LING/C SC 581:
Advanced Computational Linguistics

Lecture 14

Today's Topics

- Homework 6 Review
- Stanford **CoreNLP**
  - online use (*graphical output*)
  - use with programs (*textual output*)
- Stanford **Stanza**
  - Deep Learning parser etc., but also can provide access to CoreNLP
  - interfacing with Python

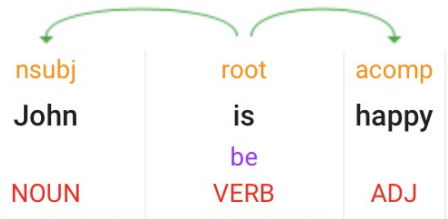
Homework 6 review

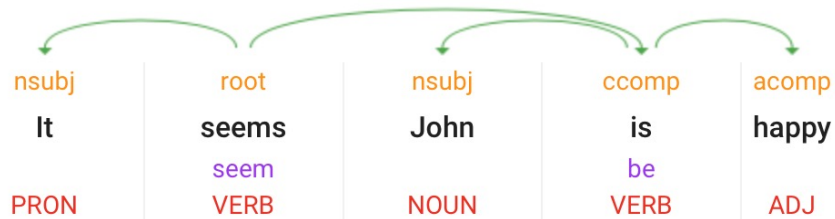
Question 1

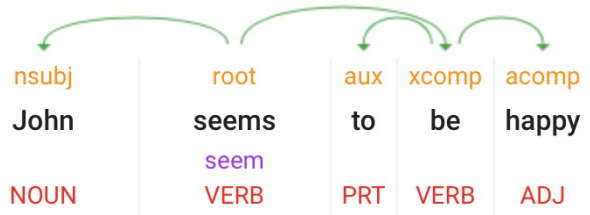
- **Raising verb *seems***:
  - John is happy
  - John *seems* happy
  - It *seems* John is happy
  - It *seems* that John is happy
  - John *seems* to be happy

- Assume the SP unlabeled relations are labeled with the *most likely* dependency labels

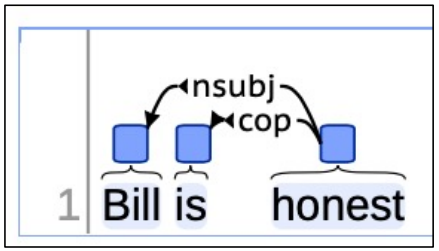
Homework 6: Question 1 Review





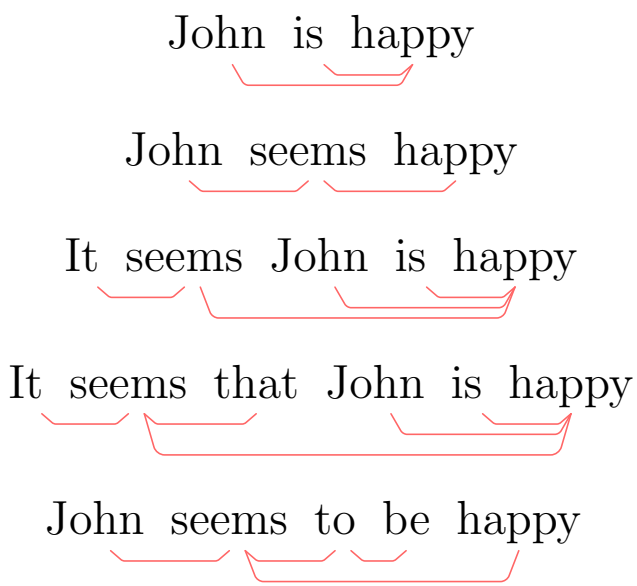


UD:



A cop (copula) is the relation of a function word used to link a subject to a nonverbal predicate

Homework 6: Question 1 Review



- Differences:
  - *John* link *happy* vs. *is*
  - **none**
  - *seems* link *happy* vs. *is*
  - *seems* link *that*, *that* link *is*
  - *John* link *happy* vs. *is*
  - *to* link *seems vs. be*
  - *seems* link *happy* vs. *be*

