MEASURES AND MODELS FOR PHRASE RECOGNITION

Steven Abney

Bell Communications Research
445 South Street
Morristown, NJ 07960

ABSTRACT

I present an entropy measure for evaluating parser performance. The measure is fine-grained, and permits us to evaluate performance at the level of individual phrases. The parsing problem is characterized as statistically approximating the Penn Treebank annotations. I consider a series of models to "calibrate" the measure by determining what scores can be achieved using the most obvious kinds of information. I also relate the entropy measure to measures of recall/precision and grammar coverage.

1. INTRODUCTION

Entropy measures of parser performance have focussed on the parser's contribution to word prediction. This is appropriate for evaluating a parser as a language model for speech recognition, but it is less appropriate for evaluating how well a parser does at *parsing*. I would like to present an entropy measure for phrase recognition, along with closely-related measures of precision and recall. I consider a series of models, in order to establish a baseline for performance, and to give some sense of what parts of the problem are hardest, and what kinds of information contribute most to a solution.

Specifically, I consider the problem of recognizing *chunks* (Abney 1991)—non-recursive pieces of major-category phrases, omitting post-head complements and modifiers. Chunks correspond to prosodic phrases (Abney 1992) and can be assembled into complete parse trees by adding head-head *dependencies*.

2. THE PARSING PROBLEM

Parsing is usually characterized as the problem of recovering parse trees for sentences, *given a grammar* that defines the mapping of sentences to parse-trees. However, I wish to characterize the problem without assuming a grammar, for two reasons. First, we cannot assume a grammar for unrestricted English. For unrestricted English, failure of coverage will be a significant problem for any grammar, and we would like a measure of performance that treats failure of coverage and failures within the grammar uniformly.

Second, I am particularly interested in parsers like Fidditch (Hindle 1983) and Cass (Abney 1990) that avoid search by relying on highly reliable patterns for recognizing individual phrases. Such parsers may need to consider competing patterns when scoring a given pattern—for example, Cass relies heavily on a preference for the pattern that matches the longest prefix of the input. Such cross-pattern dependencies cannot be expressed within, for example, a stochastic context-free grammar (SCFG). Hence I am interested in a more general evaluation framework, one that subsumes both Fidditch/Cass-style parsers and SCFG parsing.

Instead of assuming a grammar, I take the Penn Treebank (Marcus & Santorini 1991) to provide a representative sample of English, viewed as a function from sentences to parse trees. A parser's task is to statistically approximate that function. We can measure the (in)accuracy of the parser by the amount of additional information we must provide in order to specify the correct (Treebank) parse for a sentence, given the output of the parser. This is the entropy of the corpus given the parser, and approaches zero as the parser approaches perfect emulation of Treebank annotation.

We can characterize the parser's task at two levels of granularity. At the level of the sentence, the task is to assign a probability distribution over the set of possible parse-trees for the sentence. At the phrase level, the problem is to give, for each candidate phrase c, the probability that c belongs to the correct parse. I will focus on the latter characterization, for several reasons: (1) as mentioned, I am interested in developing reliable patterns for recognizing individual phrases, in order to reduce the necessity for search and to increase parsing speed, (2) evaluating at the phrase level allows us to assign blame for error at a finer grain, (3) there are applications such as data extraction where we may have good models for certain phrase types, but not for entire sentences, and (4) a phrase model can easily be embedded in a sentence model, so evaluating at the finer grain does not exclude evaluation at the coarser grain.

3. MEASURES

Given a sentence, the chunk *candidates* are all tuples $c = (x,i,j)$, for x a syntactic category, and i and j the start and end positions of the chunk. For each candidate, there are two possible events in the Treebank: the candidate is indeed a phrase in the Treebank parse (T), or it is not a true phrase (~T). For each candidate, the parsing model provides $P(T|c)$, the probability of the candidate being a true phrase, and $P(\sim T|c) = 1 - P(T|c)$.

