Using Compositional Semantics: Analyzing the sentences

We have auto generated Context free Grammar using Tree productions and then we have implemented semantic attachments attached to it. So that we were actually able to generate FCFG file which gets linked to our program.

Sample FCFG grammar:

```
% start S

S[SEM=(?np + WHERE + ?vp)] -> NP[SEM=?np] VP[SEM=?vp]

VP[SEM=(?v + ?pp)] -> IV[SEM=?v] PP[SEM=?pp]

VP[SEM=(?v + ?ap)] -> IV[SEM=?v] AP[SEM=?ap]

NP[SEM=(?det + ?n)] -> Det[SEM=?det] N[SEM=?n]

PP[SEM=(?p + ?np)] -> P[SEM=?p] NP[SEM=?np]

AP[SEM=?pp] -> A[SEM=?a] PP[SEM=?pp]

NP[SEM='Country="greece"'] -> 'Greece'

NP[SEM='Country="china"'] -> 'China'

Det[SEM='SELECT'] -> 'Which' | 'What'

N[SEM='City FROM city_table'] -> 'cities'

IV[SEM=''] -> 'are'

A[SEM=''] -> 'located'

P[SEM=''] -> 'in'
```

Sample program to execute such type of file:

```
from nltk import load_parser

>>> cp = load_parser('grammars/book_grammars/sql0.fcfg')

>>> query = 'What cities are located in China'

>>> trees = list(cp.parse(query.split()))

>>> answer = trees[0].label()['SEM']

>>> answer = [s for s in answer if s]

>>> q = ' '.join(answer)

>>> print(q)
```

Sample output from that was generated such type of files:

SELECT City FROM city_table WHERE Country="china"

Through the above query we can generate our answer for the given question. In this way given sentences can be analyzed through compositional semantics.

So everytime the file will be generated, through that grammar file and finally output will be generated in string format which is like SQL query.

Through by using SQL query. We used to fetch the data from Database and we have displayed the output.

There were mainly three types of sentences. Those are Classified as follows like verb “to be” questions, auxiliary “do” questions and Wh- questions and we do have three Databases to pass the SQL query into they are Music, Movies and Geography. At first, we have categorized where the given question belong to like whether it goes to Music database, Movie Database or Geography Database and then for those particular questions which rules need to be embedded in those categories.

