283

ENCYCLOPEDIA OF COGNITIVE SCIENCE

2000

Macmillan Reference Ltd

Statistical Methods

Statistical methods#computational linguistics#machine learning#stochastic grammars

Abney, Steven

Steven Abney

AT&T Laboratories - Research

"Statistical methods" refers here specifically to statistical methods in computational linguistics. This represents a new body of practice in computational linguistics that has become standard over the last decade.

Introduction

Over the last decade or so a new body of practice has become standard in computational linguistics. It is known variously as *corpus-based*, *empirical*, or *statistical methods of language analysis*, most common being the simple rubric *statistical methods*. Present-day computational linguistics differs from "traditional" computational linguistics in the pervasiveness of probabilities in its theoretical models, the centrality of large data collections, including *text corpora* and *treebanks*, and the emphasis on rigorous empirical evaluation. The change in computational linguistics is part of a larger shift to statistical methods in computer science, particularly in artificial intelligence, pattern recognition, speech recognition, and machine learning.

The Re-Emergence of Empirical Linguistics

American Structuralism

The statistical paradigm has strong precedents in American structural linguistics. One of the major aims in structuralism was the development of procedures for taking a representative corpus of raw language data and determining the elements of which it is composed (the sounds and words of the language) and the conditions on their distribution. The motivation was methodological and philosophical: one wished to report only what was evident in the data, thereby avoiding the speculation and subjectivity that had been characteristic of earlier "philosophical" linguistics. Bloomfield famously wrote that "the only useful generalizations about language are inductive generalizations" (Bloomfield, 1933, p.20). Only the observable regularities in the data were of concern. Unobservables - in particular, meaning - had no place in structuralist descriptions.

Two important classes of linguistic elements are phonemes and morphemes. In the structuralist view, they are not abstract postulates, but rather, features of the data. Structuralist procedures for identifying phonemes and morphemes (and other aspects of structure) are generally known as *discovery procedures*, but in a real sense they are not so much concerned with *discovering* elements of structure as with *defining* them. A phoneme (for example) is defined to be what a given procedure returns when applied to the data. Faced with two alternative phonemicizations, a structuralist does not ask which one is correct. Definitions cannot be right or wrong. In the words of Harris, "they differ not in validity but in their usefulness for one purpose or another (e.g. for teaching the language, for describing its structure, for comparing it with genetically related languages.)" (Harris, 1951, p.9, fn.8).

Current computational linguistics takes a similar stance on the question of truth. Unlike structuralism, it does not eschew "deep" or "hidden" models - witness the discussion of stochastic grammars below - but its concern is utility rather than Platonic truth, and it insists on rigorous evaluation of model utility against a quantifiable measure of success at some task.

Current computational linguistics also follows structuralism in its interest in mechanically inducing from a corpus the elements of the language and the conditions on their arrangement. There are obvious parallels between some of the structuralist procedures and newer statistical methods. For example, one of the procedures that Harris used to segment a corpus into morphemes is the following. Consider an utterance, for example, /hiyzklev^{e}r/ "he's clever." Count the number of phonemes that could have occurred after /h/. That is, among all utterances in the corpus beginning with /h/, how many distinct segments appear in the second position? In Harris' estimate, there are nine (the English vowels and semi-vowels). After /hi/, there are 14 possibilities, after /hiy/, 29, after /hiyz/, 29, and after /hiyzk/, 11. Morpheme boundaries occur where the number of possibilities is highest, the intuition being that the constraints on succession are more stringent within a word than between words. Hence morpheme boundaries are defined to occur after /hiy/ and /hiyz/, but not after /h/ or /hiyzk/.

If we consider *probabilities* of phonemes instead of just *possibilities*, a natural analogue of Harris' proposal is to measure how much phoneme probabilities are affected by the context:[1]

(1) $p(\text{phoneme} \mid \text{context}) / p(\text{phoneme})$

The measure (1) is precisely the measure proposed by Stolz (1965) in an experiment to induce phrase boundaries. In information theory, the average of the logarithm of (1) is called *mutual information*; it is a key measure of coherence in modern statistical methods, and is commonly used to induce phrases.

