Data description

Gideon Maillette de Buy Wenniger
University of Amsterdam
gemdbw - at - gmail - dot -com

1 Remarks about the data

1.1 Training Treebank

Binarised Wall Street Journal trees.

- The binarization nodes are marked with @ at their end e.g. VP@. The trees also have a TOP node. Before evaluation, these nodes have to be removed from the output of the parser and their children pulled up in the tree. This brings you back to the WSJ tree style

- Unaries have been pulled out for you: an unary production $X \rightarrow Y$ in the tree is now marked as a single node X%%%%%Y .
  Like the @, the %%%%% transform that I applied has to be reversed by you: transform the output of your parser before evaluation such that X%%%%%Y is put back as an unary production $X \rightarrow Y$ in your output tree. This allows evaluation of parser output in the style of the WSJ.

- The unaries $A \rightarrow B$ where both A and B are nonterminals that dominate other nonterminals have been removed from the data to avoid cycles. The case of B being a pre-terminal POS-tag like the example of $NP \rightarrow NNP$ remains in the treebank: this is a much simpler case than the general case and will not lead to cycles. That said, if you wish to simplify further, the remaining “POS tag level unaries” can also be removed (but that could harm your results later). I think that remaining two kinds of unaries (TOP level and above POS level), and the knowledge that the remaining unaries do not create chains of nonterminals longer than one level, should simplify the algorithm (no need to worry about the risk of cycles).

There is one other place were a unary production exist, which is at the top of trees (TOP $\rightarrow$ Something). See other questions and answers about this in the FAQ also.

1.2 Test Trees

Remarks:
These trees are not binarized. You are supposed to de-transform the binarization from the output of your parser by taking out all nodes labeled @ from their output trees *before* evaluation against this gold-standard.

1.3 Test sentences

All sentences from section 23 (test section) of the Penn treebank. You can choose to report results for sentences up to 15 words in length.

1.4 Evaluation

1.4.1 EvalC

This is a Graphical tool for constituency parsing evaluation, similar to EvalB (last update 25.05.2010).
Author: Federico Sangati
source: http://staff.science.uva.nl/ fsangati/

This program is written in java, is more up to date and probably easier to use then the original EvalB. (EvalB is writen in C and may give you compilation errors).

1.4.2 EVALB

This is a bracket scoring program. It reports precision, recall, F-measure, non crossing and tagging accuracy for given data.
Authors: Satoshi Sekine (New York University), Michael Collins (University of Pennsylvania)

There is a REAME file with instuctions as to how to use the program. Parameter files are also provided.
(More information about this program can be found at http://nlp.cs.nyu.edu/evalb/)

1.4.3 General remarks

Important: Please report results for ALL sentences in your test data set. If there is a parse failure, then you must output a dummy parse with just the TOP node, so that the recall for the parse is zero.