y)$ assuming independence, where x and y are two words of interest.

Variations of the general measures of paradigmatic and syntagmatic relatedness have been developed to infer a variety of lexical, syntactic, and semantic information. Examples are technical terminology, semantic ontologies, named entities, phrases with distinctive syntaxes (such as dates, times, money amounts, and the like), subcategorization frames,[22] selectional restrictions of all sorts,[23] and prepositional attachment preferences.[24] Such information has been used in virtually every natural language processing task, including parsing, information extraction, natural language generation, and machine translation. In the most general case, the inference of classes and co-occurrences can be applied to learning complete phrase-structure grammars for a language.[25]

STOCHASTIC GRAMMARS

Stochastic grammars are the most linguistically sophisticated of the probabilistic language models. There was a flurry of work on formal language theory in the 1960s, and it included some work on stochastic grammars; that thread has been picked up and expanded in computational linguistics since the 1990s. Probabilistic regular grammars are represented by Markov chains, discussed above. In *probabilistic context-free grammars,* each rewrite rule has a probability, summing to one among rules with the same left-hand side. Generation becomes a stochastic process, in which the random choices are among the rules expanding a given syntactic category. The generative process associates a probability with each sentence in the language defined by the grammar. *Consistency* becomes an issue: it is possible for some probability mass to be lost to infinite derivations. An effective method exists to test for consistency, and grammars whose probabilities are estimated by relative frequency are guaranteed to be consistent.[26]

Standard parsing algorithms such as the Cocke–Younger–Kasami (CYK) algorithm are easily adapted to probabilistic grammars. Probabilistic CYK parsing can be viewed as a generalization of the Viterbi algorithm; or rather, both are special cases of dynamic programming. For unsupervised grammar learning, there is a generalization of the forward–backward algorithm known as the *inside–outside algorithm*.[27] As a practical matter, however, it yields disappointing results, and supervised learning is almost always used for probabilistic parsers. Since labeled parsing data take the form of a *treebank*—a corpus of sentences annotated with syntax trees—the setting is often called *treebank parsing*. A variety of models have been employed, including decision trees,[28] generative models,[29] and maxent-inspired models.[30] It has become common to employ a model with limited statistical dependencies to produce an initial *n*-best list of parses, and use richer features (such as the tree kernels mentioned earlier) to do *parse reranking*.[31]

Above context-free grammars in the Chomsky hierarchy of grammar complexity are attribute-value grammars. *Stochastic attribute-value grammars* are based on random fields.[32] Probabilistic versions of other context-sensitive formalisms, such as Tree Adjoining Grammar, have also been developed.[33]

Grammatical inference is the problem of learning a grammar from unlabeled data. Unlike the estimation methods previously discussed (the forward–backward and inside–outside algorithms), the target of learning is the grammar rules themselves, not just the weights on the rules. Inference of finite-state automata has been well studied.[34,35] Finite-state automata have been used in constructing language models for speech recognition, and similar techniques have been used in the unsupervised learning of morphology.[36] Methods for inferring context-free grammars have also been explored,[37–40] though with mixed results; recent work focuses on learning *dependency grammars*,[41,42] which are formally equivalent to context-free grammars but appear more tractable for learning. A common approach, both for finite-state automata and context-free grammars, is to start with an initial grammar, and iteratively modify it to optimize an *objective function*. One attractive objective function, derived from information theory, is *description length*, which is the total number of bits required first to encode the grammar, and then to encode the corpus using the grammar.[43]