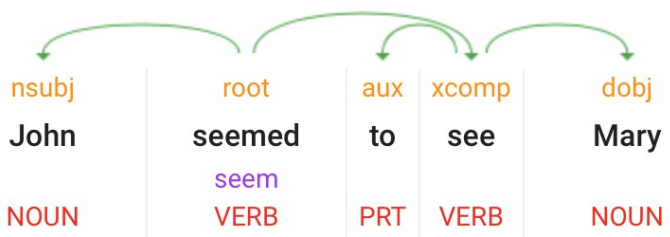
Homework 6 Review

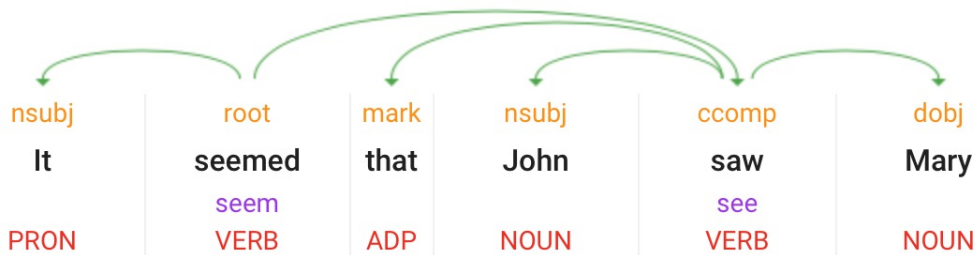
Question 2

- suppose the lower clause is:
  - John saw Mary
- and the raising predicate is *seemed*:
  - John seemed to see Mary
  - It seems (that) John saw Mary

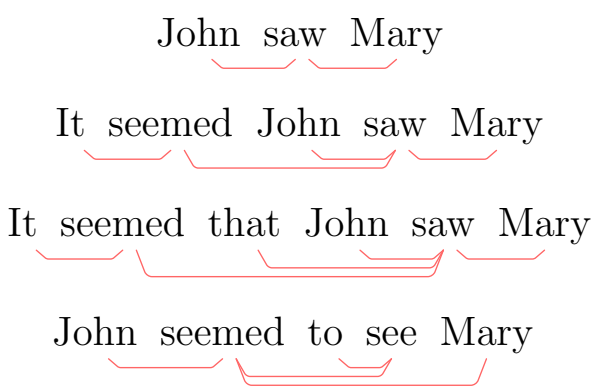
- Does the SP program do better on these examples?
  - John saw Mary
  - John *seems* saw Mary (cf. Q1)
  - It *seems* John saw Mary
  - It *seems* that John saw Mary
  - John seemed to see Mary

Homework 6: Question 2 Review





Homework 6: Question 2 Review



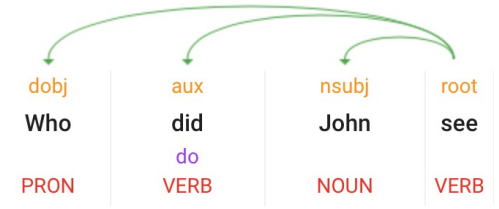
- Differences:
  - **none**
  - **none**
  - **none**
  - **seemed … Mary**

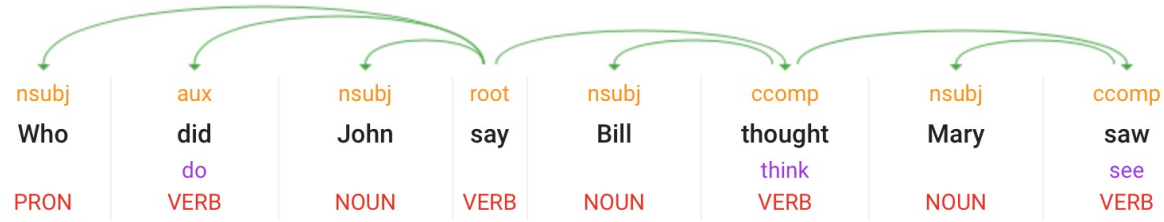
Homework 6: Question 3 Review

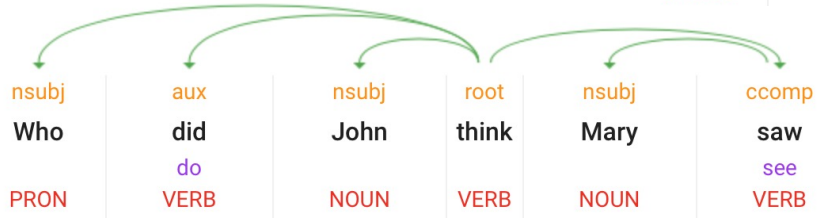
Question 3

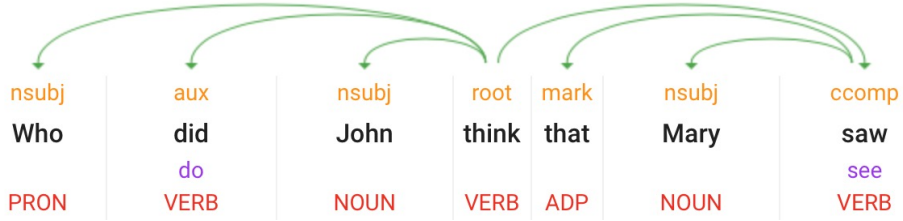
- Consider object *wh*-question formation:
  - **Who** did John **see**
  - **Who** did John think (that) Mary **saw**
  - **Who** did John (that) say Bill (that) thought Mary **saw**

