## WordNet Summary

WordNet is a project that was started in the 1980's by a psychologist who wanted to understaned hwo humans heirarchicaly organize concepts. It gives developers the ability to view the heirarchical organization of nouns, verbs, adjectives and more.

```
import nltk
import math
nltk.download('wordnet')
nltk.download('omw-1.4')
nltk.download('sentiwordnet')
nltk.download('gutenberg')
nltk.download('genesis')
nltk.download('inaugural')
nltk.download('nps_chat')
nltk.download('webtext')
nltk.download('treebank')
nltk.download('stopwords')
from nltk.corpus import wordnet as wn
from nltk.wsd import lesk
from nltk.corpus import sentiwordnet as swn
from nltk.book import text4
```

```
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package omw-1.4 to /root/nltk_data...
[nltk_data]   Package omw-1.4 is already up-to-date!
[nltk_data] Downloading package sentiwordnet to /root/nltk_data...
[nltk_data]   Package sentiwordnet is already up-to-date!
[nltk_data] Downloading package gutenberg to /root/nltk_data...
[nltk_data]   Package gutenberg is already up-to-date!
[nltk_data] Downloading package genesis to /root/nltk_data...
[nltk_data]   Package genesis is already up-to-date!
[nltk_data] Downloading package inaugural to /root/nltk_data...
[nltk_data]   Package inaugural is already up-to-date!
[nltk_data] Downloading package nps_chat to /root/nltk_data...
[nltk_data]   Package nps_chat is already up-to-date!
[nltk_data] Downloading package webtext to /root/nltk_data...
[nltk_data]   Package webtext is already up-to-date!
[nltk_data] Downloading package treebank to /root/nltk_data...
[nltk_data]   Package treebank is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
wn.synsets('house')
```

```
[Synset('house.n.01'),
 Synset('firm.n.01'),
 Synset('house.n.03'),
 Synset('house.n.04'),
 Synset('house.n.05'),
 Synset('house.n.06'),
 Synset('house.n.07'),
 Synset('sign_of_the_zodiac.n.01'),
 Synset('house.n.09'),
 Synset('family.n.01'),
 Synset('theater.n.01'),
 Synset('house.n.12'),
 Synset('house.v.01'),
 Synset('house.v.02')]
```

```
wn.synset('house.n.01').definition()
```

```
'a dwelling that serves as living quarters for one or more families'
```

```
wn.synset('house.n.01').examples()
```

```
['he has a house on Cape Cod', 'she felt she had to get out of the house']
```

```
wn.synset('house.n.01').lemmas()
```

```
[Lemma('house.n.01.house')]
```

```
house = wn.synset('house.n.01')
hyp = house.hypernyms()[0]
top = wn.synset('entity.n.01')
while hyp:
  print(hyp)
  if hyp == top:
    break
  if hyp.hypernyms():
    hyp = hyp.hypernyms()[0]
```

```
    Synset('building.n.01')
    Synset('structure.n.01')
    Synset('artifact.n.01')
    Synset('whole.n.02')
    Synset('object.n.01')
    Synset('physical_entity.n.01')
    Synset('entity.n.01')
```

We can see here based off the heirarchy that the noun gets more general the furter up you get and it stops at the base case for all nouns which is an entity.

```
print(wn.synset('house.n.01').hypernyms())
print(wn.synset('house.n.01').hyponyms())
print(wn.synset('house.n.01').part_meronyms())
print(wn.synset('house.n.01').part_holonyms())
print(wn.synset('house.n.01').lemmas()[0].antonyms())
```

```
    [Synset('building.n.01'), Synset('dwelling.n.01')]
    [Synset('beach_house.n.01'), Synset('boarding_house.n.01'), Synset('bungalow.n.01'), Synset('cabin.n.02'), Synset('chalet.n.01'), Synset
    [Synset('library.n.01'), Synset('loft.n.02'), Synset('porch.n.01'), Synset('study.n.05')]
    []
    []
```

```
print(wn.synsets('play'))
```

```
    [Synset('play.n.01'), Synset('play.n.02'), Synset('play.n.03'), Synset('maneuver.n.03'), Synset('play.n.05'), Synset('play.n.06'), Synse
```

```
wn.synset('act.v.03').definition()
```

```
    'play a role or part'
```

```
wn.synset('act.v.03').examples()
```

```
    ['Gielgud played Hamlet',
     'She wants to act Lady Macbeth, but she is too young for the role',
     "She played the servant to her husband's master"]
```

```
wn.synset('act.v.03').lemmas()
```

```
    [Lemma('act.v.03.act'), Lemma('act.v.03.play'), Lemma('act.v.03.represent')]
```

```
play = wn.synset('act.v.03')
hyp_set = play.hypernyms()
while len(hyp_set) > 0:
  print(hyp_set[0])
  hyp_set = hyp_set[0].hypernyms()
```

```
    Synset('re-create.v.01')
    Synset('make.v.03')
```