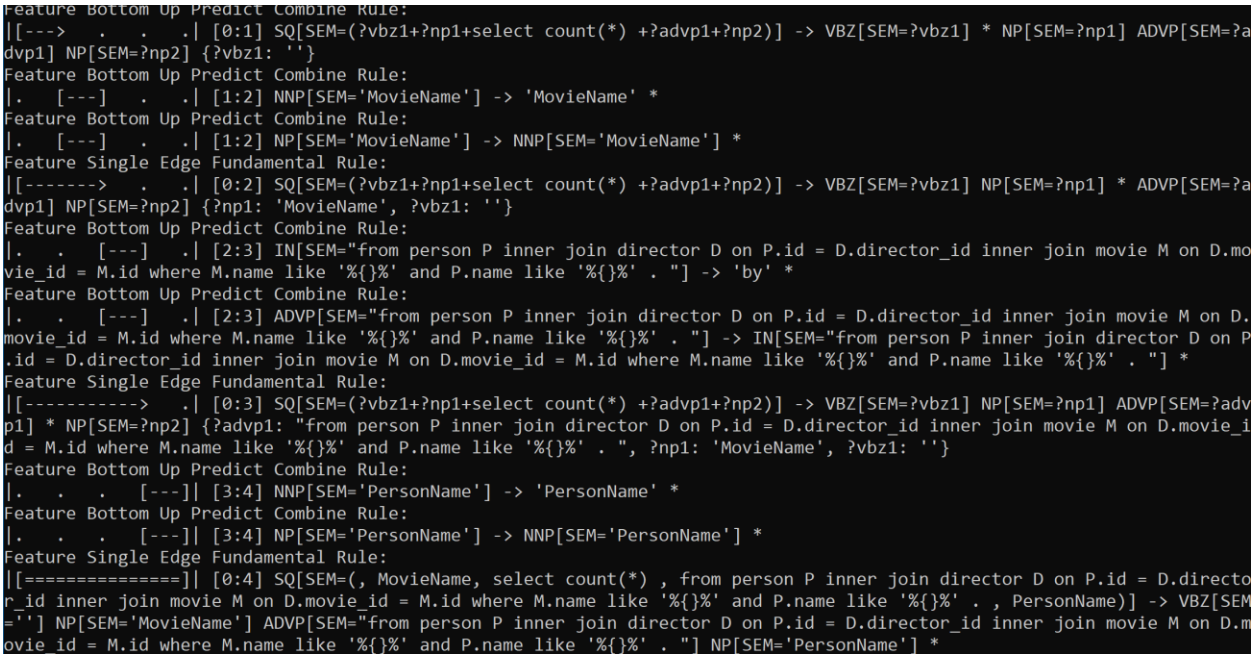
Verb: “to be” questions

The above mentioned questions usually start with Is/Was . If questions start with Is/Was, we have actually embedded “**from statements” in the production of “VERB”** like “VBN”,”VB”,”VBP” etc.

“Select statements” in the production of main clause are embedded with **“Select” + “count(*)”.**

Through the concept of lambda calculus we have implemented this technique and finally rule by rule and compositionally it will form a tree and at last the query will be obtained .

Sample tree in the output:



In the above screenshot it will form a query compositionally as you

Where statements will get replaced when there is occurrence of propernouns and or else some of the normal nouns(NN's) which were given in the rules accordingly.

Few situations where we struggled in verb: "to be" like "ADVP" Phrase where we have embedded "From statements ".

Sample output's for the given questions (verb : "to be"):

The parsed trees for the input are as follows

<QUESTION> Was Beyonce born in Italy?<QUERY>

SELECT COUNT(*) FROM ARTIST A WHERE A.NAME LIKE '%ARTISTNAME%' AND A.PLACEOFBITH LIKE '%ITALY%'

No

<QUESTION> Is Rome the capital of Italy?<QUERY>

SELECT COUNT(*) FROM CITIES CI INNER JOIN CAPITALS CA ON CI.ID = CA.CITYID INNER JOIN COUNTRIES CY ON CA.COUNTRYID = CY.ID WHERE CI.NAME LIKE '%ROME%' AND CY.NAME LIKE '%ITALY%'

Yes

<QUESTION> Is France in Europe?<QUERY>

SELECT COUNT(*) FROM COUNTRIES CY INNER JOIN COUNTRYCONTINENTS CC ON CY.ID = CC.COUNTRYID INNER JOIN CONTINENTS CO ON CC.CONTINENTID = CO.ID WHERE CY.NAME LIKE '%FRANCE%' AND CO.CONTINENT LIKE '%EUROPE%'

Yes

<QUESTION> Is Kubrick a director?<QUERY>

SELECT COUNT(*) FROM PERSON P INNER JOIN DIRECTOR D ON P.ID = D.DIRECTOR_ID WHERE P.NAME LIKE "%KUBRICK%"

Yes

<QUESTION> Is Mighty Aphrodite by Allen?<QUERY>

SELECT COUNT(*) FROM PERSON P INNER JOIN DIRECTOR D ON P.ID = D.DIRECTOR_ID INNER JOIN MOVIE M ON D.MOVIE_ID = M.ID WHERE M.NAME LIKE "%MIGHTY APHRODITE%" AND P.NAME LIKE "%ALLEN%"

Yes

<QUESTION> Was Birdman the best movie in 2015?<QUERY>

SELECT COUNT(*) FROM MOVIE M INNER JOIN OSCAR O ON M.ID = O.MOVIE_ID WHERE M.NAME LIKE "%BIRDMAN%" AND O.YEAR LIKE "%2015%"

Yes

<QUESTION> Was Loren born in Italy?<QUERY>

SELECT COUNT(*) FROM PERSON P WHERE P.NAME LIKE "%LOREN%" AND P.POB LIKE "%ITALY%"

Yes

Auxiliary "Do" questions:

The above mentioned questions usually start with "Do, Did, Does" . If questions start with such auxiliary, we have actually embedded "**from statements" in the production of "VERB"** like "VBN","VB","VBP" etc.