Homework 6: Question 3 Review

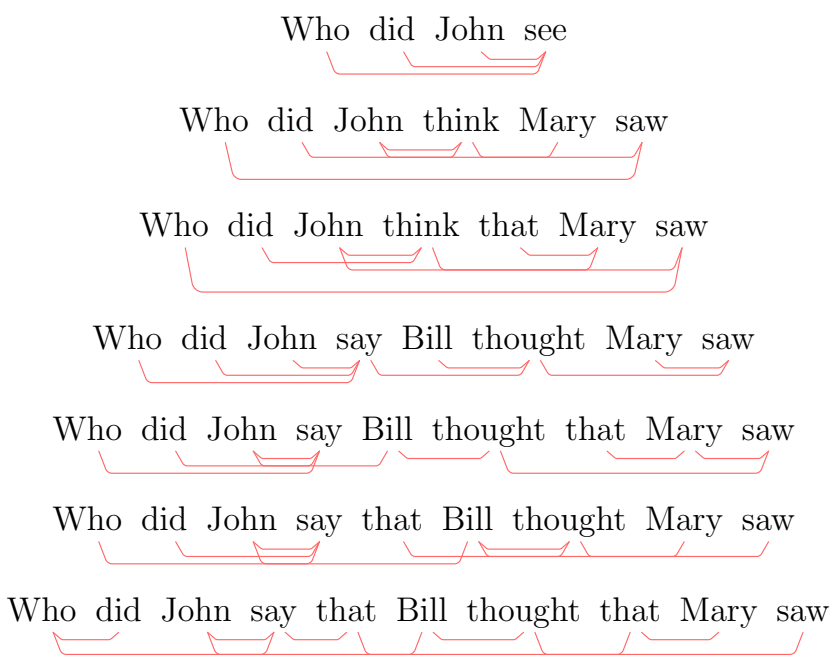








Homework 6: Question 3 Review

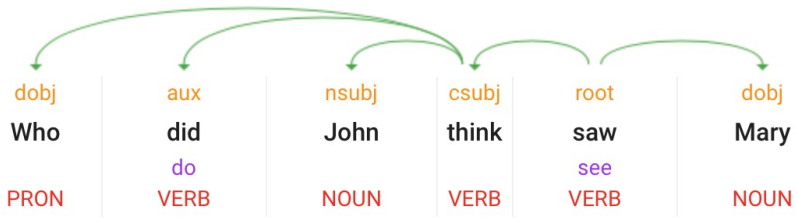


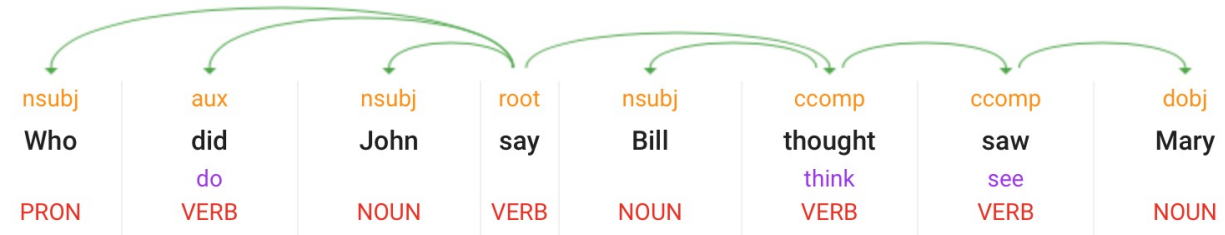
Homework 6

Question 4

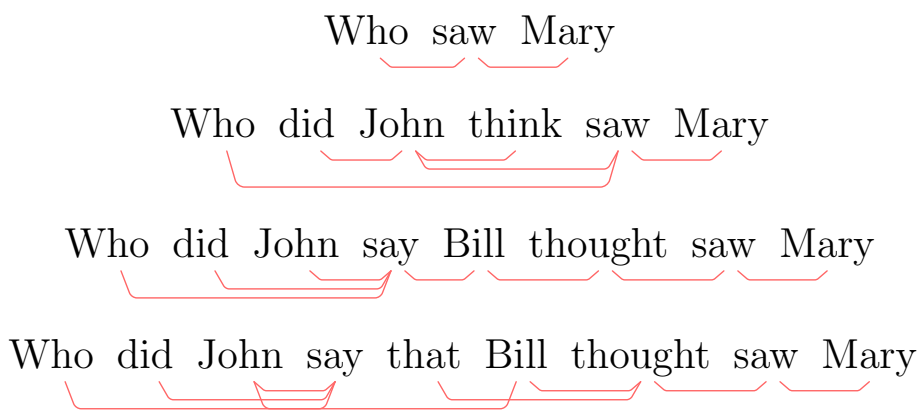
- Consider subject *wh*-question formation:
  - **Who saw** Mary
  - **Who** did John think **saw** Mary
  - **Who** did John say (that) Bill thought **saw** Mary

