Using the Naive Bayes Method to determine a Phrase Structure Tree for E-Type Anaphora with NP Deletion

By Jacob Pawlak for Dr. Jaromczyk and Dr. Stump
CS315 | December 4th - 14th 2016

Abstract

Sentences with E-Type Anaphora, or Donkey Sentences, are a linguistic obstacle for semanticists. With NP Deletion, the replaced pronoun becomes ambiguous and makes the sentence difficult to analyze semantically. An example is as follows:

No farmer who buys a donkey can sell that donkey.
*No farmer who buys a donkey can sell it. <- Here the ‘it’ is an ambiguous pronoun

There are a few movements at the logical level (tree) that make for a correct interpretation, but these really need to be done out on paper after an inspection (before movement)

I wrote a python script that takes in a text file with multiple sentences, and then prints out the correct bracketed expression to be used in tools like Dr. Finkle and Dr. Stump’s CATS CLAW

Problem Statement

Notes on E-Type Anaphora

- Semantically Ambiguous
- Multiple Possible Syntactic Interpretations of E-Type pronouns
- Lots of Syntactic Movement at Logical Form
- Difficult to determine Phrase Structure Trees (for correct semantic interpretation)***

Notes on Naive Bayes

- Used to classify tokenized words from sentences

Importance of the Problem

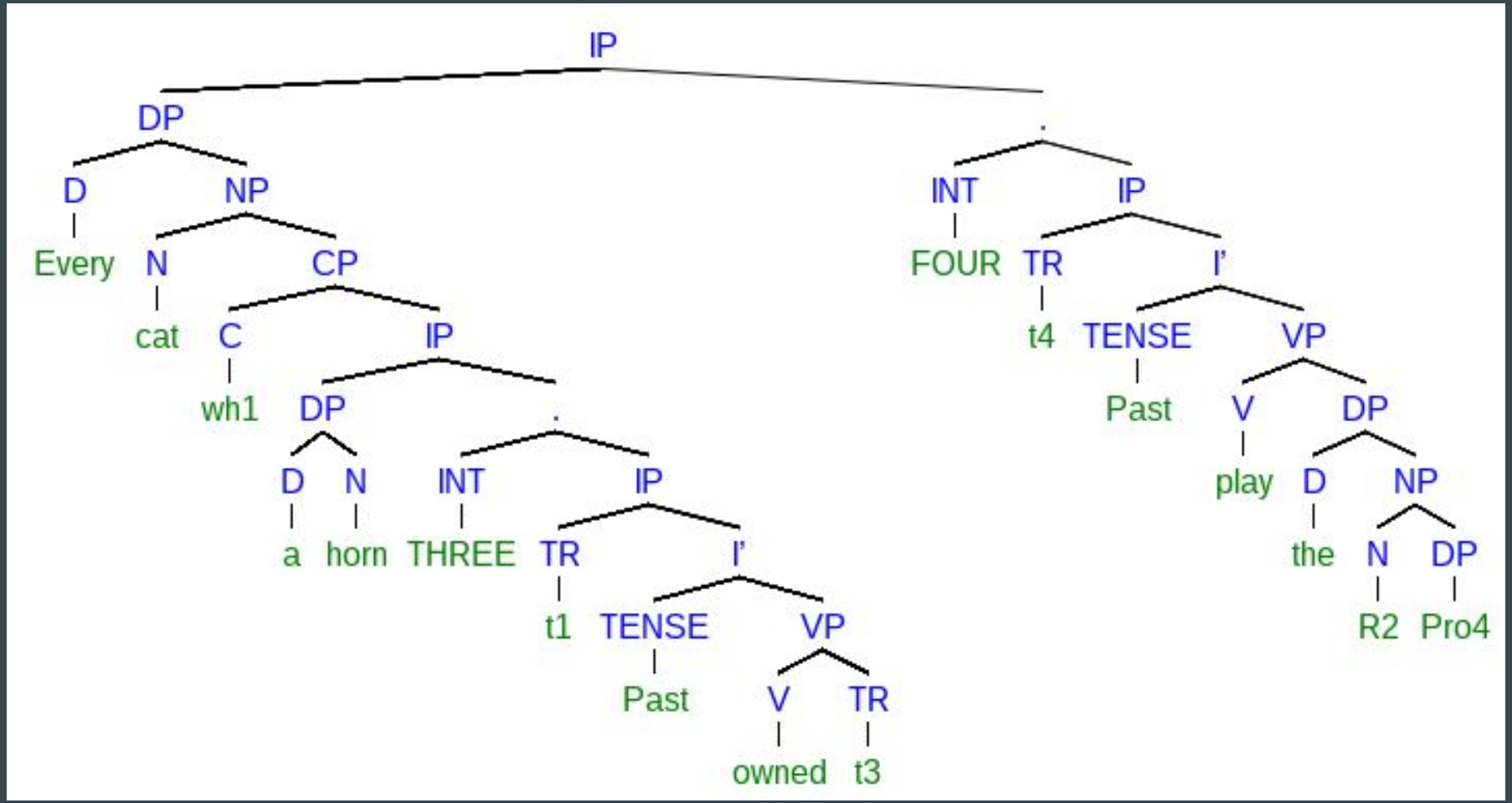
This problem of ambiguous E-Type Anaphora (donkey pronouns) lies in the fact, that while it is possible to draw a correct Phrase Structure Tree, it is much more difficult to predict the proper movement needed to produce the correct tree at logical form. This movement also involves a Cooper style analysis of E-Type Anaphora, changing the logical form tree even further.