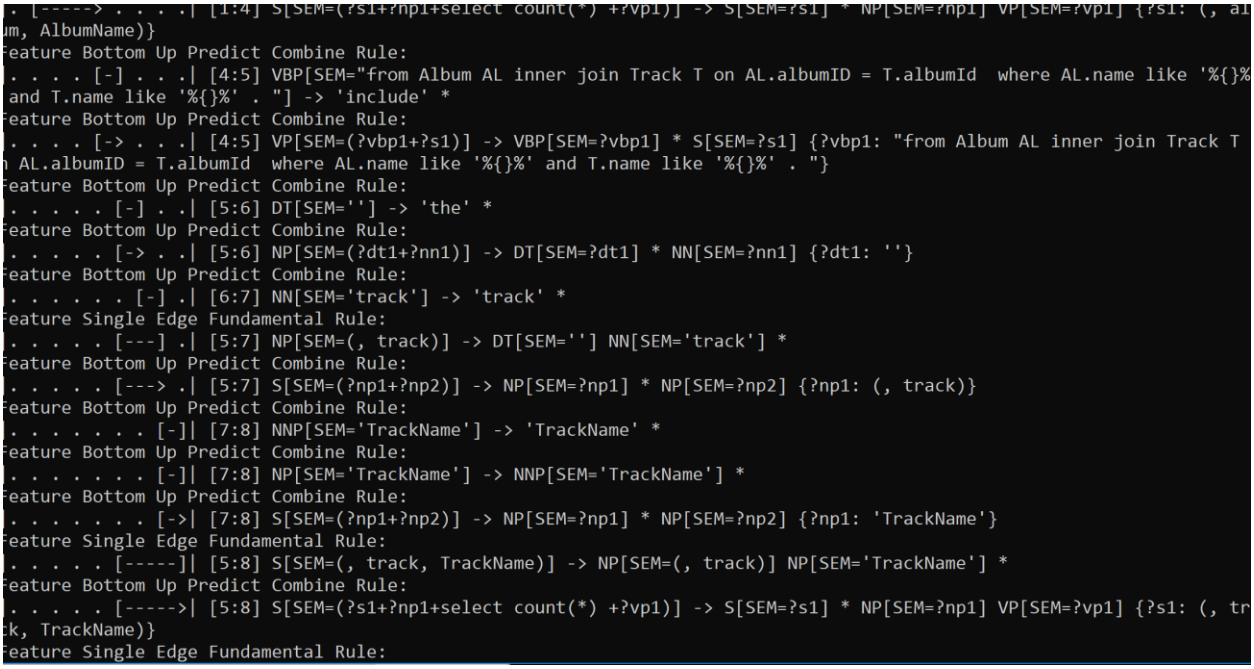
"Select statements" in the production of main clause are embedded with **"Select" + "count(*)".**

Through the concept of lambda calculus we have implemented this technique and finally rule by rule and compositionally it will form a tree and at last the query will be obtained .

Few cases where we have struggled are like when there is occurrence of JJS we can't actually put it in the stopword like "best" then we have taken there as bestcount and oscarcount to resolve such problems.