REFERENCES

1. Shannon CE. A mathematical theory of communication. *Bell Syst Tech J* 1948, 27:379–423 and 623–656.
2. Brown P, Della Pietra S, Della Pietra V, Lai J, Mercer R. An estimate of an upper bound for the entropy of English. *Comput Linguist* 1992, 18:31–40.
3. Dempster AP, Laird NM, Rubin DB. Maximum likelihood from incomplete data via the EM algorithm. *J R Stat Soc B* 1977, 39:1–38.
4. Chen SF. Building probabilistic models for natural language. Doctoral dissertation. Harvard University; 1996.
5. Weaver W. Translation. In: Locke WN, Booth AD, eds. *Machine Translation of Languages: Fourteen Essays*. Cambridge, MA: Technology Press of the Massachusetts Institute of Technology; 1955.
6. Brown P, Della Pietra S, Della Pietra V, Mercer R. The mathematics of statistical machine translation: parameter estimation. *Comput Linguist* 1993, 19:263–312.
7. Berger A, Della Pietra S, Della Pietra V. A maximum entropy approach to natural language processing. *Comput Linguist* 1996, 22:39–72.
8. Freund Y, Schapire RE. Large margin classification using the perceptron algorithm. *Mach Learn* 1999, 37:277–296.
9. Freund Y, Schapire RE. A decision-theoretic generalization of on-line learning and an application to boosting. *J Comput Syst Sci* 1997, 55:119–139.
10. Burges CJC. A tutorial on support vector machines for pattern recognition. *Data Mining Knowl Discov* 1998, 2:121–167.

11. Vapnik VN. *Statistical Learning Theory*. New York: John Wiley & Sons; 1998.
12. Dietterich TG, Bakiri G. Solving multiclass learning problems via error-correcting output codes. *J Artif Intell Res* 1995, 2:263–286.
13. Schölkopf B, Smola AJ. *Learning with Kernels*. Cambridge, MA: MIT Press; 2002.
14. Collins M, Duffy N. New ranking algorithms for parsing and tagging: kernels over discrete structures, and the voted perceptron. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Stroudsburg, PA: Association for Computational Linguistics; 2002.
15. Yarowsky D. Unsupervised word sense disambiguation rivaling supervised methods. *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*. Stroudsburg, PA: Association for Computational Linguistics; 1995, 189–196.
16. Blum A, Mitchell T. Combining labeled and unlabeled data with co-training. *Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT)*. San Francisco, CA: Morgan Kaufmann Publishers; 1998, 92–100.
17. Zhu X, Ghahramani Z, Lafferty J. Semi-supervised learning using Gaussian fields and harmonic functions. In: *Machine Learning: Proceedings of the 20th International Conference (ICML)*, Washington, DC: International Machine Learning Society; 2003.
18. Zipf GK. *Human Behaviour and the Principle of Least Effort*. Reading MA: Addison Wesley; 1949.
19. Newman MEJ. Power laws, Pareto distributions and Zipf's law. *Contemp Phys* 2005, 46:323–351.
20. Church KW, Gale WA. Poisson mixtures. *Nat Lang Eng* 1995, 1:163–190.
21. Deerwester S, Dumais ST, Furnas GW, Landauer TK, Harshman R. Indexing by latent semantic indexing. *J Am Soc Inf Sci* 1990, 41:391–407.
22. Brent MR. Automatic acquisition of subcategorization frames from untagged text. *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*. Stroudsburg, PA: Association for Computational Linguistics; 1991, 209–214.
23. Resnik P. Selection and information. Doctoral dissertation. University of Pennsylvania; 1993.
24. Hindle D, Rooth M. Structural ambiguity and lexical relations. *Comput Linguist* 1993, 18: 103–120.
25. Finch SP. Finding structure in language. Doctoral dissertation. University of Edinburgh; 1993.
26. Chi Z, Geman S. Estimation of probabilistic context-free grammars. *Comput Linguist* 1998, 24:299–305.
27. Lari K, Young SJ. The estimation of stochastic context-free grammars using the Inside-Outside algorithm. *Comput Speech Lang* 1990, 4:35–56.
28. Magerman D. Natural language parsing as statistical pattern recognition. Doctoral dissertation. Stanford University; 1994.
29. Collins M. Head-driven statistical models for natural language parsing. Doctoral dissertation. University of Pennsylvania; 1999.
30. Charniak E. A maximum-entropy-inspired parser. *Proceedings of the Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL)*. Stroudsburg, PA: Association for Computational Linguistics; 2000, 132–139.
31. Collins M. Discriminative reranking for natural language parsing. *Comput Linguist* 2005, 31: 25–70.
32. Abney SP. Stochastic attribute-value grammars. *Comput Linguist* 1997, 23:597–618.
33. Resnik P. Probabilistic Tree-Adjoining Grammar as a framework for statistical natural language processing. *Proceedings of the International Conference on Computational Linguistics (COLING)*. Sheffield: International Committee on Computational Linguistics; 1992, 418–424.
34. Freund Y, Kearns M, Ron D, Rubinfeld R, Schapire RE, Sellie L. Efficient learning of typical finite automata from random walks. *Proceedings of the 25th ACM Symposium on the Theory of Computing*. New York, NY: ACM Press; 1993, 315–341.
35. Fu KS, Booth TL. Grammatical inference: introduction and survey. *IEEE Trans Syst Man Cybern* 1975, 5:59–72 and 409–423.
36. Goldsmith J. Unsupervised learning of the morphology of a natural language. *Comput Linguist* 2001, 27:153–198.
37. de Marcken C. Unsupervised language acquisition. Doctoral dissertation. Massachusetts Institute of Technology; 1996.
38. Horning JJ. A study of grammatical inference. Doctoral dissertation. Stanford University; 1969.
39. Stolcke A, Omohundro S. Inducing probabilistic grammars by Bayesian model merging. *Grammatical Inference and Applications, Second International Colloquium on Grammatical Inference*. Berlin: Springer Verlag; 1994.
40. Wolff JG. Language acquisition, data compression and generalization. *Lang Commun* 1982, 2:57–89.
41. Bechet D. K-valued link grammars are learnable from strings. In: *Proceedings of the 8th Conference on Formal Grammar*. Formal Grammar Committee, Haifa; 2003.
42. Klein D. The unsupervised learning of natural language structure. Doctoral dissertation. Stanford University; 2005.
43. Rissanen J. Modeling by shortest data description. *Automatica* 1978, 14:465–471.