Homework 6: Question 4 Review





Homework 6: Question 4 Review



- Differences:
  - **None**
  - **Who … saw**
  - **Who … Bill**

Stanford CoreNLP

- Java-based
- "**CoreNLP** enables users to derive linguistic annotations for text, including token and sentence boundaries, parts of speech, named entities, numeric and time values, **dependency and constituency parses**, coreference, sentiment, quote attributions, and relations."
- "CoreNLP currently supports 8 languages: Arabic, Chinese, English, French, German, Hungarian, Italian, and Spanish."
- Run it online or download and run it from your own machine.
- URL:
  - https://stanfordnlp.github.io/CoreNLP/index.html

Java 8

- In JDK 8 and JRE 8, the version strings are 1.8 and 1.8.0.
- Example:

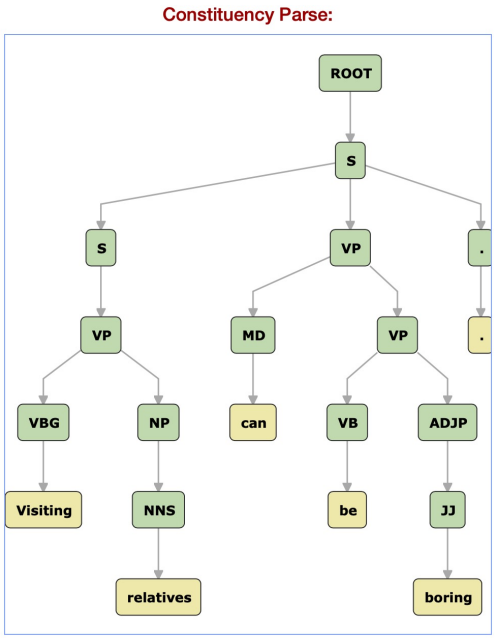
```
java -version
java version "1.8.0_191"
Java(TM) SE Runtime Environment (build 1.8.0_191-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.191-b12, mixed mode)
```

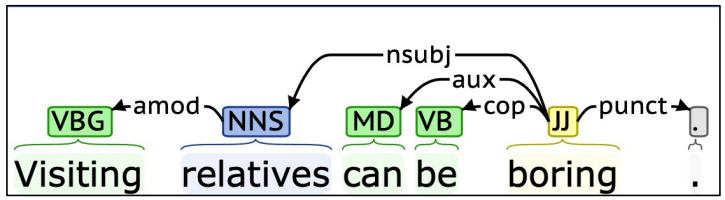
CoreNLP Online

- URL:
  - https://corenlp.run

CoreNLP Online

Visiting relatives can be boring.



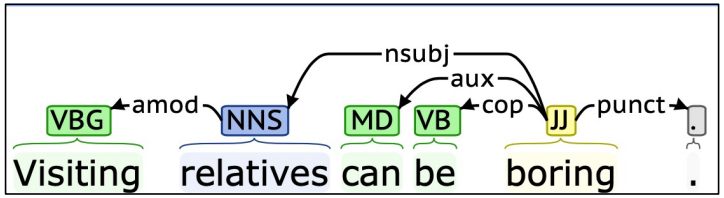


CoreNLP: command line

```
(base) stanford-corenlp-4.4.0$ java -cp "*" edu.stanford.nlp.pipeline.StanfordCoreNLP -file input2.txt

[main] INFO edu.stanford.nlp.pipeline.StanfordCoreNLP - Searching for resource:
StanfordCoreNLP.properties ... found.

…

[main] INFO edu.stanford.nlp.tagger.maxent.MaxentTagger - Loading POS tagger from
edu/stanford/nlp/models/pos-tagger/english-left3words-distsim.tagger ... done [0.4 sec].

…

[main] INFO edu.stanford.nlp.parser.nndep.DependencyParser - Initializing dependency parser ... done
[1.9 sec].

…

Processing file /Users/sandiway/Downloads/stanford-corenlp-4.4.0/input2.txt ... writing to
/Users/sandiway/Downloads/stanford-corenlp-4.4.0/input2.txt.out

Annotating file /Users/sandiway/Downloads/stanford-corenlp-4.4.0/input2.txt ... done [0.3 sec].

Annotation pipeline timing information:

TOTAL: 0.3 sec. for 6 tokens at 17.9 tokens/sec.

Pipeline setup: 21.5 sec.

Total time for StanfordCoreNLP pipeline: 21.8 sec.
```

