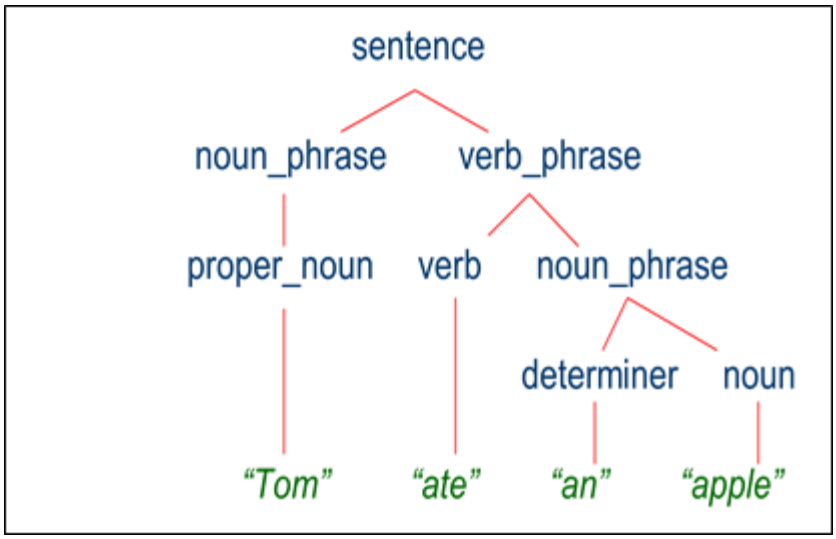
Stanford CoreNLP Parser

What is a parser?

In Natural Language Processing (NLP) a parser is used to determine the syntactic structure of a text by analyzing its constituent words based on an underlying grammar of the language of the text (e.g., English). A parser would automatically figure out, for instance, the structure of the sentence "Tom ate an apple" assigning each word in the sentence its proper syntactic label.



As we read on the Stanford CoreNLP website (https://nlp.stanford.edu/software/lex-parser.shtml) parsers' "development was one of the biggest breakthroughs in natural language processing in the 1990s."

Freeware open-source parsers

There are several available freeware opensource parsers (e.g., GATE (General Architecture for Text Engineering), https://gate.ac.uk/; Apache OpenNLP, https://opennlp.apache.org/; NLTK (Natural Language Toolkit) http://www.nltk.org/). See the TIPS file TIPS_NLP_NLP software

options.pdf.

Stanford CoreNLP is a leader in freeware, open source NLP tools (https://nlp.stanford.edu/software/lex-parser.shtml).

The Stanford CoreNLP parsers

Stanford CoreNLP provides different parsing options depending upon the users' needs: Probabilistic Context Free Grammar (PCFG), the recommended default for English, Shift-reduce constituency parser, Neural-network dependency parser (https://nlp.stanford.edu/software/lex-parser.shtml).

The NLP Suite implements two types of CoreNLP parsers:

1. *Probabilistic Context Free Grammar* (PCFG), the recommended default parser for the English language, first described in Klein and Manning (2003)
2. *Neural-network dependency parser*, a high-performance dependency parser powered by a neural network, first described in Chen and Manning (2014) (https://nlp.stanford.edu/software/nndep.html).

The parsers are constantly updated (https://stanfordnlp.github.io/CoreNLP/parse.html).

System requirements

The parser requires a reasonable amount of memory (at least 100MB to run as a PCFG parser on sentences up to 40 words in length; typically around 500MB of memory to be able to parse similarly long typical-of-newswire sentences using the factored model).

Java

Stanford CoreNLP is an NLP package written in Java. You will need to install the **freeware** Java on your machine.

Java can be downloaded from https://www.oracle.com/java/technologies/downloads/archive/

To download Java from the Oracle website, you will need to sign in in your Oracle account (you must create a FREE Oracle account if you do not have one).

If Java is not installed on your machine, the NLP Suite will ask you if you want to install it, then open the Java download website. You will need to select the most current Java SE version then download the JDK suited for your machine (Mac/Windows) and finally run the downloaded executable.

Input

The Stanford CoreNLP parser takes in input a single text file or a set of text files in a directory.

Output: The CoNLL table

In output the script produces a Json file that the NLP Suite Python 3 algorithm reprocesses in a

CoNLL table with the following 8 fields: ID, word (FORM), lemma (LEMMA), postag (POSTAG), ner (NER) (23 classes), governor (HEAD), deprel (DEPREL), Clausal Tags (these are not produced by the neural-network parser). For convenience, the NLP Suite Python 3 algorithms automatically adds the following fields to the standard 7 fields of a CoNLL table: RECORD NUMBER, DOCUMENT NUMBER, SENTENCE NUMBER, DOCUMENT NAME (INPUT filename), and DATE (if the filename embeds a date; e.g., The New York Time_12-11-2020).