Given the probabilities provided by the parsing model, the information that must be provided to specify that T occurs (that the candidate is a true phrase) is $-\lg P(T|c)$; and to specify that ~T occurs, $-\lg P(\sim T|c)$. The entropy of the corpus given the model is the average $-\lg P(E_c|c)$, for E_c being T or ~T according as candidate c does or does not appear in the Treebank parse. That is,

$$H = -(1/N) \sum_c \lg P(E_c|c)$$ for N the number of candidates

A perfect model would have $P(E_c|c) = 1$ for all c, hence $H = 0$. At the other extreme, a 'random-guess' model would have $P(E_c|c) = 1/2$ for all c, hence $H = 1$ bit/candidate (b/c). This provides an upper bound on H, in the sense that any model that has $H > 1$ b/c can be changed into a model with $H < 1$ by systematically interchanging $P(T|c)$ and $P(\sim T|c)$. Hence, for all models, $0 \le H \le 1$ b/c.

There are some related measures of interest. We can translate entropy into an equivalent number of equally-likely parses (perplexity) by the relation:

$$PP = 2^{\alpha H}$$

for H in bits/candidate and α the number of candidates per sentence. In the test corpus I used, $\alpha = 8880$, so PP ranges from 1 to $2^{8880} = 10^{2670}$.

We can also measure expected precision and recall, by considering $P(T|c)$ as a probabilistic 'Yes' to candidate c. For example, if the model says $P(T|c) = 3/4$, that counts as 3/4 of a 'Yes'. Then the expected number of Yes's is the sum of $P(T|c)$ over all candidates, and the expected number of correct Yes's is the sum of $P(T|c)$ over candidates that are true chunks. From that and the number of true chunks, which can simply be counted, we can compute precision and recall:

$$E(\#Y) = \sum_c P(T|c)$$

$$E(\#TY) = \sum_{\text{true } c} P(T|c)$$

$$EP = E(\#TY) / E(\#Y)$$

$$ER = E(\#TY) / \#T$$

4. MODELS

To establish a baseline for performance, and to determine how much can be accomplished with 'obvious', easily-acquired information, I consider a series of models. Model 0 is a zero-parameter, random-guess model; it establishes a lower bound on performance. Model 1 estimates one parameter, the proportion of true chunks among candidates. Model XK takes the category and length of candidates into account. Model G induces a simple grammar from the training corpus. Model C considers a small amount of context. And model S is a sentence-level model based on G.

4.1. Models 0 and 1

Models 0 and 1 take P(T|*c*) to be constant. Model 0 (the random-guess model) takes P(T) = 1/2, and provides a lower bound on performance. Model 1 (the one-parameter model) estimates P(T) as the proportion of true chunks among candidates in a training corpus. The training corpus I used consists of 1706 sentences, containing 19,025 true chunks (11.2 per sentence), and 14,442,484 candidates (8470 per sentence). The test corpus consisted of 1549 sentences, 17,676 true chunks (11.4 per sentence), and 13,753,628 candidates (8880 per sentence). The performance of the random-guess and one-parameter models is as follows:

	b/c	prs/sent	EP	ER
0	1	10^{2670}	.129%	(50%)
1	.014 2	10^{38}	.129%	(.132%)

For these two models (in fact, for any model with P(T|*c*) constant), precision is at a minimum, and equals the proportion of true chunks in the test corpus. Recall is uninformative, being equal to P(T|*c*).

