Osman Momoh

Project I: Report

COMP 6751: Natural Language Analysis

Fall 2021

Purpose

This report is to present the Python modules used to satisfy the requirements of this assignment. Please refer to the source code for additional information.

Modules

Main.py

In this python file the file ids of the Reuters corpus are selected. These files are preprocessed and the results printed to the screen.

Pipeline.py

This module is the pipeline used to turn the raw texts into parsed and tagged structures. The pipeline is initialized with the corpus (Reuters) and file id. Calling *preprocess*() triggers the pipeline to run. We can then call the various functions to print relevant results. These functions are: *print_tokens(), print_sentences(), print_pos_tags(), print_written_numbers(),* and *print_dates().*

For tokenization, refer to the function *tokenize(raw_text).* Here, we use the NLTK.*word_tokenize()* function. We also modify the tokenizer results by converting tokens to lower case, and splitting tokens around slashes.

For number tokenization, refer to *normalize_numbers(tokens).* Here, we split tokens which refer to numbers around dashes. There was no need to normalize decimal numbers as this was already done by the vanilla NLTK tokenizer.

DateParser.py

This module is used for parsing dates. Examples of parsed dates can be found in the source code comments and in the Demo.pdf file. In order to construct the context-free grammar (CFG), multiple methods and for loops are used. This is due to potential thousands of years (we used 0000 -> 3000), and large amounts of days possible. There are just over 3000 productions in this CFG. Here are the productions, with most in-between values removed (they can be easily inferred however).

```
Grammar with 3232 productions (start state = DATE)
    DATE -> YEAR SEP MONTHDIG SEP DAYDIG
    DATE -> DAYDIG SEP MONTHDIG SEP YEAR
    DATE -> FULLYEAR
    DATE -> MONTH DAY
    DATE -> MONTH
    DATE -> ORDINAL OF MONTH
    DATE -> DAY OF MONTH
    DATE -> ORDINAL OF MONTH
    MONTH -> 'january'
    MONTH -> 'february'
    MONTH -> 'march'
    MONTH -> 'april'
    MONTH -> 'september'
    MONTH -> 'october'
    MONTH -> 'november'
    MONTH -> 'december'
    MONTHDIG -> '01'
    MONTHDIG -> '02'
    MONTHDIG -> '03'
    MONTHDIG -> '04'
    MONTHDIG -> '08'
    MONTHDIG -> '09'
    MONTHDIG -> '10'
    MONTHDIG -> '11'
    MONTHDIG -> '12'
    ORDINAL -> SMALLORDINAL
    ORDINAL -> LARGEORDINAL SMALLORDINAL
    SMALLORDINAL -> 'first'
    SMALLORDINAL -> 'second'
    SMALLORDINAL -> 'third'
    SMALLORDINAL -> 'fourth'
    SMALLORDINAL -> 'fifth'
    SMALLORDINAL -> 'sixth'
    SMALLORDINAL -> 'sixteenth'
    SMALLORDINAL -> 'seventeenth'
    SMALLORDINAL -> 'eighteenth'
    SMALLORDINAL -> 'nineteenth'
    LARGEORDINAL -> 'twenty'
    LARGEORDINAL -> 'twentieth'
    LARGEORDINAL -> 'thirty'
    LARGEORDINAL -> 'thirtieth'
    SEP -> '/'
    OF -> 'of'
    DAY -> '1st'
    DAY -> '2nd'
    DAY -> '3rd'
    DAY -> '4th'
    DAY -> '5th'
    DAY -> '6th'
    DAY -> '25th'
    DAY -> '26th'
    DAY -> '27th'
    DAY -> '28th'
    DAY -> '29th'
    DAY -> '30th'
    DAY -> '31st'
    DAYDIG -> '0'
    DAYDIG -> '00'
    DAYDIG -> '1'
    DAYDIG -> '01'
    DAYDIG -> '2'
    DAYDIG -> '02'
    DAYDIG -> '3'
    DAYDIG -> '03'
    DAYDIG -> '25'
    DAYDIG -> '26'
    DAYDIG -> '27'
    DAYDIG -> '28'
    DAYDIG -> '29'
    DAYDIG -> '30'
    DAYDIG -> '31'
    FULLYEAR -> '0000'
    FULLYEAR -> '0001'
    FULLYEAR -> '0002'
    FULLYEAR -> '0003'
    FULLYEAR -> '0004'
    FULLYEAR -> '0061'
    FULLYEAR -> '1983'
    FULLYEAR -> '1984'
    FULLYEAR -> '1985'
    FULLYEAR -> '1986'
    FULLYEAR -> '1987'
    FULLYEAR -> '1988'
    FULLYEAR -> '1989'
    FULLYEAR -> '1990'
    FULLYEAR -> '1991'
    FULLYEAR -> '1992'
    FULLYEAR -> '1993'
    FULLYEAR -> '2030'
    FULLYEAR -> '2031'
    FULLYEAR -> '2032'
    FULLYEAR -> '2033'
    FULLYEAR -> '2995'
    FULLYEAR -> '2996'
    FULLYEAR -> '2997'
    FULLYEAR -> '2998'
    FULLYEAR -> '2999'
    YEAR -> '00'
    YEAR -> '01'
    YEAR -> '02'
    YEAR -> '03'
    YEAR -> '04'
    YEAR -> '05'
    YEAR -> '06'
    YEAR -> '07'
    YEAR -> '08'
    YEAR -> '09'
    YEAR -> '10'
    YEAR -> '68'
    YEAR -> '69'
    YEAR -> '70
    YEAR -> '86'
    YEAR -> '87'
    YEAR -> '88'
    YEAR -> '89'
```

```
   YEAR -> '90'
   YEAR -> '91'
   YEAR -> '92'
   YEAR -> '93'
   YEAR -> '94'
   YEAR -> '95'
   YEAR -> '96'
   YEAR -> '97'
   YEAR -> '98'
   YEAR -> '99'
```