Here is what the output CoNLL table for the story of *The Three Little Pigs* in csv format looks like.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	index	word	lemma	postag	ner	governor	deprel	Clausal Ta	RecordID	SentenceII	DocumentI	Document						
2	1	THERE	there	NN	O	4	nsubj	NP	1	1	1	C:/Users/myself/Desktop/CORPUS DATA/The Three Little Pigs.txt						
3	2	was	be	VBD	O	4	cop	VP	2	1	1	C:/Users/myself/Desktop/CORPUS DATA/The Three Little Pigs.txt						
4	3	an	a	DT	O	4	det	NP	3	1	1	C:/Users/myself/Desktop/CORPUS DATA/The Three Little Pigs.txt						
5	4	old	old	JJ	O	0	ROOT	0	4	1	1	C:/Users/myself/Desktop/CORPUS DATA/The Three Little Pigs.txt						
6	5	sow	sow	VB	O	4	dep	VP	5	1	1	C:/Users/myself/Desktop/CORPUS DATA/The Three Little Pigs.txt						
7	6	with	with	IN	O	9	case	0	6	1	1	C:/Users/myself/Desktop/CORPUS DATA/The Three Little Pigs.txt						
8	7	three	three	CD	NUMBER	9	nummod	NP	7	1	1	C:/Users/myself/Desktop/CORPUS DATA/The Three Little Pigs.txt						
9	8	little	little	JJ	O	9	amod	0	8	1	1	C:/Users/myself/Desktop/CORPUS DATA/The Three Little Pigs.txt						
10	9	pigs	pig	NNS	O	5	nmod:witl	0	9	1	1	C:/Users/myself/Desktop/CORPUS DATA/The Three Little Pigs.txt						
11	10	,	,	,	O	4	punct	0	10	1	1	C:/Users/myself/Desktop/CORPUS DATA/The Three Little Pigs.txt						
12	11	and	and	CC	O	4	cc	0	11	1	1	C:/Users/myself/Desktop/CORPUS DATA/The Three Little Pigs.txt						
13	12	as	as	IN	O	14	mark	SBAR	12	1	1	C:/Users/myself/Desktop/CORPUS DATA/The Three Little Pigs.txt						
14	13	she	she	PRP	O	14	nsubj	NP	13	1	1	C:/Users/myself/Desktop/CORPUS DATA/The Three Little Pigs.txt						
15	14	had	have	VBD	O	22	advcl:as	VP	14	1	1	C:/Users/myself/Desktop/CORPUS DATA/The Three Little Pigs.txt						
16	15	not	not	RB	O	14	neg	0	15	1	1	C:/Users/myself/Desktop/CORPUS DATA/The Three Little Pigs.txt						
17	16	enough	enough	JJ	O	14	xcomp	0	16	1	1	C:/Users/myself/Desktop/CORPUS DATA/The Three Little Pigs.txt						
18	17	to	to	TO	O	18	mark	VP	17	1	1	C:/Users/myself/Desktop/CORPUS DATA/The Three Little Pigs.txt						
19	18	keep	keep	VB	O	16	xcomp	VP	18	1	1	C:/Users/myself/Desktop/CORPUS DATA/The Three Little Pigs.txt						
20	19	them	they	PRP	O	18	dobj	NP	19	1	1	C:/Users/myself/Desktop/CORPUS DATA/The Three Little Pigs.txt						
21	20	,	,	,	O	22	punct	0	20	1	1	C:/Users/myself/Desktop/CORPUS DATA/The Three Little Pigs.txt						
22	21	she	she	PRP	O	22	nsubj	0	21	1	1	C:/Users/myself/Desktop/CORPUS DATA/The Three Little Pigs.txt						
23	22	sent	send	VBD	O	4	conj:and	VP	22	1	1	C:/Users/myself/Desktop/CORPUS DATA/The Three Little Pigs.txt						

The neural-network dependency parser and clausal tags

The neural-network parser does not produce clausal tags. The column will be empty in the CoNLL table.

Faulty results?

NLP parsers are far from perfect, although they are getting better and better at 95% accuracy.

But… if you suspect that the NLP Suite implementation of Stanford CoreNLP may have given faulty results for some sentences, you can test those sentences directly on the Stanford CoreNLP demo website at https://corenlp.run.

References

Chen, Danqi and Christopher D. Manning. 2014. "A Fast and Accurate Dependency Parser using Neural Networks." *Proceedings of EMNLP 2014*.

Klein, Dan and Christopher D. Manning. 2003. "Accurate Unlexicalized Parsing." *Proceedings of the 41st Meeting of the Association for Computational Linguistics*, pp. 423-430.

Manning, Christopher D., Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, David McClosky. 2014. "The Stanford CoreNLP Natural Language Processing Toolkit." *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp. 55-60.

TIPS_NLP_NLP software options.pdf
TIPS_NLP_Stanford CoNLL table.pdf