Sample tree in the output:

```
Does the album AlbumName include the track TrackName?
|.D.t.a.A.i.t.t.T.|
Leaf Init Rule:
|[-] . . . . . .| [0:1] 'Does'
|. [-] . . . . .| [1:2] 'the'
|. . [-] . . . .| [2:3] 'album'
|. . . [-] . . .| [3:4] 'AlbumName'
|. . . . [-] . .| [4:5] 'include'
|. . . . . [-] .| [5:6] 'the'
|. . . . . [-] .| [6:7] 'track'
|. . . . . [-]| [7:8] 'TrackName'
Feature Bottom Up Predict Combine Rule:
|[-] . . . . . .| [0:1] VBZ[SEM='Does'] -> 'Does' *
Feature Bottom Up Predict Combine Rule:
|[-> . . . . . .| [0:1] VP[SEM=(?vbz1+?np1)] -> VBZ[SEM=?vbz1] * NP[SEM=?np1] {?vbz1: 'Does'}
Feature Bottom Up Predict Combine Rule:
|. [-] . . . . .| [1:2] DT[SEM=''] -> 'the' *
Feature Bottom Up Predict Combine Rule:
|. [-> . . . . .| [1:2] NP[SEM=(?dt1+?nn1)] -> DT[SEM=?dt1] * NN[SEM=?nn1] {?dt1: ''}
Feature Bottom Up Predict Combine Rule:
|. . [-] . . . .| [2:3] NN[SEM='album'] -> 'album' *
Feature Single Edge Fundamental Rule:
|. [---] . . . .| [1:3] NP[SEM=(, album)] -> DT[SEM=''] NN[SEM='album'] *
Feature Bottom Up Predict Combine Rule:
|. [---> . . . .| [1:3] S[SEM=(?np1+?np2)] -> NP[SEM=?np1] * NP[SEM=?np2] {?np1: (, album)}
Feature Single Edge Fundamental Rule:
|[-----] . . . .| [0:3] VP[SEM=(Does, , album)] -> VBZ[SEM='Does'] NP[SEM=(, album)] *
Feature Bottom Up Predict Combine Rule:
```



Sample output's for the given questions (Auxiliary : do):

<QUESTION> Did Neeson star in Schindler's List?<QUERY>

SELECT COUNT(*) FROM PERSON P INNER JOIN ACTOR A ON P.ID = A.ACTOR_ID INNER JOIN MOVIE M ON A.ACTOR_ID = M.ID WHERE P.NAME LIKE "%NEESON%" AND M.NAME LIKE "%SCHINDLER'S LIST%"

No

<QUESTION> Did Swank win the oscar in 2000?<QUERY>

SELECT COUNT(*) FROM PERSON P INNER JOIN OSCAR O ON P.ID = O.PERSON_ID WHERE P.NAME LIKE "%SWANK%" AND O.YEAR LIKE "%2000%"

Yes

<QUESTION> Did a French actor win the oscar in 2012?<QUERY>

SELECT COUNT(*) FROM PERSON P INNER JOIN OSCAR O ON P.ID = O.PERSON_ID WHERE P.POB LIKE '%FRANCE%' AND O.TYPE LIKE '%ACTOR%' AND O.YEAR LIKE "%2012%"

Yes

<QUESTION> Did a movie with Neeson win the oscar for best film?<QUERY>

SELECT COUNT(*) FROM PERSON P INNER JOIN DIRECTOR D ON P.ID = D.DIRECTOR_ID INNER JOIN OSCAR O ON D.MOVIE_ID = O.MOVIE_ID WHERE P.NAME LIKE "%NEESON%" AND O.TYPE LIKE '%PICTURE%'

No

<QUESTION> Did Madonna sing Papa Do not Preach?<QUERY>

SELECT COUNT(*) FROM ARTIST A INNER JOIN ALBUM AL ON A.ID = AL.ARTSITID INNER JOIN TRACK T ON AL.ALBUMID = T.ALBUMID WHERE A.NAME LIKE '%ARTISTNAME%' AND T.NAME LIKE '%TRACKNAME%'