The verbs dont have much of a heirarchy as we can see as compared to nouns. Verba are an action and generalizes to 'be'.

```
print(wn.morphy('act'))
```

```
    act
```

```
firstWord = wn.synset('run.v.01')
secondWord = wn.synset('jog.v.01')
print('Wu Palmer Similarity is ' + str(wn.wup_similarity(firstWord,secondWord)))
print('The Lesk algorithm for these two words are: ')
print(lesk(secondWord.definition(), 'run'))
print(lesk(firstWord.definition(), 'jog'))
```

```
     Wu Palmer Similarity is 0.18181818181818182
     The Lesk algorithm for these two words are:
     Synset('tend.v.01')
     Synset('trot.v.01')
```

Wu Palmer tries to get the similarity betweeen the words, while the Lesk algorithm attempts to look at the context and get the correct synset from that.

SentiWordNet is able to detect the tone of text and find emotinally charged words through lexical analysis based on the context of the sentence and the definition of the given word. This could be used on social media platforms to detect bullying or violent wording, for example.

```
word = swn.senti_synset('disgust.n.01')
print(word)
print('Positive Score = ', word.pos_score())
print('Negative Score = ', word.neg_score())
print('Objective Score = ', word.obj_score())
sentence = 'The man was filled with disgust as he saw the pickpotter running away'
tokens = sentence.split()
for token in tokens:
  syn_list = list(swn.senti_synsets(token))
  if syn_list:
    syn = syn_list[0]
    print(token + ' ' + 'Pos score = ' + str(syn.pos_score()) + ' Negative score = ' + str(syn.neg_score()))
```

```
     <disgust.n.01: PosScore=0.375 NegScore=0.375>
     Positive Score =  0.375
     Negative Score =  0.375
     Objective Score =  0.25
     man Pos score = 0.0 Negative score = 0.0
     was Pos score = 0.0 Negative score = 0.0
     filled Pos score = 0.0 Negative score = 0.0
     disgust Pos score = 0.375 Negative score = 0.375
     as Pos score = 0.0 Negative score = 0.0
     he Pos score = 0.0 Negative score = 0.0
     saw Pos score = 0.0 Negative score = 0.0
     running Pos score = 0.0 Negative score = 0.0
     away Pos score = 0.0 Negative score = 0.0
```

These scores are able to tell the true meaning of the sentence and whether or not each word has a positive or negative tone in the sentence. It allows the program to understand the tone of the sentence and not just what it means.

Collocations are essentially phrases within a sentence that happen more often than not within a text that you cannot substitute with other words to get the same meaning.

```
text4.collocations()
```

```
     United States; fellow citizens; years ago; four years; Federal
     Government; General Government; American people; Vice President; God
     bless; Chief Justice; one another; fellow Americans; Old World;
     Almighty God; Fellow citizens; Chief Magistrate; every citizen; Indian
     tribes; public debt; foreign nations
```

```
text = ' '.join(text4.tokens)
collocation = 'Fellow Citizens'
vocab = len(set(text4))
hg = text.count('Fellow Citizens')/vocab
print("p(Fellow Citizens) = ",hg )
h = text.count('Fellow')/vocab
print("p(Fellow) = ", h)
g = text.count('Citizens')/vocab
print('p(Citizens) = ', g)
pmi = math.log2(hg / (h * g))
print('pmi = ', pmi)
```

```
     p(Fellow Citizens) =  9.975062344139652e-05
     p(Fellow) =  0.002394014962593516
```

```
p(Citizens) =  0.0006982543640897755
pmi =  5.898997193450885
```

Based on the Mutual Information of Fellow Citizens, it is likley that this is a collocation with a high score like that. This collocation is likely significant in the context and cant be replaced with other words so easily.

✓ 0s completed at 3:59 PM