CoreNLP: command line

```
Document: ID=input2.txt (1 sentences, 6 tokens)

Sentence #1 (6 tokens):
Visiting relatives can be boring.

Tokens:
[Text=Visiting CharacterOffsetBegin=0 CharacterOffsetEnd=8 PartOfSpeech=VBG Lemma=visit NamedEntityTag=O]
[Text=relatives CharacterOffsetBegin=9 CharacterOffsetEnd=18 PartOfSpeech=NNS Lemma=relative NamedEntityTag=O]
[Text=can CharacterOffsetBegin=19 CharacterOffsetEnd=22 PartOfSpeech=MD Lemma=can NamedEntityTag=O]
[Text=be CharacterOffsetBegin=23 CharacterOffsetEnd=25 PartOfSpeech=VB Lemma=be NamedEntityTag=O]
[Text=boring CharacterOffsetBegin=26 CharacterOffsetEnd=32 PartOfSpeech=JJ Lemma=boring NamedEntityTag=O]
[Text=. CharacterOffsetBegin=32 CharacterOffsetEnd=33 PartOfSpeech=. Lemma=. NamedEntityTag=O]
```




```
Dependency Parse (enhanced plus plus dependencies):
root(ROOT-0, boring-5)
amod(relatives-2, Visiting-1)
nsubj(boring-5, relatives-2)
aux(boring-5, can-3)
cop(boring-5, be-4)
punct(boring-5, .-6)

Extracted the following NER entity mentions:
```

Textual (Stanford)
or **CoNLL-U** format
useful for further processing

Stanza

Stanza – A Python NLP Package for Many Human Languages

pypi v1.3.0 | conda v1.3.0 | python 3.6 | 3.7 | 3.8

Stanza is a collection of accurate and efficient tools for the linguistic analysis of many human languages. Starting from raw text to syntactic analysis and entity recognition, Stanza brings state-of-the-art NLP models to languages of your choosing.

Stanza is a Python natural language analysis package. It contains tools, which can be used in a pipeline, to convert a string containing human language text into lists of sentences and words, to generate base forms of those words, their parts of speech and morphological features, to give a syntactic structure dependency parse, and to recognize named entities. The toolkit is designed to be parallel among more than 70 languages, using the Universal Dependencies formalism.

- Stanza is built with highly accurate neural network components that also enable efficient training and evaluation with your own annotated data. The modules are built on top of the PyTorch library. You will get much faster performance if you run the software on a GPU-enabled machine.
- In addition, Stanza includes a Python interface to the **CoreNLP Java package** and inherits additional functionality from there, such as constituency parsing, coreference resolution, and linguistic pattern matching.

stanfordnlp.github.io/stanza

Stanza – A Python NLP Package for Many Human Languages

- Native Python implementation requiring minimal efforts to set up;
- Full neural network pipeline for robust text analytics, including tokenization, multi-word token (MWT) expansion, lemmatization, part-of-speech (POS) and morphological features tagging, dependency parsing, and named entity recognition;
- Pretrained neural models supporting 66 (human) languages;
- A stable, officially maintained Python interface to CoreNLP.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton and Christopher D. Manning. 2020. Stanza: A Python Natural Language Processing Toolkit for Many Human Languages. In Association for Computational Linguistics (ACL) System Demonstrations. 2020.

- Install:

```
~$ which python3
/Library/Frameworks/Python.framework/Versions/3.8/bin/python3
~$ which pip3
/Library/Frameworks/Python.framework/Versions/3.8/bin/pip3
~$ pip3 install stanza
```

