STANFORD CORENLP

Final project

- Individual
- Any Python pipeline using, NLTK or CoreNLP, or other such resource
- Some input corpus/data
- Some language
- Some NLP task
  - Tokenization, sentence splitting
  - POS tagging
  - NER
  - Parsing (constituency or dependency)
  - Information extraction
  - Sentiment analysis
  - Relations extraction
- Some non-trivial component of innovation
- Some evaluation
- A write-up

Working on the final project

- Proposal due next Friday evening (2 pp. max)
  - Answer the questions on the previous slide
- April Mondays: meet in the lab (NOT JKB)
- (M)WF lab sessions starting next week: work on your project
- Final exam period: 10 minute presentation

Stanford Core NLP

- An integrated NLP toolkit with a broad range of grammatical analysis tools
- A fast, robust annotator for arbitrary texts, widely used in production
- A modern, regularly updated package, with the overall highest quality text analytics
- Support for a number of major (human) languages
- Available APIs for most major modern programming languages
  - Including a Python wrapper
- Ability to run as a simple web service

- Easy construction of pipelines

Running CoreNLP from Python

- This week: you'll learn how from PyCharm

- stanfordcorenlp by Lynten Guo. A Python wrapper to Stanford CoreNLP server, version 3.8.0. Also: PyPI page.
- pycorenlp, A Python wrapper for Stanford CoreNLP by Smitha Milli that uses the new CoreNLP v3.6+ server. Available on PyPI.
- corenlp-pywrap by Sherin Thomas also uses the new CoreNLP v3.6+ server. Python 3.x (only). Also: PyPI page.
- Stanford CoreNLP Python Interface: A reference implementation of a Python interface to the Stanford CoreNLP server. By Arun Chaganty. PyPI page.
- pynlp A (Pythonic) Python wrapper for Stanford CoreNLP by Sina. PyPI page.
- NLTK since version 3.2.3 has a new interface to Stanford CoreNLP using the StanfordCoreNLPServer. Among other places, see instructions on using the dependency parser and the code for this module, and if you poke around the documentation, you can find equivalent interfaces to other CoreNLP components; for example here is Stanford CoreNLP NER. Much of the work for this was done by Dmitrijs Milajevs.

Misc. utilities/frameworks using CoreNLP

- python-corenlp-protobuf: Stanford CoreNLP Python Bindings by Arun Chaganty. This package contains python bindings for Stanford CoreNLP's protobuf specifications, as generated by protoc. These bindings can used to parse binary data produced by, e.g., the Stanford CoreNLP server. PyPI page.
- PyStanfordDependencies , a Python interface for converting Penn Treebank trees to Stanford Dependencies by David McClosky (see also: PyPI page). Last we checked, it is at Stanford CoreNLP v3.5.2 and can do Universal and Stanford dependencies (though it's currently missing Universal POS tags and features).
- corenlp-xml, a library for handling interactions with CoreNLP's XML output by Robert Elwell. Available on PyPI. Documentation.
- corpkit, a sophisticated corpus linguistics toolkit with GUI by Daniel McDonald. Interfaces with CoreNLP v3.6.0 to parse documents, and uses Tregex/CoreNLP XML to find patterns in corpora. Available on PyPI. A graphical interface is also available.
- corenlp-xml-reader by Edward Newell on GitHub and there it's a PyPI package

CoreNLP annotators (miscellaneous)

- cleanxml: removes xml annotation from running text
- quote: parses out quoted statements from running text
- truecase: predicts natural case on running text
- ssplit: separates running text into sentences

CoreNLP annotators (lexical, morphology)

- pos: POS tagger
- lemma: finds lemmas
- tokenize: separates string into tokens

CoreNLP annotators (syntax, semantics, pragmatics)

- ner: a named-entity recognizer
- regexner: an NER engine with additional regex capabilities
- parse: a shift-reduce parser
- depparse: a dependency parser
- sentiment: a sentiment analysis engine
- dcoref: discourse coreference
- relation: finds and labels relationships between named entities
- natlog: natural log semantics including quantifier scope, polarity domains, etc.

Annotator dependencies

PROPERTY NAME	ANNOTATOR CLASS NAME	REQUIREMENTS
tokenize	TokenizerAnnotator	None
cleanxml	CleanXmlAnnotator	`tokenize`
ssplit	WordsToSentenceAnnotator	`tokenize`
pos	POSTaggerAnnotator	`tokenize, ssplit`
lemma	MorphaAnnotator	`tokenize, ssplit, pos`
ner	NERClassifierCombiner	`tokenize, ssplit, pos, lemma`
regexner	RegexNERAnnotator	?
sentiment	SentimentAnnotator	?
parse	ParserAnnotator	`tokenize, ssplit`
depparse	DependencyParseAnnotator	`tokenize, ssplit, pos`
dcoref	DeterministicCorefAnnotator	`tokenize, ssplit, pos, lemma, ner, parse`
relation	RelationExtractorAnnotator	`tokenize, ssplit, pos, lemma, ner, depparse`
natlog	NaturalLogicAnnotator	`tokenize, ssplit, pos, lemma, depparse` (Can also use `parse`)
quote	QuoteAnnotator	None

Language support

- Download as separate resources

LANGUAGE	MODEL JAR	VERSION
Arabic	download	3.9.1
Chinese	download	3.9.1
English	download	3.9.1
English (KBP)	download	3.9.1
French	download	3.9.1
German	download	3.9.1
Spanish	download	3.9.1

ANNOTATOR	AR	ZH	EN	FR	DE	ES
Tokenize / Segment	✓	✓	✓	✓		✓
Sentence Split	✓	✓	✓	✓	✓	✓
Part of Speech	✓	✓	✓	✓	✓	✓
Lemma			✓			
Named Entities		✓	✓		✓	✓
Constituency Parsing	✓	✓	✓	✓	✓	✓
Dependency Parsing		✓	✓	✓	✓	
Sentiment Analysis			✓			
Mention Detection		✓	✓			
Coreference		✓	✓			
Open IE			✓			

Output

- Lots of formats
  - Raw text
  - XML
  - CoNLL
  - JSON
  - Serialized

Machine learning

- Can train, save your own models
- Evaluation of results is very easy
  - Easy precision/recall/F-measures
  - Easy training/testing splits