WrittenNumberParser.py

This module is used for parsing written numbers. Examples of parsed numbers can be found in the source code comments and in the Demo.pdf file. Here is the CFG for this module:

```
Grammar with 39 productions (start state = NUMBER)
   NUMBER -> SMALLCARDINAL
   NUMBER -> MEDIUMCARDINAL
   NUMBER -> MEDIUMCARDINAL SMALLCARDINAL
   NUMBER -> SMALLCARDINAL LARGECARDINAL AND SMALLCARDINAL
   NUMBER -> SMALLCARDINAL LARGECARDINAL AND MEDIUMCARDINAL
   NUMBER -> SMALLCARDINAL LARGECARDINAL AND MEDIUMCARDINAL SMALLCARDINAL
   SMALLCARDINAL -> 'one'
   SMALLCARDINAL -> 'two'
   SMALLCARDINAL -> 'three'
   SMALLCARDINAL -> 'four'
   SMALLCARDINAL -> 'five'
   SMALLCARDINAL -> 'six'
   SMALLCARDINAL -> 'seven'
   SMALLCARDINAL -> 'eight'
   SMALLCARDINAL -> 'nine'
   SMALLCARDINAL -> 'ten'
   SMALLCARDINAL -> 'eleven'
   SMALLCARDINAL -> 'twelve'
   SMALLCARDINAL -> 'thirteen'
   SMALLCARDINAL -> 'fourteen'
   SMALLCARDINAL -> 'fifteen'
   SMALLCARDINAL -> 'sixteen'
   SMALLCARDINAL -> 'seventeen'
   SMALLCARDINAL -> 'eighteen'
   SMALLCARDINAL -> 'nineteen'
   MEDIUMCARDINAL -> 'twenty'
   MEDIUMCARDINAL -> 'thirty'
   MEDIUMCARDINAL -> 'forty'
   MEDIUMCARDINAL -> 'fifty'
   MEDIUMCARDINAL -> 'sixty'
   MEDIUMCARDINAL -> 'seventy'
   MEDIUMCARDINAL -> 'eighty'
   MEDIUMCARDINAL -> 'ninety'
   LARGECARDINAL -> 'hundred'
   LARGECARDINAL -> 'thousand'
   LARGECARDINAL -> 'million'
   LARGECARDINAL -> 'billion'
   LARGECARDINAL -> 'trillion'
   AND -> 'and'
```