stanza

- Installation details:

```
Collecting stanza
  Downloading stanza-1.2-py3-none-any.whl (282 kB)
     |████████████████████████████████| 282 kB 3.2 MB/s
Collecting torch>=1.3.0
  Downloading torch-1.8.1-cp38-none-macosx_10_9_x86_64.whl (119.6 MB)
     |████████████████████████████████| 119.6 MB 365 kB/s
Requirement already satisfied: requests in /Library/Frameworks/Python.framework/Versions/3.8/lib/python3.8/site-packages (from stanza) (2.22.0)
Collecting tqdm
  Downloading tqdm-4.60.0-py2.py3-none-any.whl (75 kB)
     |████████████████████████████████| 75 kB 3.5 MB/s
Collecting protobuf
  Downloading protobuf-3.17.0-cp38-cp38-macosx_10_9_x86_64.whl (959 kB)
     |████████████████████████████████| 959 kB 3.3 MB/s
Requirement already satisfied: numpy in ./Library/Python/3.8/lib/python/site-packages (from stanza) (1.18.1)
Collecting typing-extensions
  Downloading typing_extensions-3.10.0.0-py3-none-any.whl (26 kB)
Requirement already satisfied: certifi>=2017.4.17 in /Library/Frameworks/Python.framework/Versions/3.8/lib/python3.8/site-packages (from requests->stanza) (2020.6.20)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /Library/Frameworks/Python.framework/Versions/3.8/lib/python3.8/site-packages (from requests->stanza) (1.25.8)
Requirement already satisfied: idna<2.9,>=2.5 in /Library/Frameworks/Python.framework/Versions/3.8/lib/python3.8/site-packages (from requests->stanza) (2.8)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in /Library/Frameworks/Python.framework/Versions/3.8/lib/python3.8/site-packages (from requests->stanza) (3.0.4)
Requirement already satisfied: six>=1.9 in ./Library/Python/3.8/lib/python/site-packages (from protobuf->stanza) (1.13.0)
Installing collected packages: typing-extensions, torch, tqdm, protobuf, stanza
Successfully installed protobuf-3.17.0 stanza-1.2 torch-1.8.1 tqdm-4.60.0 typing-extensions-3.10.0.0
```

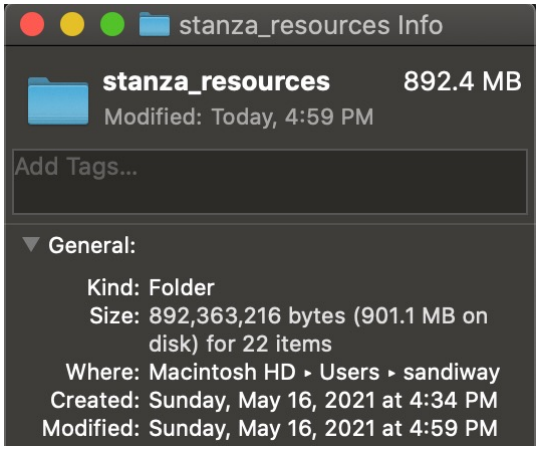
stanza

- Run:

```
~$ python3
Python 3.8.3 (v3.8.3:6f8c8320e9, May 13 2020, 16:29:34)
[Clang 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import stanza
>>> stanza.download('en')
Downloading https://raw.githubusercontent.com/stanfordnlp/stanza-resources/master/resources_1.2.0.json: 128kB [00:00, 45.8MB/s]
2021-05-16 16:34:20 INFO: Downloading default packages for language: en (English)...
Downloading http://nlp.stanford.edu/software/stanza/1.2.0/en/default.zip: 100%|
2021-05-16 16:36:16 INFO: Finished downloading models and saved to /Users/sandiway/stanza_resources.
>>> nlp = stanza.Pipeline('en')
2021-05-16 16:38:41 INFO: Loading these models for language: en (English):
=========================
| Processor | Package   |
-------------------------
| tokenize  | combined  |
| pos       | combined  |
| lemma     | combined  |
| depparse  | combined  |
| sentiment | sstplus   |
| ner       | ontonotes |
=========================
2021-05-16 16:38:41 INFO: Use device: cpu
2021-05-16 16:38:41 INFO: Loading: tokenize
2021-05-16 16:38:41 INFO: Loading: pos
2021-05-16 16:38:41 INFO: Loading: lemma
2021-05-16 16:38:41 INFO: Loading: depparse
2021-05-16 16:38:42 INFO: Loading: sentiment
2021-05-16 16:38:42 INFO: Loading: ner
2021-05-16 16:38:43 INFO: Done loading processors!
```

stanza

- Files in the home directory:



stanza: example

- **Assume:**

```
import stanza
stanza.download('en')
nlp = stanza.Pipeline('en')
```

- **Example again:**

```
>>> doc = nlp('Visiting relatives is boring.')
>>> for s in doc.sentences:
...     for w in s.words:
...             print(w.id, w.text, w.pos, w.deprel, w.head)
...
1 Visiting VERB csubj 4
2 relatives NOUN obj 1
3 is AUX cop 4
4 boring ADJ root 0
5 . PUNCT punct 4
```

stanza: example

```
>>> import stanza
>>> nlp = stanza.Pipeline('en')
>>> doc = nlp('Visiting relatives is boring.')
>>> doc.sentences[0]
… lots of output
>>> doc.sentences[1]
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
IndexError: list index out of range
>>> doc.sentences[0].print_dependencies()
('Visiting', 4, 'csubj')
('relatives', 1, 'obj')
('is', 4, 'cop')
('boring', 0, 'root')
('.', 4, 'punct')
>>>
```

stanza: CoreNLP

- https://stanfordnlp.github.io/stanza/corenlp_client.html

Stanford CoreNLP Client

Stanza allows users to access our Java toolkit, Stanford CoreNLP, via its server interface, by writing native Python code. Stanza does this by first launching a Stanford CoreNLP server in a background process, and then sending annotation requests to this server process. The response from the CoreNLP server will then be parsed and rendered into a Document protobuf object. As a result of this server-client communication, users can obtain annotations by writing native Python program at the client side, and do not need to worry about anything on the Java server side.

