Evaluating Parsers

- We need a measure to evaluate parser performance against gold standard
  - ratio of fully correct sentences parses too coarse
  - ratio of correct constituents

- Does correct mean precision?
  \[
  \text{precision} = \frac{\text{count}(\text{matching constituents})}{\text{count}(\text{predicted constituents})}
  \]

- Does correct mean recall?
  \[
  \text{precision} = \frac{\text{count}(\text{matching constituents})}{\text{count}(\text{gold standard constituents})}
  \]

High Precision, Low Recall

system:

- John
  - NNP
  - IN
  - NN
  - CC
  - NNP

- from
  - NN
  - PP
  - CC
  - NNP

- Hoboken and Jim
  - NNP
  - IN
  - NN
  - PP

gold standard:

- John from Hoboken and Jim
  - NNP
  - IN
  - NN
  - CC
  - NNP

all predicted constituents match gold standard (precision 1/1)

... but we are missing quite a few (recall 1/6)
Low Precision, High Recall

system

gold standard

all gold standard constituents are predicted (recall 6/6)
... but we are predicting many more (precision 6/10)

PARSEVAL

- F-measure: balance of precision and recall
  \[ F_1 = \frac{\text{precision} \times \text{recall}}{(\text{precision} + \text{recall})/2} \]

- F-measure is used in many other NLP tasks and may be adjusted to give more emphasis to either precision or recall

Outline

- Evaluating Parsers
- Parsing Complexity
- Discriminative Approaches

Parsing Complexity

- CKY decoding involves the construction of a chart

- The chart has \( O(n^2) \) contiguous spans
Parsing Complexity (2)

• When building entries for a span, $O(n)$ different combination of smaller spans are possible

$W_1 W_2 W_3 W_4$

- ... assuming binary grammars (at most two non-terminal on the right)
- but then grammars can always be binarized

⇒ Parsing complexity is $O(n^3)$

Comments on Parsing Complexity

• CKY parsing is $O(n^3)$ with respect to sentence length
  - the number of different non-terminals also plays a role

• Not the end of the world, but long sentences are a problem

• And this assumes binary grammars
  - more complex grammars may be binarized
  - ... but that increases the number of non-terminals dramatically

• Parsing speed may be improved with heuristic beam search that focuses on the more promising parses

Coarse-to-Fine Parsing

• Parsing the sentence in stages
  - First, use a reduced grammar (e.g. fewer non-terminals)
  - Then, reduced search with full grammar

• Reduction in search
  - limit exploration of intermediate spans to viable paths to full parse
  - use first stage to obtain outside cost estimates

Outside Cost Estimation

• Heuristic estimate on how expensive it will be to parse the rest of the sentence
Outside Cost Estimates

Some spans are more promising than others

Outline

- Evaluating Parsers
- Parsing Complexity
- Discriminative Approaches

Global Features

- For instance: parallel structures in coordination

Structured Prediction

- The proposed statistical parsing model is a generative model
  - predicting the parse tree is broken down into a sequence of steps (derivation)
  - each step is modeled by a conditional probability distribution
  - probability distributions are estimated over the training data
- Discriminative approach
  - each possible parse tree is defined by a set of features
  - each feature has a weight that determines its importance
  - directly optimize on a performance criterion (parser performance)
Global Linear Models

- A generating function $\text{GEN}$ maps an input $x$ to candidates trees $y_1, \ldots, y_n$
  \[ \text{GEN}(x) = \{y_1, \ldots, y_n\} \]
- Each feature function $h_i$ maps a parse tree $(x, y)$ to a feature value $h(x, y)$
- Features are combined in a linear model
  \[ \sum_i \lambda_i h_i(x, y) \]
- The goal of learning is to find the feature weights $\lambda_i$

Features in Parsing

- Rule applications
  - number of times the rule $\text{NP} \rightarrow \text{NP PP}$ is used
- Long distance features
  - number of times $\text{Mary}$ is object of $\text{likes}$
- Complex structural features
  - number of parallel co-ordinations
- Other models
  - parse probability under Collins’ generative parsing model

$n$-Best List Re-ranking

- Use the generating function to generate the top $n$ most likely parses for an input sentence
  - for instance, using the generative parsing model
- Evaluate each parse tree against the gold standard
- Use a machine learning method to optimize re-ranking of the $n$-best list so that the highest scoring (or at least higher scoring) parse come out at the top
  - for instance, the Perceptron algorithm

Perceptron Algorithm

Input: set of sentences with gold standard parses $(x, y)$, set of features $h_i$
Output: set of weights $\lambda_i$ for each feature
1: $\lambda_i = 0$ for all $i$
2: while not converged do
3:   for all sentences $x$ do
4:     $y_{\text{best}} = \text{best parse tree according to model}$
5:     $y_{\text{gold}} = \text{gold standard parse tree}$
6:     if $y_{\text{best}} \neq y_{\text{gold}}$ then
7:       for all features $h_i$ do
8:         $\lambda_i += h_i(x, y_{\text{gold}}) - h_i(x, y_{\text{best}})$
9:       end for
10:   end if
11: end for
12: end while
Oracle Performance

• It is often useful to ask: what is possible?

• Oracle performance
  – for each sentence
    * match all candidates against goals standard
    * store best-matching candidate
  – compute overall performance over this set

⇒ upper limit of what can be gained with re-ranking

Oracle Performance (2)

• Often one finds:
  – $n$-best lists are too limiting
  – parse forests (extracted from search) better
  – re-parsing with new model best
    ... but often too expensive

• Caution
  – Oracle often too optimistic for actual possible performance
  – higher Oracle does not imply a better set of candidates

Grand Challenge in Structured Prediction

• Many algorithms require computation of the current best parse under the model
  – for instance, Perceptron, gradient descent, ...

• Finding best parse often computationally hard
  – especially when using global features

• Challenge: find efficient search methods for structured prediction problems