In a similar way, the structuralist use of substitutability to define natural classes of elements has modern parallels. In structuralism, the class of nouns is defined as a class of elements that appear in similar contexts. Information theory provides a

[1] Harris' measure can actually be put into this form. Let "occur" be an indicator variable over phoneme types that has value 1 for phonemes that occur in a context and 0 otherwise. Then the number of phoneme types occurring in a context is proportional to $p(\text{occur} \mid \text{context}) / p(\text{occur})$. Note that $p(\text{occur})$ is 1 if "occur" is restricted to phoneme types in the language.

mathematically well-founded measure of substitutability: two elements are intersubstitutable if the *divergence* of their distributions is small. The divergence between distributions p and q is the average (with respect to p) of the log of $p(x)/q(x)$. It has been used for constructing classes of similarly-distributed elements (Finch, 1993).

Language Models

As the previous discussion suggests, information theory is an important tool for putting structural induction procedures on a firmer mathematical foundation. Information theory was motivated by the problem of transmitting information over a *noisy channel*. To transmit text (for example) over a telegraph wire with maximum efficiency and minimum error, it is necessary to identify the redundancies, which is to say, the regularities, in the text. This is precisely the task that the structuralists had set for themselves (though with a very different motivation).

In his seminal work on information theory, Shannon introduced the following problem, which has become known as the *Shannon Game* (Shannon, 1951). Take a random text and uncover it one element (e.g., one letter) at a time. At each point, the task is to predict the next element; it is not revealed until you guess correctly. The quantity used to measure difficulty in guessing is effective vocabulary size or *perplexity*: this is the vocabulary size that would cause a random guesser to make the same number of mistakes as you make. Your estimate of the text's perplexity is a measure of how good you are as a guesser. But Shannon also showed that there is a limit on how good any guesser can be. This *Shannon limit* is the inherent perplexity of the text (Shannon, 1948).

Algorithms that play the Shannon Game are known as *language models*. (To be precise, a language model is a probability distribution over all possible sequences of elements, and its ability to play the Shannon Game, as measured by its perplexity, is the measure of its quality as a language model.) A simple family of language models is the family of n-th order *Markov models*, which approximate the conditional probability $p(x_t \mid x_1 \ldots x_{t-1})$ of an element x_t given the corpus up to time t as $p(x_t \mid x_{t-(n-1)}, \ldots, x_{t-1})$, the probability of x_t given the preceding n-1 elements. Shannon showed that Markov models of increasing order converge to the Shannon limit: by choosing n large enough, the true distribution over language elements can be approximated arbitrarily closely.

However, Chomsky criticized Markov models (Chomsky, 1956). First, he emphasized that any given Markov model accounts for dependencies only up to a certain distance, leaving a residue of longer-distance dependencies not captured. Second, he pointed out that simple frequency counts are not adequate for estimating any but the lowest-order Markov models.

Concerning the first criticism, Markov models are mathematically useful because of their extreme simplicity, and they are surprisingly effective in practice, but they are obviously inadequate for many purposes, particularly for representing language semantics. (Unlike structuralism, modern computational linguistics is very much concerned with language meaning.) Soon after Chomsky defined context-free grammars, stochastic versions were explored, and have since been well developed; they are discussed below.

The second of Chomsky's criticisms is in part addressed by moving to more expressive grammars, and in part it is a technical issue concerning estimation of model parameters in the face of *sparse data*. This has been a major topic in speech recognition, and very sophisticated *smoothing* techniques have been developed to address it.

Causes of the Revival of Statistical Methods

In the late 1970s, Shannon's noisy channel model, and Markov models in particular, were applied to speech recognition by Jelinek and his colleagues. The speaker is approximated by a Markov model, and the channel includes both the conversion of words into sounds and the transmission of the sounds to the hearer. The setting as a whole is approximated by a Hidden Markov Model (HMM), a generalization of Markov models to the case in which the elements of the text are not directly observable (Baum et al., 1970). The result was a dramatic improvement in speech recognizer performance, and by the mid-1980s virtually all work on speech recognition was based on HMMs.