4.2. Model XK

Model XK is motivated by the observation that very long chunks are highly unlikely. It takes P(T|*c*) = P(T|*x*,*k*), for *x* the category of *c* and *k* its length. It estimates P(T|*x*,*k*) as the proportion of true chunks among candidates of category *x* and length *k* in the training corpus. As expected, this model does better than the previous ones:

	b/c	prs/sent	EP	ER
XK	.007 95	10^{21}	5.5%	5.6%

4.3. Models G and C

For model G, I induced a simple grammar from the training corpus. I used Ken Church's tagger (Church 1988) to assign part-of-speech probabilities to words. The grammar contains a rule $x \rightarrow \gamma$ for every Treebank chunk $[_x\ \gamma]$ in the training corpus. (x is the syntactic category of the chunk, and γ is the part-of-speech sequence assigned to the words of the chunk.) $[_x\ \gamma]$ is counted as being observed P(γ) times, for P(γ) the probability of assigning the part-of-speech sequence γ to the words of the chunk. I used a second corpus to estimate P(T|*x*,γ) for each rule in the grammar, by counting the proportion of true phrases among candidates of form $[_x\ \gamma]$. For candidates that matched no rule, I estimated the probabilities P(T|*x*,*k*) as in the XK model.

Model C is a variant of model G, in which a small amount of context, namely, the following part of speech, is also taken into account.

The results on the test corpus are as follows:

	b/c	prs/sent	EP	ER
G	.003 81	10^{10}	47.3%	48.2%
C	.003 36	10^{9}	54.5%	58.7%

The improvement in expected precision and recall is dramatic.

4.4 Assigning Blame

We can make some observations about the sources of entropy. For example, we can break out entropy by category:

	%H	–E(%H)
NP	39.0	+18.7
PP	21.1	+7.2
VP	19.0	+4.4
Null	7.5	–8.4
AdjP	3.9	+1.4
other (23)	9.5	–23.4

The first column represents the percentage of total entropy accounted for by candidates of the given category. In the second column, I have subtracted the amount we would have expected if entropy were divided among candidates without regard to category. The results clearly confirm our intuitions that, for example, noun phrases are more difficult to recognize than verb clusters, and that the Null category, consisting mostly of punctuation and connectives, is easy to recognize.

We can also break out entropy among candidates covered by the grammar, and those not covered by the grammar. The usual measure of grammar coverage is simply the proportion of true chunks covered, but we can more accurately determine how much of a problem coverage is by measuring how much we stand to gain by improving coverage, versus how much we stand to gain by improving our model of covered candidates. On our test corpus, only 4% of the candidates are uncovered by the grammar, but 19% of the information cost (entropy) is due to uncovered candidates.

4.5. Model S

None of the models discussed so far take into account the constraint that the set of true chunks must partition the sentence. Now, if a perfect sentence model exists—if an algorithm exists that assigns to each sentence its Treebank parse—then a perfect phrase model also exists. And to the extent that a model uses highly reliably local patterns (as I would like), little information is lost by not evaluating at the sentence level. But for other phrase-level models, such as those considered here, embedding them in a sentence-level model can significantly improve performance.

Model S is designed to gauge how much information is lost in model G by not evaluating parses as a whole. It uses model G's assignments of probabilities P(T|*c*) for individual candidates as the basis for assigning probabilities P(*s*) to entire parses, that is, to chunk-sequences *s* that cover the entire sentence.

To choose a sequence of chunks stochastically, we begin with *s* = the null sequence at position *i* = 0. We choose from among the candidates at position *i*, taking the probability P(*c*) of choosing candidate *c* to be proportional to P(T|*c*). The chosen chunk *c* is appended to *s*, and the current position *i* is advanced to the end position of *c*. We iterate to the end of the sentence. In brief:

$$P(c) = P(T|c) \Big/ \sum_{c\text{« at } i} P(T|c\text{«}) \qquad \text{for } i \text{ the start position of } c$$

$$P(s) = \prod_{c \text{ in } s} P(c)$$

The entropy of a sentence given the model is –lg P(*s*), for *s* the true sequence of chunks. We can also compute actual (not expected) precision and recall by counting the true chunks in the most-likely parse according to the model. The results on the test corpus are:

M	b/s	prs/sent	Precision	Recall
S	14.1	10^4	74.1%	75.6%

(By way of comparison, the bits/sentence numbers for the other models are as follows:)

0	1	XK	G	C	S
8880	126	70.6	33.8	29.8	14.1

For model S, the number of parses per sentence is still rather high, but the precision and recall are surprisingly good, given the rudimentary information that the model takes into account. I think there is cause for optimism that the chunk recognition problem can be solved in the near term, using models that take better account of context and word-level information.