stanza: CoreNLP

- installation (*you get an extra copy of CoreNLP*):

```
>>> stanza.install_corenlp()
2022-03-01 11:13:55 INFO: Installing CoreNLP package into
/Users/sandiway/stanza_corenlp...
Downloading https://huggingface.co/stanfordnlp/CoreNLP/resolve/main/stanford-cor
```

- Uses a server (i.e *load only once, stay alive*)
- Run:

```
>>> from stanza.server import CoreNLPClient
>>> text = "Visiting relatives can be boring."
>>> with CoreNLPClient(annotators=['tokenize','ssplit','pos','lemma','ner',
'parse', 'depparse','coref'], timeout=30000, memory='6G') as client:
...     ann = client.annotate(text)
...
```

stanza: CoreNLP

2022-03-01 11:27:10 INFO: Writing properties to tmp file: corenlp_server-982e55f3530e4961.props

2022-03-01 11:27:10 INFO: Starting server with command: java -Xmx6G -cp /Users/sandiway/stanza_corenlp/* edu.stanford.nlp.pipeline.StanfordCoreNLPServer -port 9000 -timeout 30000 -threads 5 -maxCharLength 100000 -quiet False -serverProperties corenlp_server-982e55f3530e4961.props -annotators tokenize,ssplit,pos,lemma,ner,parse,depparse,coref -preload -outputFormat serialized

[main] INFO CoreNLP - --- StanfordCoreNLPServer#main() called ---

[main] INFO CoreNLP - Server default properties:

(Note: unspecified annotator properties are English defaults)

annotators = tokenize,ssplit,pos,lemma,ner,parse,depparse,coref

inputFormat = text

outputFormat = serialized

prettyPrint = false

threads = 5

[main] INFO CoreNLP - Threads: 5

...

[main] INFO edu.stanford.nlp.ie.AbstractSequenceClassifier - Loading classifier from edu/stanford/nlp/models/ner/english.all.3class.distsim.crf.ser.gz ... done [0.8 sec].

[main] INFO edu.stanford.nlp.ie.AbstractSequenceClassifier - Loading classifier from edu/stanford/nlp/models/ner/english.muc.7class.distsim.crf.ser.gz ... done [0.4 sec].

[main] INFO edu.stanford.nlp.ie.AbstractSequenceClassifier - Loading classifier from edu/stanford/nlp/models/ner/english.conll.4class.distsim.crf.ser.gz ... done [1.6 sec].

[main] INFO edu.stanford.nlp.time.JollyDayHolidays - Initializing JollyDayHoliday for SUTime from classpath edu/stanford/nlp/models/sutime/jollyday/Holidays_sutime.xml as sutime.binder.1.

[main] INFO edu.stanford.nlp.time.TimeExpressionExtractorImpl - Using following SUTime rules: edu/stanford/nlp/models/sutime/defs.sutime.txt,edu/stanford/nlp/models/sutime/english.sutime.txt,edu/stanford/nlp/models/sutime/english.holidays.sutime.txt

[main] INFO edu.stanford.nlp.pipeline.TokensRegexNERAnnotator - ner.fine.regexner: Read 580705 unique entries out of 581864 from edu/stanford/nlp/models/kbp/english/gazetteers/regexner_caseless.tab, 0 TokensRegex patterns.

[main] INFO edu.stanford.nlp.pipeline.TokensRegexNERAnnotator - ner.fine.regexner: Read 4867 unique entries out of 4867 from edu/stanford/nlp/models/kbp/english/gazetteers/regexner_cased.tab, 0 TokensRegex patterns.

...

[main] INFO edu.stanford.nlp.pipeline.StanfordCoreNLP - Adding annotator depparse

[main] INFO edu.stanford.nlp.parser.nndep.DependencyParser - Loading depparse model: edu/stanford/nlp/models/parser/nndep/english_UD.gz ... Time elapsed: 1.1 sec

[main] INFO edu.stanford.nlp.parser.nndep.Classifier - PreComputed 20000 vectors, elapsed Time: 1.098 sec

[main] INFO edu.stanford.nlp.parser.nndep.DependencyParser - Initializing dependency parser ... done [2.2 sec].