The state of the art in speech recognition systems is the trigram Markov language model, which predicts each word on the basis of the preceding two words. A trigram model is obviously a poor model of language - for example, if one generates text by random sampling from the distribution it defines, the results are not mistakable for real English text. However, it has proven remarkably difficult to improve on trigrams. Only within the last year or two have more sophisticated models been developed that significantly outperform trigrams.

Early on, HMMs were applied to the problem of assigning parts of speech to words (the *tagging* problem). That work, when it became known in the computational linguistics community, was the proximate cause of the surge of interest in statistical methods.

The impressive performance of statistical taggers attracted the attention of computational linguists, but the reason the statistical approach so quickly became the dominant paradigm is because it directly addressed several issues that had frustrated computational linguists immensely.

First was the desire for *robustness*. Real user input is noisy: it is full of misspellings, unanticipated syntactic constructions, and so on; and computational linguistics to that time had failed to develop genuinely noise-tolerant systems. A hallmark of statistical techniques is their noise tolerance.

Second was the desire for *portability*. Applying a manually constructed system to a new subject domain or a new language requires a prohibitive effort. By contrast, algorithms based on statistical methods can be adapted to new domains or new languages by training them on language corpora, and collecting and annotating corpora is usually easier than adapting systems by hand.

A third issue that had frustrated at least some computational linguists was the lack of measurable progress in the field. The statistical approach provides objective measures of success. One desires models that *generalize*: models that capture genuine regularities, not idiosyncracies of a given data set. A model that captures

idiosyncracies is said to *overfit* the training data. Generalization is measured by a model's performance on a *test set* that is representative of the universe of data of interest, but never seen during construction or training of the model.

Kinds of Statistical Methods

For expository purposes, statistical methods can be divided into three broad classes, which we consider in the following sections: corpus statistics, generative models, and classification and clustering.

Corpus Statistics

Corpus statistics are descriptive statistics that operationalize linguistic concepts. They are closest in spirit to the structuralist procedures, though with an important difference: the operationalizations are not taken to *define* linguistic concepts, but to *approximate* them. Examples are the use of mutual information as a measure of coherence, and divergence as a measure of distributional (dis)similarity. As mentioned above, they can be used to induce grammars; they can also be used to induce lexical information. Mutual information, for example, is used to identify multi-word terms such as "stock market." Other targets of lexical acquisition include subcategorization frames and selectional restrictions.

Generative Models

A second class of statistical methods involves probability distributions over families of structures. They can be classified by the complexity of the structures involved. Stochastic finite-state automata define distributions over strings, stochastic context-free grammars define distributions over trees, and stochastic attribute-value grammars define distributions over attribute-value structures.

For each class of grammars, there are three main questions of interest: how probabilities are attached to a grammar in such a way as to give a well-behaved probability distribution to the structures generated by the grammars; how the most likely structure can be computed for an arbitrary input; and how the probabilities in the stochastic grammar can be estimated from a sample.

Finite-State Automata (Hidden Markov Models)

A finite-state automaton (FSA) consists of a set of states, and a set of arcs leading from one state to another. Finite-state transducers of the most familiar sort (called "Mealey machines") associate output symbols with arcs. In stochastic FSAs, however, it is more common to use automata that associate output symbols with states ("Moore machines"). Machine computations consist in alternately producing an output symbol from the current state ("emission"), then following an arc to a new state ("transition").

In a stochastic FSA, a probability distribution is associated with the outputs from a given state, and a probability distribution is placed on the outgoing arcs from a given state. "Hidden Markov Model" is another name for a stochastic FSA of this type.

The probability of a computation is the product of probabilities of the individual emissions and transitions constituting the computation. The string generated by a computation is the concatenation of the strings generated in each emission step. The *derivation* of a string is the computation - that is, the sequence of states - by which it was generated.

A Hidden Markov Model is hidden in the sense that the derivation of a string cannot in general be uniquely determined. Nothing prevents there being more than one state that emits a given output symbol. To determine the most likely derivation, one can in principle enumerate all derivations of the string and compute their probabilities. This is impractical for strings of any length, inasmuch as the number of derivations increases exponentially with the length of the string.