My project adds value to the world of Computational Linguistics. It will be one of the first online tools for computing the trees for E-Type Anaphora, and it will definitely reduce time spent using drawing tools or pen and paper.

Examples Illustrating the Problem

E-Type Anaphora prove to be semantically challenging for some computers. Even if the computer has control over lexical semantics, pragmatics of the sentence will often give a different result than the actual outcome.

It may also be difficult for a computer to learn the discrete steps of the movement of Determiner Phrases (DP's - the syntactic label for the Cooper Style analysis of E-Type Anaphora). With the Naive Bayes algorithm, we hope to overcome this computational obstacle.


IP
DP
.
D
NP
INT
IP
Every
N
CP
FOUR
TR
I'
cat
C
IP
t4
TENSE
VP
wh1
DP
.
Past
V
DP
D
N
INT
IP
play
D
NP
a
horn
THREE
TR
I'
the
N
DP
t1
TENSE
VP
R2
Pro4
Past
V
TR
owned
t3

Solutions | Algorithms

Naive Bayes

Naive Bayes is a probabilistic classifier used in determining, over time, a more correct classification for the objects it computes over.
The Naive Bayes algorithm, paired with my own function for producing the bracketed expression to be used in CATS CLAW, provides a decent model for the tree building of Donkey Sentences.

Complexity

The ‘fun’ and complex part of this assignment was not implementing the algorithm, but designing the function to determine correct movement.

Notably, Donkey Sentences themselves are very complex leaving ambiguities in even the most salient context.

My Implementation

Jacob, switch to the terminal and run “python donkey.py demo.txt”

Every cat that owned a horn played it

[IP [DP [D Every] [NP [N cat] [CP [C wh1] [IP [DP [D a] [N horn]] [. [INT THREE] [IP [TR t1] [I’ [TENSE Past] [VP [V owned] [TR t3]]]]]]]] [. [INT FOUR] [IP [TR t4] [I’ [TENSE Past] [VP [V play] [DP [D the] [NP [N R2] [DP Pro4]]]]]]]

Conclusions and Closing Remarks

This program is not yet perfect or done! It still needs touching up (in the wh movement sector), and could do with some optimization for the Natural Language ToolKit library.

Questions, please!