No

<QUESTION> Does the album Thriller include the track Beat It?<QUERY>

SELECT COUNT(*) FROM ALBUM AL INNER JOIN TRACK T ON AL.ALBUMID = T.ALBUMID WHERE AL.NAME LIKE '%DOES%' AND T.NAME LIKE '%ALBUM%'

No

Wh-questions:

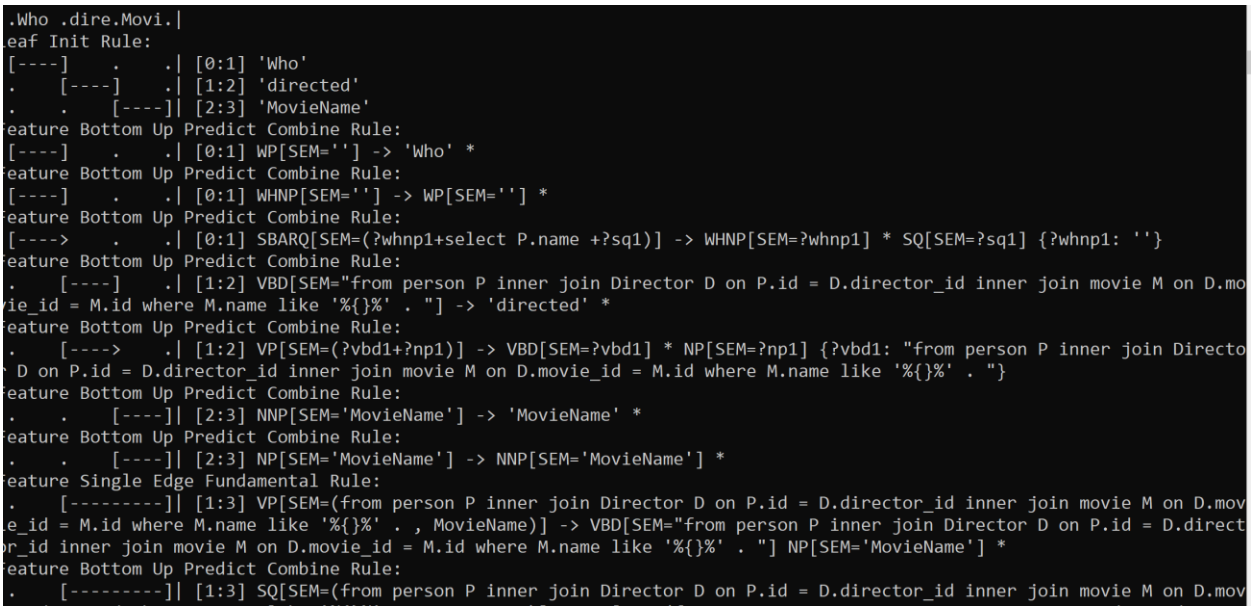
The above mentioned questions usually start with “Wh” . If questions start with such type, we have actually embedded “**from statements” in the production of “VERB”** like “VBN”,”VB”,”VBP” etc .Then we have clearly mentioned conditions for win + prize and win + category + prize. The code which we have written works for the above cases.