Fortunately, there is an algorithm (the *Viterbi algorithm*) for computing the most-likely derivation in time linear in the length of the string. The Viterbi algorithm is a special case of dynamic programming. The key observation is that the most-likely partial derivation leading to state q at string position t consists of the most-likely partial derivation leading to some state q' at the previous position t-1, followed by the transition from q' to q. Instead of keeping track of all (exponentially-many) partial derivations at position t, we need only keep track of the most-likely partial derivation for each state q at t.

The discussion to now has assumed that transition and emission probabilities are given. In practice, however, they are not given, but must be estimated from a *training corpus*. A *labelled corpus* is one containing not only natural-language text, but also the sequence of states that the HMM passed through to generate the text. With a labelled corpus, we can essentially estimate HMM probability parameters by counting. For example, the probability of a transition from state q_1 to state q_2 is estimated as the relative frequency of transitions from q_1 to q_2 among transitions out of q_1 in the labelled corpus.

With an *unlabelled* training corpus, in which only the text is available, the sequence of actions taken by the HMM is unknown, and its probabilities obviously cannot be estimated by simple counting. In this case, the standard algorithm is the *forward-backward algorithm*, which is a special case of the *Expectation-Maximization (EM)* algorithm (Dempster et al., 1977). One begins with arbitrary parameter estimates - for example, uniform probabilities. The probability of every possible derivation is computed, and one pretends that a derivation occurs a fractional number of times, in proportion to its probability. This gives one a labelled corpus, from which new probabilities can be estimated by relative frequency. One then repeats the process with the new probability estimates. It can be shown that this method improves the probability estimates at each iteration, measuring goodness by the standard *maximum likelihood* criterion.

Even counting relative frequencies involves some subtleties. It is complicated by the *sparse data problem*: the fact that many unobserved actions fail to occur only because the training corpus is not large enough. Indeed, in most cases of practical interest, the majority of possible actions fail to occur even in the largest available corpora. Methods to address the sparse data problem are known as *smoothing methods*. A large variety of them have been studied (Chen, 1996). The easiest is simply to pretend that every possible action occurred at some low frequency. That is, one adds a

small count (usually 1 or 1/2) to every count, including the zero counts, before taking relative frequencies. Much better smoothing methods are available; the most commonly used are *Katz back-off* and *deleted interpolation*.

In addition to speech recognition and part-of-speech tagging, stochastic FSAs have applications in *entity recognition* (that is, detecting references in text to people, companies, dates, times, monetary amounts, and so on) and *partial parsing* (recognizing the major phrases and clauses of a text without completely parsing it).

Stochastic Context-Free Grammars

The next more complex grammar class comprises the context-free grammars. A context-free grammar consists in a set of rules of form $A \rightarrow Y_1 \ldots Y_n$, in which A is a nonterminal symbol and $Y_1 \ldots Y_n$ is a (possibly empty) sequence of mixed terminal and nonterminal symbols. $Y_1 \ldots Y_n$ is said to be an *expansion* of A.

A derivation begins with a single, distinguished, nonterminal symbol S. At each point in the derivation, the leftmost nonterminal symbol is replaced with one of its expansions. The derivation continues until no nonterminal symbols remain. A derivation is equivalent to a parse-tree. Each node in the tree represents an expansion: the parent node is labeled with the nonterminal A that is being expanded, and its child nodes are labeled with the expansion symbols $Y_1 \ldots Y_n$.

In a *stochastic context-free grammar*, probabilities are associated with expansions. For any given nonterminal symbol, the probabilities of all its expansions sum to one. The probability of a derivation is the product of the probabilities of the expansions constituting the derivation.

One can modify practically any context-free parsing algorithm to recover the most-likely parse. For example, the *CKY parsing algorithm* proceeds as follows. The grammar is assumed to be in Chomsky-normal form, meaning that all expansions are of form $A \rightarrow B\ C$ (two nonterminals in the expanded form) or $A \rightarrow a$ (one terminal in the expanded form). This assumption involves no loss of generality, as any CFG can be converted to an equivalent grammar in Chomsky-normal form. All possible parse-tree nodes are constructed, in order of increasing width, where the width of a parse-tree node is the number of words of input it covers. For each triple (X,i,w), where X is a nonterminal category, i is a start position, and w is a width, only the most-probable subtree is recorded. Since (X,i,w) is constructed out of subtrees of smaller width, all its possible components are guaranteed to exist.

