

ADAMS

Advanced **D**ata mining **A**nd **M**achine learning **S**ystem

Module: adams-nlp

NLP

Peter Reutemann

January 8, 2020

©2013-2019



THE UNIVERSITY OF
WAIKATO
Te Whare Wānanga o Waikato



Except where otherwise noted, this work is licensed under
<http://creativecommons.org/licenses/by-sa/4.0/>

Contents

1	Introduction	7
2	Flow	9
	Bibliography	11

List of Figures

1.1	Graphical representation of Stanford parse tree.	7
-----	--	---

Chapter 1

Introduction

The Stanford Parser is a GPL-licensed¹ library for natural language parsing. For instance, parsing this sentence:

The quick brown fox jumps over the lazy dog

will result in a parse tree like this:

```
(ROOT (NP (NP (DT The) (JJ quick) (JJ brown) (NN fox)) (NP (NP (NNS jumps))
  (PP (IN over) (NP (DT the) (JJ lazy) (NN dog))))))
```

The same parse tree in graphical representation:

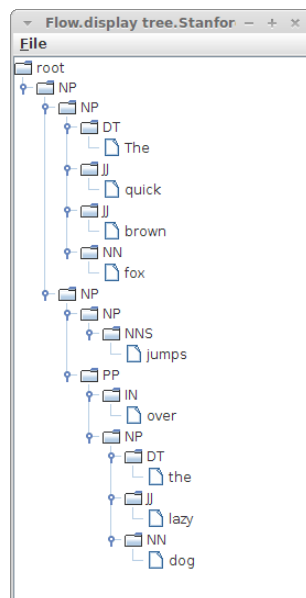


Figure 1.1: Graphical representation of Stanford parse tree.

¹version 2 or later <http://www.gnu.org/licenses/gpl-2.0.html>

Chapter 2

Flow

The following actors are available:

- *CoNLLFileReader* – transformer parsing text files in CoNLL format with one or more token sequences and turning them into spreadsheets.
- *DocumentToSentences* – transformer for splitting document strings into sentences using a specified splitter/tokenizer.¹
- *EditDistance* – computes the edit distance between a supplied base string and the strings received.
- *GenerateWordCloud* – transformer for generating a word cloud image from word frequencies.
- *StanfordGrammaticalStructure* – transformer for generating a grammatical structure from a parse tree.²
- *StanfordLexicalizedParser* – transformer for generating a parse tree from a string.³
- *StanfordParseTreeDisplay* – sink for displaying a parse tree.⁴
- *Stemmer* – transformer for performing stemming on words.
- *Tokenize* – transformer for splitting strings into words.
- *TweetParser* – transformer for POS tagging tweets (requires the external TweetParser executables).
- *TweetNLPTagger* – transformer for tokenizing/tagging tweets.
- *WordFrequencyAnalyzer* – transformer for generating word frequencies from text.

The following conversions are available:

- *SpreadSheetToWordFrequencies* – turns a spreadsheet with two columns (word/frequency) into a word frequency array.
- *StanfordParseTreeToSpreadSheet* – turns the leaves of a parse tree into a spreadsheet.
- *StanfordParseTreeToXML* – turns a parse tree into an XML string.⁵

¹adams-nlp-split_document_and_parse.flow

²adams-nlp-parse.flow

³adams-nlp-parse.flow

⁴adams-nlp-parse.flow

⁵adams-nlp-parse.flow

- *WordFrequenciesToSpreadShet* – turns a word frequency array into a spread-sheet.
- *WordFrequencyToString* – generates a string representation of a single word frequency object.

The following Weka tokenizers are available:

- *TwitterNLPTokenizer* – uses TwitterNLP’s Twokenize.

The following Weka filters are available:

- *unsupervised.attribute.TwitterNLPPos* – batch filter that adds frequencies of tag terms as determined by the TwitterNLP POS tagger.

Bibliography

- [1] *ADAMS* – Advanced Data mining and Machine learning System
<https://adams.cms.waikato.ac.nz/>
- [2] *The Stanford Parser* – A statistical parser
<http://nlp.stanford.edu/software/lex-parser.shtml>