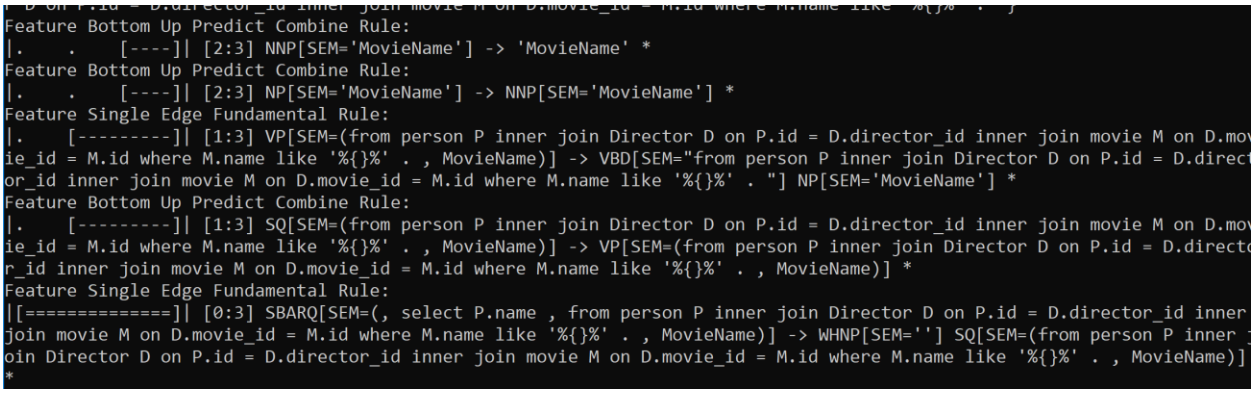
“Select statements” in the production of main clause are embedded with **“Select” + “count(*)”.**

Through the concept of lambda calculus we have implemented this technique and finally rule by rule and compositionally it will form a tree and at last the query will be obtained .

Few cases where we have struggled are when there is subtree inside the tree then establishing conditions for such cases and where to insert “select statements” and “from statements” made us little tricky.

Sample output of the tree:





Sample output's for the given questions (Wh-questions):

<QUESTION> Who directed Schindler's List?<QUERY>

SELECT P.NAME FROM PERSON P INNER JOIN DIRECTOR D ON P.ID = D.DIRECTOR_ID INNER JOIN MOVIE M ON D.MOVIE_ID = M.ID WHERE M.NAME LIKE "%SCHINDLER'S LIST%"

Steven Spielberg

<QUESTION> Who won the oscar for best actor in 2005?<QUERY>

SELECT P.NAME FROM PERSON P INNER JOIN OSCAR O ON P.ID = O.PERSON_ID WHERE O.YEAR LIKE "%2005%" AND O.TYPE LIKE '%BEST-ACTOR%'

Jamie Foxx

<QUESTION> Which movie won the oscar in 2000?<QUERY>

SELECT M.NAME FROM MOVIE M INNER JOIN OSCAR O ON M.ID = O.PERSON_ID WHERE O.TYPE LIKE '%PICTURE%' AND O.YEAR LIKE "%2000%"

None

<QUESTION> When did Blanchett win an oscar for best actress?<QUERY>

SELECT O.YEAR FROM OSCAR O INNER JOIN PERSON P ON O.PERSON_ID = P.ID WHERE O.TYPE LIKE '%BEST-ACTRESS%' AND P.NAME LIKE "%BLANCHETT%"

Yes

<QUESTION> Which American actress won the oscar in 2012?<QUERY>

SELECT P.NAME FROM PERSON P INNER JOIN OSCAR O ON P.ID = O.PERSON_ID WHERE P.POB LIKE '%USA%' AND O.TYPE LIKE '%BEST-ACTRESS%' AND O.YEAR LIKE "%2012%"

Meryl Streep

<QUESTION> Who directed the best movie in 2010?<QUERY>

SELECT P.NAME FROM PERSON P INNER JOIN DIRECTOR D ON P.ID = D.DIRECTOR_ID INNER JOIN OSCAR O ON D.MOVIE_ID = O.MOVIE_ID WHERE O.YEAR LIKE "%2010%" AND O.TYPE LIKE '%PICTURE%'

Kathryn Bigelow

<QUESTION> What is the capital of Italy?<QUERY>

SELECT CI.NAME FROM CITIES CI INNER JOIN CAPITALS CA ON CI.ID = CA.CITYID INNER JOIN COUNTRIES CY ON CA.COUNTRYID = CY.ID WHERE CY.NAME LIKE '%ITALY%'

Rome

<QUESTION> Where was Gaga born?<QUERY>

SELECT A.PLACEOFBITH FROM ARTIST A WHERE A.NAME LIKE '%GAGA%'

Manhattan, NY