There is also a specialization of the EM algorithm, known as the *inside-outside algorithm*, that can be used to estimate the probabilities of an SCFG from unlabelled data. Unfortunately, grammars trained using the inside-outside algorithm produce parse-trees that are not useful for sentence interpretation. This is attributed to the fact that the inside-outside algorithm is designed to minimize sentence perplexity; it has no source of information concerning sentence meaning. As a practical matter, stochastic parsers are trained using labelled data, known as *treebanks*.

Stochastic Attribute-Value Grammars

For our purposes, attribute-value structures can be thought of as directed acyclic graphs (DAGs) with labeled edges. DAGs differ from trees in that DAGs contain *re-entrancies*: nodes that have multiple parents. DAG nodes represent either parse-tree nodes or their values for given attributes. For example, a singular noun phrase can be represented as a node labeled "noun phrase" linked by an edge labeled "number" (an attribute) to a node labeled "singular" (a value). The constituents of the noun phrase are distinguished by edges with labels "child 1", "child 2", etc.

Attribute-value structures are generated by attribute-value grammars. Rules in an attribute-value grammar are context-free rules equipped with constraints. Constraints determine the re-entrancies in the DAG. For example, the rule

$$S \rightarrow NP\ VP;\ NP.number = VP.number$$

specifies that the node representing the noun phrase's value for attribute "number" is the very same node as the one representing the verb phrase's value for "number".

Attribute-value grammars are stochasticized by attaching weights to their rules. The probability of a DAG is the product of weights of the rules that define it. Unlike in the finite-state and context-free cases, however, the rule weights cannot be called probabilities. In the finite-state and context-free cases, structures are built up of a number of independent stochastic decisions, and because of the independence of local decisions, the probability of the structure as a whole is the product of local decision probabilities. In the attribute-value case, re-entrancies introduce dependencies among local decisions. Global probabilities are defined as products of local weights, but because of the dependencies among local decisions, the weights are not local probabilities.

Stochastic attribute-value grammars are essentially a variant of *Markov random fields* or *graphical models*. There is a considerable literature on estimation of graphical models. For stochastic attribute-value grammars, estimation methods that have been considered include varieties of *Iterative Scaling*.

No tractable exact parsing algorithm is known for stochastic attribute-value grammars. Because of the dependencies among substructures, dynamic programming is not possible. For practical purposes, a common technique is to use stochastic context-free parsing to obtain a small set of candidate structures, which are then evaluated using the full attribute-value grammar.

The unavailability of a dynamic programming algorithm also means that there is no advantage in attaching weights solely to local rules. Typically, weights are attached to features that span much more of the structure than a local expansion.

Specialized Generative Models

Specialized generative models are often developed for specific tasks. Prominent examples are *corpus alignment* and *machine translation*. Corpus alignment involves lining up sentences or smaller phrases between corpora in two different languages, one of which is a translation of the other. Machine translation can be thought of as a similar task, but one in which the problem is to generate the source-language text that

would align best with the (observed) target-language text. There have been efforts to estimate fairly direct transfer models, as well as efforts to equip more traditional "deep" translation models with probabilities.

Classification and Clustering

There is a lively interchange between computational linguistics and machine learning. New machine learning techniques are continually introduced into computational linguistics, and the unique challenges of language learning stimulate new directions of research in machine learning.

A central topic in machine learning is classification. The aim of classification is to determine which of a fixed number of classes a given item belongs to. Classification is a supervised learning method - the learning algorithm is given a training corpus of correctly classified examples. A simple example of a classification problem in computational linguistics is prepositional phrase attachment. One widely-used data set is constructed from verb phrases of the form verb - noun phrase - prepositional phrase, for example "was selling machine parts from Dresden." The task is to classify each such example as "noun attachment" (parts that are from Dresden) or "verb attachment" (they were sold from Dresden). Generative models can be used in service of classification, but classification can also be done without generative models. A wide variety of classification techniques have been applied to computational linguistic problems, including classification and regression trees (CART), decision lists, Naive Bayes, likelihood ratios, and margin-based methods such as support vector machines (SVMs) and boosting.

