A probabilistic parser produces pairs \((y, x)\), where \(y\) is a parse tree and \(x\) is a sentence.

A probabilistic context-free grammar is a set \(R\) of rules \(A \rightarrow \alpha\), each associated with a parameter \(\theta_{A \rightarrow \alpha}\), subject to:

\[
\sum_{\alpha: (A \rightarrow \alpha) \in R} \theta_{A \rightarrow \alpha} = 1
\]

This ensures that the parameters are probabilities.

For each rule \(r \in R\), let \(f_r(y)\) be the number of times \(r\) is used in the derivation \(y\). Then the probability distribution over trees is:

\[
P_{\theta}(y) = \prod_{(A \rightarrow \alpha) \in R} (\theta_{A \rightarrow \alpha})^{f_{A \rightarrow \alpha}(y)}
\]

\(\theta\) can be computed using a relative frequency estimator:

\[
\hat{\theta}_{A \rightarrow \alpha} = \frac{\sum_{i=1}^{n} f_{A \rightarrow \alpha}(y_i)}{\sum_{i=1}^{n} \sum_{\alpha': (A \rightarrow \alpha') \in R} f_{A \rightarrow \alpha'}(y_i)}
\]

**Example**: assume a training corpus with the following two trees.
Most standard models in NLP are generative models. They are trained to maximize the joint distribution over the training data:

$$\hat{\theta} = \arg \max_\theta \prod_{j=1}^n P_\theta(y_j, x_j)$$

(2)

Here, $D = ((y_1, x_1), \ldots, (y_n, x_n))$ is a training corpus in which a hidden variable $Y$ is paired with an observed variable $X$.

For probabilistic context-free grammars, the relative frequency estimator in (1) is the maximum likelihood estimator in (2).
An alternative is to use **discriminative models**. They are trained to maximize the following conditional distribution:

$$\hat{\theta} = \arg \max_{\theta} \prod_{i=1}^{n} P_{\theta}(y_i|x_i)$$

This means we use **maximum conditional likelihood estimation** instead of maximum likelihood estimation (Johnson 2001).

**Discriminative Parsing**

The standard probabilistic parsing model is a **generative model**:

- a parse tree is generated through a sequence of steps (a derivation);
- each step is modeled by a conditional probability distribution;
- training: estimating these distributions from the training data;

**Alternative: discriminative parsing model**:

- a parse tree is defined by a set of features;
- each feature has a weight that determines its importance;
- training: estimating these weights so as to optimize a performance metric (here: $F$-score).

Advantages of the discriminative approach:

- arbitrary features can be used (including things that are hard to encode in a derivation, e.g., global properties of trees);
- can use large numbers of features, even related ones, the weighting procedure will select the important ones;
- discriminative model can be applied on top of a standard generative parser.
Discriminative Parsing

Example for a global feature: parallel structure:

```
NP
  NP
  |  CC
  |   NP
  |   |  PP
John  IN  NN
  from Hoboken
```

Example for a global feature: non-parallel structure:

```
NP
  NP
  |  CC
  |   NP
  |   |  PP
Jim  IN  NN
  from Covina
```

Re-ranking approach to parsing:
- use a generative parser to generate the top $n$ most probable parses for an input sentence;
- evaluate each parse tree against the gold standard;
- learn a model that re-ranks the $n$-best list so that the highest scoring parse come out at the top;
- for instance, using a maximum entropy model.

We will discuss Charniak and Johnson’s (2005) approach in detail.

Features

A parse $y$ has a feature vector $f(y) = (f_1(y), \ldots, f_m(y))$. Each $f_j$ is a function that maps the parse to a number:
- $f_1(y) = \log p(y)$ is the probability assigned to $y$ by the base parser before re-ranking;
- the other features indicate tree-configurations; $f_j(y)$ is the number of times configuration $f_j$ occurs in the parse;
- features belong to feature schemas which generalize over the specifics of the features;
- schemas can be parametrized.

Feature selection: count threshold (exclude features that don’t vary on the parses of at least 5 sentences).
Example: feature scheme *Heads*, whose instances are head-head dependencies. Parameters:

- category of the least common ancestor;
- lexical or functional heads;
- lexical items or parts of speech of the head included.

Heads has 208,599 instances in total.

Other features:

- **Copar, CoLenPar**: parallelism at various levels (same POS) and difference in length for conjuncts;
- **RightBranch**: number of non-terminals on paths from root to rightmost terminal;
- **Heavy**: length of a node (number of pre-terminals dominated);
- **Neighbors**: POS of pre-terminals to the node’s left and right;
- **Rule, HeadTree**: local tree with contextual information (head annotation, grandparent annotation, etc.);
- **NGram, NGramTree**: tuples of adjacent child nodes of the same parent nodes;
- **WProj**: pre-terminals and POS of their maximal projection;
- **Word**: lexical item with POS of their immediate ancestors.

Assume that $y(s)$ is a parse of sentence $s$, and the set of $n$-best parses of $s$ is $Y(s)$. We use a maximum entropy model to compute:

$$P_\theta(y|Y) = \frac{\exp v_\theta(y)}{\sum_{y' \in Y} \exp v_\theta(y')},$$

where

$$v_\theta(y) = \theta \cdot f(y) = \sum_{j=1}^{m} \theta_j f_j(y)$$

with feature functions $f = (f_1, \ldots, f_m)$ and feature weights $\theta = (\theta_1, \ldots, \theta_m)$. The training data is $D = (s_1, \ldots, s_n)$. 

**LexFunHeads**: POS of lexical and functional head of a node.
Estimation

To estimate $\theta$, the parameters of the maximum entropy model, minimize the loss function $L_D$ relative to the training data $D$:

$$\hat{\theta} = \arg \min_{\theta} L_D(\theta) + R(\theta)$$

Here, $R$ is a regularization term that prevents the parameters from going to infinity. A quadratic regularizer is used here:

$$R(\theta) = c \sum_{j=1}^{m} (\theta_j^2)$$

where $c$ is a free parameter.

Evaluation

Standard Parseval scores are used to evaluate the re-ranker:

Compute precision and recall of labeled brackets with respect to the gold standard parses for an unseen section of the corpus. Report $F$-score, i.e., harmonic mean of precision and recall.

<table>
<thead>
<tr>
<th>Generative parser</th>
<th>$F$-score after re-ranking</th>
</tr>
</thead>
<tbody>
<tr>
<td>Charniak &amp; Johnson</td>
<td>0.9102</td>
</tr>
<tr>
<td>Collins</td>
<td>0.9037</td>
</tr>
</tbody>
</table>

Charniak and Johnson (2005) use their own parser which performs coarse-to-fine search to optimize parsing speed.

$F = 0.9102$ is very close to the state of the art for Penn Treebank parsing.

Summary

- Generative models maximize joint likelihood, discriminative models maximize conditional likelihood;
- discriminative models can be used for parse re-ranking: re-score the trees produced by a generative model;
- arbitrary features, including non-local ones, can be used in a discriminative re-ranker;
- Charniak and Johnson’s (2005) re-ranker uses a large set of features, including parallelism, heaviness, head information, and n-gram information;
- these features are combined in a maximum entropy model.