[main] INFO edu.stanford.nlp.pipeline.StanfordCoreNLP - Adding annotator coref

[main] INFO edu.stanford.nlp.coref.statistical.SimpleLinearClassifier - Loading coref model edu/stanford/nlp/models/coref/statistical/ranking_model.ser.gz ... done [0.8 sec].

[main] INFO edu.stanford.nlp.pipeline.CorefMentionAnnotator - Using mention detector type: dependency

[main] INFO CoreNLP - Starting server...

[main] INFO CoreNLP - StanfordCoreNLPServer listening at /0:0:0:0:0:0:0:0:9000

[pool-1-thread-3] INFO CoreNLP - [/0:0:0:0:0:0:0:1:64727] API call w/annotators tokenize,ssplit,pos,lemma,ner,parse,depparse,coref

Visiting relatives can be boring.

[pool-1-thread-3] INFO edu.stanford.nlp.pipeline.StanfordCoreNLP - Adding annotator tokenize

[pool-1-thread-3] INFO edu.stanford.nlp.pipeline.StanfordCoreNLP - Adding annotator ssplit

[pool-1-thread-3] INFO edu.stanford.nlp.pipeline.StanfordCoreNLP - Adding annotator pos

[pool-1-thread-3] INFO edu.stanford.nlp.pipeline.StanfordCoreNLP - Adding annotator lemma

[pool-1-thread-3] INFO edu.stanford.nlp.pipeline.StanfordCoreNLP - Adding annotator ner

[pool-1-thread-3] INFO edu.stanford.nlp.pipeline.StanfordCoreNLP - Adding annotator parse

[pool-1-thread-3] INFO edu.stanford.nlp.pipeline.StanfordCoreNLP - Adding annotator depparse

[pool-1-thread-3] INFO edu.stanford.nlp.pipeline.StanfordCoreNLP - Adding annotator coref

[Thread-0] INFO CoreNLP - CoreNLP Server is shutting down.

stanza: CoreNLP

```
>>> s = ann.sentence[0]
>>> parse = s.parseTree
>>> print(parse)
child {
 child {
  child {
   child {
    child {
     value: "Visiting"
    }
    value: "VBG"
    score: -9.652259826660156
   }
   child {
    child {
     child {
      value: "relatives"
     }
     value: "NNS"
     score: -8.183259010314941
    }
    value: "NP"
    score: -10.886569023132324
   }
   value: "VP"
   score: -22.3214054107666
  }
  value: "S"
  score: -23.077627182006836
 }
 child {
  child {
   child {
    value: "can"
   }
   value: "MD"
   score: -2.0786099433898926
  }
  child {
   child {
    child {
     value: "be"
    }
    value: "VB"
    score: -0.009304866194725037
   }
   child {
    child {
     child {
      value: "boring"
     }
     value: "JJ"
     score: -8.483654975891113
    }
    value: "ADJP"
    score: -9.187541007995605
   }
   value: "VP"
   score: -13.377534866333008
  }
  value: "VP"
  score: -17.71254539489746
 }
 child {
  child {
   value: "."
  }
  value: "."
  score: -0.05752464756369591
 }
 value: "S"
 score: -47.101341247558594
}
value: "ROOT"
score: -47.27272033691406
```

stanza: CoreNLP

```
>>> print(s.basicDependencies)
```

```
node {
  sentenceIndex: 0
  index: 2
}
node {
  sentenceIndex: 0
  index: 1
}
node {
  sentenceIndex: 0
  index: 5
}
node {
  sentenceIndex: 0
  index: 3
}
node {
  sentenceIndex: 0
  index: 4
}
node {
  sentenceIndex: 0
  index: 6
}
edge {
  source: 2
  target: 1
  dep: "amod"
  isExtra: false
  sourceCopy: 0
  targetCopy: 0
  language: UniversalEnglish
}
edge {
  source: 5
  target: 2
  dep: "nsubj"
  isExtra: false
  sourceCopy: 0
  targetCopy: 0
  language: UniversalEnglish
}
edge {
  source: 5
  target: 3
  dep: "aux"
  isExtra: false
  sourceCopy: 0
  targetCopy: 0
  language: UniversalEnglish
}
edge {
  source: 5
  target: 4
  dep: "cop"
  isExtra: false
  sourceCopy: 0
  targetCopy: 0
  language: UniversalEnglish
}
edge {
  source: 5
  target: 6
  dep: "punct"
  isExtra: false
  sourceCopy: 0
  targetCopy: 0
  language: UniversalEnglish
}
root: 5
```