Grammar

PROJECT 3

PREPARED FOR

Dr. Sabine Bergler

PREPARED BY

Karandeep

40104845

TABLE OF CONTENTS

Introduction:

About NLTK - NLTK aka the Natural Language Toolkit, is a suite of open source Python modules, data sets, and tutorials supporting research and development in Natural Language Processing. It contains text processing libraries for tokenization, parsing, classification, stemming, tagging and semantic reasoning. It also includes graphical demonstrations and sample data sets as well as accompanied by a cook book and a book which explains the principles behind the underlying language processing tasks that NLTK supports.

Purpose:

Hidden information often lies deep within the boundaries of what we can perceive with our eyes and our ears. Some look to data for that purpose, and most of the time, data can tell us more than we thought was imaginable. But sometimes data might not be clear cut enough to perform any sort of analytics. Language, tone, and sentence structure can explain a lot about how people are feeling, and can even be used to predict how people might feel about similar topics using a combination of the Natural Language Toolkit, a Python library used for analyzing text, and machine learning.

For this project I was to develop a Feature grammar to parse given sentences and judge the sentiment behind the sentence. The console has options to run **Sentiment Analyzer** either on a good inbuilt data or bad inbuilt data sentences to parse. Analysis of data has been done in a way that is easily comprehensible. Overall goal of the project was to create a parser and a FCFG which filters data by using NLTK access commands and creates sentence trees and visualize them in a more informative way. NLTK is not perfect and lacks in some areas which is being explored below.

Annotation of Grammar:

1. The grammar defined is a feature grammar built on NLTK feature grammar.fcfg.

Grammar.fcfg is a FCFG file where grammar is defined. Sentiment.py reads grammar.fcfg and parses sentences from it.

2. Grammar has been modified, 5 new Non-terminals are added:

Grammar is a feature context free grammar. It is parsed through Feature Earley Chart Parser to sense the sentiment of the sentence.

For Example:

Sentence: It is a compelling story.

Expected Result: Positive.

Output Tree:

```
(S[-INV]
  (NP[+WH] it)
  (VP[]
        (V[+AUX] is)
        (VP[]
        (NP[]
        (NP[] (NP[] (DT[] a)) (NP[] (ADJ[SNT='POS'] compelling)))
        (NP[+WH] story)))))
```

Result: Sentence is positive.

Grammar.fcfg takes a bottom up approach to parse the tree. It recursively parses the sentence through rules defined and structures the tree according to the rules.

It checks for words defined in grammar and matches the words and then moves to the next word. Parses until the end of the sentence is reached.

Below is the Feature Grammar used to parse sentences:

```
S[-INV] -> NP VP | NP | NP S[-INV]
S[-INV]/?x -> NP VP/?x
S[-INV] -> NP S/NP
S[-INV] -> Adv[+NEG] S[+INV]
S[+INV] -> V[+AUX] NP VP
S[+INV]/?x -> V[+AUX] NP VP/?x
```

SBar -> Comp S[-INV]
SBar/?x -> Comp S[-INV]/?x
VP -> V[SUBCAT=intrans, -AUX]
VP -> V[SUBCAT=trans, -AUX] NP
VP/?x -> V[SUBCAT=trans, -AUX] NP/?x
VP -> V[SUBCAT=clause, -AUX] SBar | VP SBar | VP SPR SBar | NP
VP/?x -> V[SUBCAT=clause, -AUX] SBar/?x
VP -> V[+AUX] VP | V[+AUX] NP | V[+AUX] | V[+AUX] Adv[+NEG] VP
VP/?x -> V[+AUX] VP/?x
V[SUBCAT=intrans, -AUX] -> 'walk' | 'sing'
V[SUBCAT=trans, -AUX] -> 'see' | 'like'
V[SUBCAT=clause, -AUX] -> 'say' | 'claim'
V[+AUX] -> 'do' | 'can' | 'is' | 'has' | 'does' | 'have' | 'making' | V[+AUX] Adv[+NEG]
NP[-WH] -> 'you'
NP[+WH] -> 'who' | 'cats' | 'story' | 'it' | 'impact' | 'example' | 'movie' | 'factor'
NP -> DT ADJ[SNT=POS] NP | ADJ[SNT=POS] NP | DT | PRP NP | NP NP |
ADJ[SNT=POS] | NP Comp
NP -> DT ADJ[SNT=NEG] NP | ADJ[SNT=NEG] NP | ADJ[SNT=NEG]
DT -> 'a' | 'the' | 'this' | 'neither'
Adv[+NEG] -> 'rarely' | 'never' | 'not'
NP/NP ->
Comp -> 'that' | 'but' | 'and' | 'nor'
PRP -> 'with' | 'of' | 'in'
SPR -> ','