Another area of especially strong interaction between machine learning and computational linguistics is unsupervised learning. In unsupervised learning, the training material is not labelled with correct answers. An example is clustering, which is used to induce classes of words, for example in language modelling or in the induction of selectional restrictions.

Intermediate between supervised and unsupervised learning is bootstrapping. In bootstrapping, the algorithm is given a very small amount of labelled data, and a large amount of unlabelled data. It can be viewed as supervised learning with supplementary unlabelled data, or as unsupervised learning in which the labelled set is used to give names to the clusters. It has been successfully applied to word-sense disambiguation and named entity classification, among other things.

Summary

Statistical methods have become the standard paradigm in computational linguistics. They can be placed in the historical perspective of American structuralism, though they derive more immediately from statistical speech recognition and Shannon's noisy channel model. They can be grouped roughly into descriptive statistics, generative models (stochastic finite-state, context-free, and attribute-value grammars), and machine learning methods (classification, clustering, bootstrapping).

References

Baum LE, Petrie T, Sopules G and Weiss N (1970) A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *Annals of Mathematical Statistics* **41**: 164-171.

Bloomfield L (1933) *Language.* New York: Holt.

Chen SF (1996) *Building Probabilistic Models for Natural Language*. Doctoral dissertation, Harvard University.

Chomsky N (1956) Three models for the description of language. *IRE Transactions on Information Theory* **IT-2**.3: 113-124.

Dempster AP, Liard NM and Rubin DB (1977) Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society* **B**.39: 1-38.

Finch SP (1993) *Finding Structure in Language*. Doctoral dissertation, University of Edinburgh.

Harris Z (1951) *Structural Linguistics*. Chicago: Chicago University Press.

Shannon CE (1948) A mathematical theory of communication. *The Bell System Technical Journal* **27**.3-4: 379-423, 623-656.

Shannon CE (1951) Prediction and entropy of printed English. *The Bell System Technical Journal* **30**: 50-64.

Stolz W (1965) A probabilistic procedure for grouping words into phrases. *Language and Speech* **8**: 219-325.

Further Reading

Charniak E (1993) *Statistical Language Learning*. Cambridge, Mass.: MIT Press.

Jelinek F (1997) *Statistical Methods for Speech Recognition*. Cambridge, Mass.: MIT Press.

Jurafksy D and Martin JH (2000) *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech.* Upper Saddle River, New Jersey: Prentice-Hall.

Manning CD and Schütze H. (1999) *Foundations of Statistical Natural Language Processing.* Cambridge, Mass.: MIT Press.

Glossary

Corpus, pl. **corpora**. A collection of text, typically representing either a random sample from numerous genres (as for example the Brown corpus), or a large archive of text from a single source (such as the archives of a single newspaper).

Divergence. An information-theoretic measure of the dissimilarity of two distributions.

Expectation-Maximization (EM) algorithm. An algorithm for estimating the parameters of models that contain hidden (unobservable) structure.

Hidden Markov Model (HMM). A stochastic finite-state model whose output is known, but for which the state sequence that produced the output is unknown.

Information theory. A branch of mathematics that quantifies the information content of communications, with applications that include compression, error correction, and encryption.

Language model. A statistical model defining a probability distribution over the word sequences of a language.

Mutual information. The amount of information that one random variable contains regarding another random variable. Uncorrelated variables have a mutual information of zero.

Overfitting. A model is said to *overfit* if it incorporates idiosyncracies of the training set that are not generally valid.

Smoothing. Statistical estimation techniques that are used when the training data is insufficient for estimating all parameters of a large model.

Supervised (unsupervised) learning. In supervised learning, the learning algorithm is provided with training data that has been manually labeled with the correct answers. In unsupervised learning, only unlabeled data is provided.

Treebank. A text corpus in which each sentence has been manually assigned a syntactic analysis.

Viterbi algorithm. An efficient algorithm which, given the output sequence produced by an HMM, computes the most-likely sequence of states that the HMM passed through.

