COMP 6751 Natural Language Analysis

Project 1 Report 1

Student: Yixuan Li 40079830

Table of Contents

Expectations of originality:

I, student 40079830, certify that this submission is my original work and meets the Faculty's Expectations of Originality.

Date: October 4, 2020

I. Modules documentation

1) Module 1: Tokenization

(a) Base regular expression tokenizer

```
(?:[A-Z]\.)+
  | \w+(?:-\w+)*
  | \$?\d+(?:\.\d+)?%?
  | \.\.\.
  | [][.,;"'?():-_`]
```

Operator	Explanation
(?:)	Non-capturing group
([A-Z]\.)+	abbreviations, e.g. U.S.A.
\w+(?:-\w+)*	words with optional internal hyphens
\$?\d+(?:\.\d+)?%?	currency and percentages
\.\.\.	ellipsis …
[][.,;"'?():-_`]	separate tokens

(b) Enhanced regular expression tokenizer

```
(?:[A-Z]\.)+
  | \$?\d+(?:,\d+)?(?:\.\d+)?%?
  | \w+(?:-\w+)*
  | \.\.\.
  | \'[sS]
  | [][.,;"'?():-_`]
  | \w+
```

Operator	Improvement
\$?\d+(?:,\d+)?(?:\.\d+)?%?	group numbers with a comma and/or period (number normalization), e.g. 1,655.8, also support for currency or percentages, e.g. $1,655.8, 1,000.5%
\'[sS]	group possessive ending “’s” and “’S”

2) Module 2: Sentence Splitting

```
sents = nltk.sent_tokenize(raw)
```

nltk.sent_tokenize returns a sentence-tokenized copy of text using sentence tokenizer PunktSentenceTokenizer

3) Module 3: POS Tagging

For each sentence, first we get a list of tokens after tokenization. Then call

```
pos_tags = nltk.pos_tag(tokens)
```

nltk.pos_tag tags the given list of tokens

4) **Module 4: Entity Detection**

   a. Compile a unit gazetteer for common units and save as “unit_gazetteer.txt”. The format is *POS_TAG UNIT WORD*. The small database is constructed from multiple websites which are listed in Reference section.

   b. Define a context-free grammar to parse the sentences

      Grammar:

```
UN: {<CD> <NN> <IN|JJ|VBG|NNP> <NN>{0,2}}
  {<CD> <NN> <FW> <FW>}
  {<CD> <NN|NNP> <NN>? <NN|NNP|NNS>?}
  {<CD> <NN>{1,3}}
  {<CD> <NNS> <IN> <NN>}
  {<CD> <CD>? <NNS>}
  {<CD> <JJ> <NNS|NNP|NN|NNPS>{1,2}}
  {<CD> <JJ> <JJ> <NNS>? <IN>? <NN|NNP>}
  {<CD> <VBG> <JJ> <NN>}
  {<CD> <CD|IN|$>}
```

   c. Traverse the parse tree and extract the subtrees which label is “UN” and match units with “unit_gazetteer.txt” database

5) **Module 5: Date Recognition**

   a. Use POS tags and a context-free grammar to extract dates in sentences.

      Grammar:

```
DATE:       {<NNP> <CD> <,>? <CD>}
{<DT> <NN> <IN> <NNP>}
{<DT> <CD> <IN> <NNP>}
{<IN> <NNP> <CD>}
{<NNP> <CD>}
{<IN> <CD>}
{<IN> <JJ>}
```

   b. Traverse the parse tree and save tokens under subtrees which label is “DATE”.

   c. Verify the tokens satisfy pre-defined date patterns:

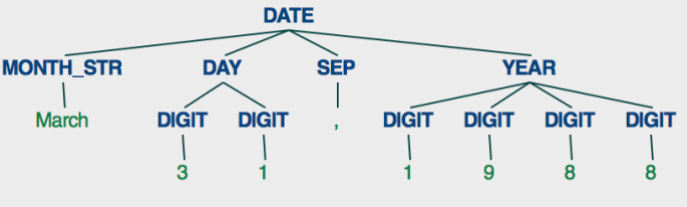
```
"(\d{4}[-/]?\d{2}[-/]?\d{2})"\
"|(\w+\s\d{1,2}[a-zA-Z]{2},?\s?\d{4}?)"\
"|(the\s\d{1,2}[a-zA-Z]{2}\sof\s[a-zA-Z]+)"\
"|(the\s\w+\sof\s\w+)"\
"|(\w+\s\d{1,2}[a-zA-Z]{2})"\
"|(\w+\s\d{1,2})"
```

   d. Save the verified tokens as date_list and return to PreProcess.