FURTHER READING

The standard computational linguistics textbooks are Jurafsky and Martin and Manning and Schütze. The former is a general introduction to computational linguistics, with good coverage of statistical methods, and the latter covers statistical methods exclusively. Their introductory chapters treat the history and organization of statistical methods. Charniak is also a useful text, and Church and Mercer provide an overview of early work. The introductory chapter of Russell and Norvig discusses the adoption of statistical methods in artificial intelligence more broadly.

Jelinek covers many of the topics that have entered computational linguistics from speech recognition, including the noisy channel model, information theory, HMMs, the EM algorithm, maxent models, and smoothing. A specific introduction to information theory is Cover and Thomas. For HMMs, Rabiner and Juang provide a concise introduction and Rabiner provides a more detailed tutorial.

There are a number of excellent textbooks on machine learning, including Mitchell; Duda, Hart, and Stork; and Hastie, Tibshirani, and Friedman. For semisupervised learning in computational linguistics, see Abney.

Finally, the ACL Anthology is an indispensable electronic resource for publications in computational linguistics.

Abney SP. *Semisupervised learning for computational linguistics*. Boca Raton, FL: Chapman & Hall/CRC; 2008.

Charniak E. *Statistical language learning*. Cambridge, MA: MIT Press; 1993.

Church K, Mercer R. Introduction to the special issue on computational linguistics using large corpora. *Comput Linguist* 1993, 19:1–24.

Cover T, Thomas J. *Elements of Information Theory*. New York: John Wiley & Sons; 1991.

Duda RO, Hart PE, Stork DG. *Pattern Classification*. 2nd ed. New York: John Wiley & Sons; 2001.

Hastie T, Tibshirani R, Friedman J. *The Elements of Statistical Learning*. Berlin: Springer-Verlag; 2001.

Jelinek F. *Statistical Methods for Speech Recognition*. Cambridge, MA: MIT Press; 1997.

Jurafsky D, Martin JH. *Speech and language processing*. 2nd ed. Upper Saddle River, NJ: Prentice Hall; 2009.

Manning C, Schuetze H. *Foundations of Statistical Natural Language Processing*. Cambridge, MA: MIT Press; 1999.

Mitchell T. *Machine Learning*. New York: McGraw-Hill; 1997.

Rabiner LR. A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proc IEEE* 1989, 77:257–285.

Rabiner LR, Juang BH. An introduction to Hidden Markov Models. *IEEE ASSP Mag* 1986, 3:4–16.

Russell SJ, Norvig P. *Artificial Intelligence: A Modern Approach*. 2nd ed. Upper Saddle River, NJ: Prentice Hall; 2002.

ACL Anthology. Available at: http://www.aclweb.org/anthology-new. Accessed September 2009.