4. CONCLUSION

To summarize, I have approached the problem of parsing English as a problem of statistically approximating the Penn Treebank. For the purposes of parsing, English is a function from sentences to parse-trees, and the Treebank provides a (sufficiently representative) sample from the extension of that function. A parsing model approximates Treebank annotation. Our basic measure of the goodness of the approximation is the amount of additional information we must provide in order to specify the Treebank parse, given the probabilities assigned by the parser. I have presented a series of models to "calibrate" the measure, showing what kind of performance is achievable using obvious kinds of information.

An impetus for this work is the success of parsers like Fidditch and Cass, which are able to greatly reduce search, and increase parsing speed, by using highly reliable patterns for recognizing phrases. The limitation of such work is the impracticality of constructing reliable patterns by hand, past a certain point. One hindrance to automatic acquisition of reliable patterns has been the lack of a framework for evaluating such parsers at a fine grain, and exploring which kinds of information contribute most to parsing accuracy.

In the current work, I have presented a framework for fine-grained evaluation of parsing models. It does not assume stochastic context-free grammars, and it quantifies parsers' performance at parsing, rather than at a more indirectly related task like word prediction.

REFERENCES

1. Steven Abney (1990). Rapid Incremental Parsing with Repair. Proceedings of the 6th New OED Conference. University of Waterloo, Waterloo, Ontario.
2. Steven Abney (1991). Parsing by Chunks. In Berwick, Abney & Tenny, eds. Principle-Based Parsing, pp.257-278. Kluwer Academic Publishers, Dordrecht.

3. Steven Abney (1992). Prosodic Structure, Performance Structure and Phrase Structure. Proc. 5th DARPA Workshop on Speech and Natural Language (Harriman, NY). Morgan Kaufmann.

4. E. Black, S. Abney, D. Flickenger, R. Grishman, P. Harrison, D. Hindle, R. Ingria, F. Jelinek, J. Klavans, M. Liberman, M. Marcus, S. Roukos, B. Santorini, and T. Strzalkowski (1991). A procedure for quantitatively comparing the syntactic coverage of English grammars. DARPA Speech and Natural Language Workshop, pp.306-311. Morgan Kaufmann.

5. Peter L. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, Jennifer C. Lai, and Robert L Mercer (1992). An Estimate of an Upper Bound for the Entropy of English. Computational Linguistics 18.1, pp.31-40.

6. Kenneth Church (1988). A Stochastic Part of Speech Tagger and Noun Phrase Parser for English. Proceedings of the 2nd Conference on Applied Natural Language Processing. Austin, Texas.

7. T. Fujisaki, F. Jelinek, J. Cocke, E. Black, T. Nishino (1989). A Probabilistic Parsing Method for Sentence Disambiguation. International Workshop on Parsing Technologies 1989, pp.85-94.

8. Donald Hindle (1983). User manual for Fidditch. Naval Reserach Laboratory Technical Memorandum #7590-142.

9. F. Jelinek. Self-Organized Language Modeling for Speech Recognition. IBM report.

10. F. Jelinek, J.D. Lafferty, and R.L. Mercer. Basic Methods of Probabilistic Context-Free Grammars. IBM report.

11. Mitchell Marcus and Beatrice Santorini (1991). Building very large natural language corpora: the Penn Treebank. Ms., University of Pennsylvania.

12. Fernando Pereira and Yves Schabes (1992). Inside-Outside Reestimation from Partially Bracketed Corpora. ACL 92, pp.128-135.