6) **Module 6: Date Parsing**

   a. The input to Date Parser is the list of dates detected during Date Recognition

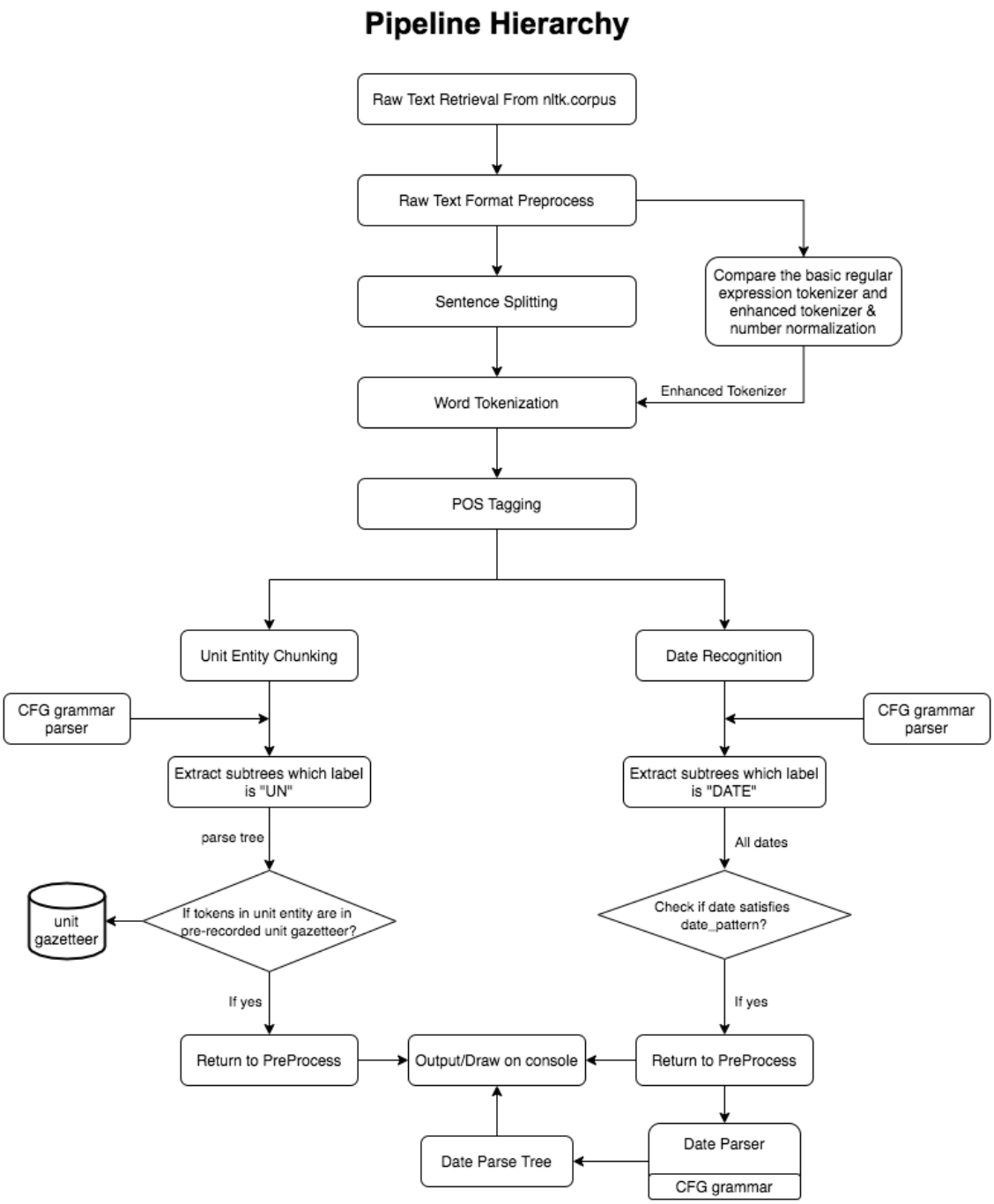
b. The output is parse tree where DATE as the root and the leaves are alphanumeric characters as year, month and day. Such as:



c. The grammar of date parser is defined as the following:

```
DATE -> IN YEAR SEP MONTH_NUM SEP DAY | YEAR SEP MONTH_NUM SEP DAY | MONTH_STR DAY SEP YEAR |
        MONTH_STR DAY | MONTH_STR YEAR | IN MONTH_STR NN_NUM | MONTH_STR NN_NUM YEAR |
        MONTH_STR NN_NUM SEP YEAR | MONTH_STR NN_NUM | DT NN_STR IN MONTH_STR | DT NN_NUM IN
        MONTH_STR | IN YEAR | IN MONTH_STR YEAR
SEP -> "/" | "-" | ","
YEAR -> DIGIT DIGIT DIGIT DIGIT
MONTH_NUM -> DIGIT | DIGIT DIGIT
DAY -> DIGIT | DIGIT DIGIT
DT -> "the"
IN -> "of" | "in" | "on"
NN_STR -> "first" | "second" | "third" | "fourth" | "fifth" | "sixth" | "seventh" | "eighth" | "ninth" | "tenth" |
        "eleventh" | "twelfth" | "thirteenth" | "fourteenth" | "fifteenth" | "sixteenth" | "seventeenth" |
        "eighteenth" | "nineteenth" | "twentieth" | "twenty-first" | "twenth-second" | "twenty-third" | "twenty-
        fourth" | "twenty-fifth" | "twenty-sixth" | "twenty-seventh" | "twenty-eighth" | "twenth-ninth" |
        "thirtieth" | "thirty-first"
MONTH_STR -> "January" | "February" | "March" | "April" | "May" | "June" | "July" | "August" | "September" |
        "October" | "November" | "December"
NN_NUM -> "1st" | "2nd" | "3rd" | "4th" | "5th" | "6th" | "7th" | "8th" | "9th" | "10th" | "11th" | "12th" | "13th" |
        "14th" | "15th" | "16th" | "17th" | "18th" | "19th" | "20th" | "21st" | "22nd" | "23rd" | "24th" | "25th" |
        "26th" | "27th" | "28th" | "29th" | "30th" | "31st"
DIGIT -> "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
```

II. Pipeline



III. Limitations

The enhanced tokenizer and context-free grammars perform well on the specified file “training/267” and all the dates (“in 1987”, “in 1986”) are detected.

However, there are still some limitations in the pipeline listed below:

1) The mini database “unit_gazetteer.txt” only covers a small set of unit of measurement words.
2) Date recognizer currently supports a list of certain date format listed below:

#	Support Date Format
1	2020/12/12

2	2020-12-12
3	December 12, 2020
4	December 12
5	December 2020
6	December 12th 2020
7	December 12th, 2020
8	December 12th
9	the twelfth of December
10	the 12th of December
11	December 12
12	in 2020
13	in December 2020

For the date formats that are not covered, for example in “training/279”, there is a date named as “in 1987/88” which indicates “from 1987 to 1988”, in this case only “in 1987” can be detected. Since it is not easy to differentiate year abbreviations and non-date integers, my grammar and regular expression only support common date formats which contains 4 digits. Hence “/88” is omitted.

3) For the 12th format above, my grammar can detect the date only if there is a preposition in front of the year. I will discuss this in Report 2 (Demo) in “II. Interesting case” section.

IV. References

1. NLTK API Tokenization Documentation (https://www.nltk.org/api/nltk.tokenize.html)
2. NLTK API POS Documentation (https://www.nltk.org/api/nltk.tag.html)
3. Unit gazetteer database references:
   http://www.ibiblio.org/units/index.html
   https://www.hobbyprojects.com/dictionary_of_units.html
   https://www.teachstarter.com/au/teaching-resource/units-of-measurement-word-wall-vocabulary/
4. NLTK book chapter 2,3,5,7,8L (http://www.nltk.org/book/)