Natural Language Processing

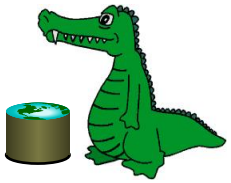
Miguel Rodríguez

Outline

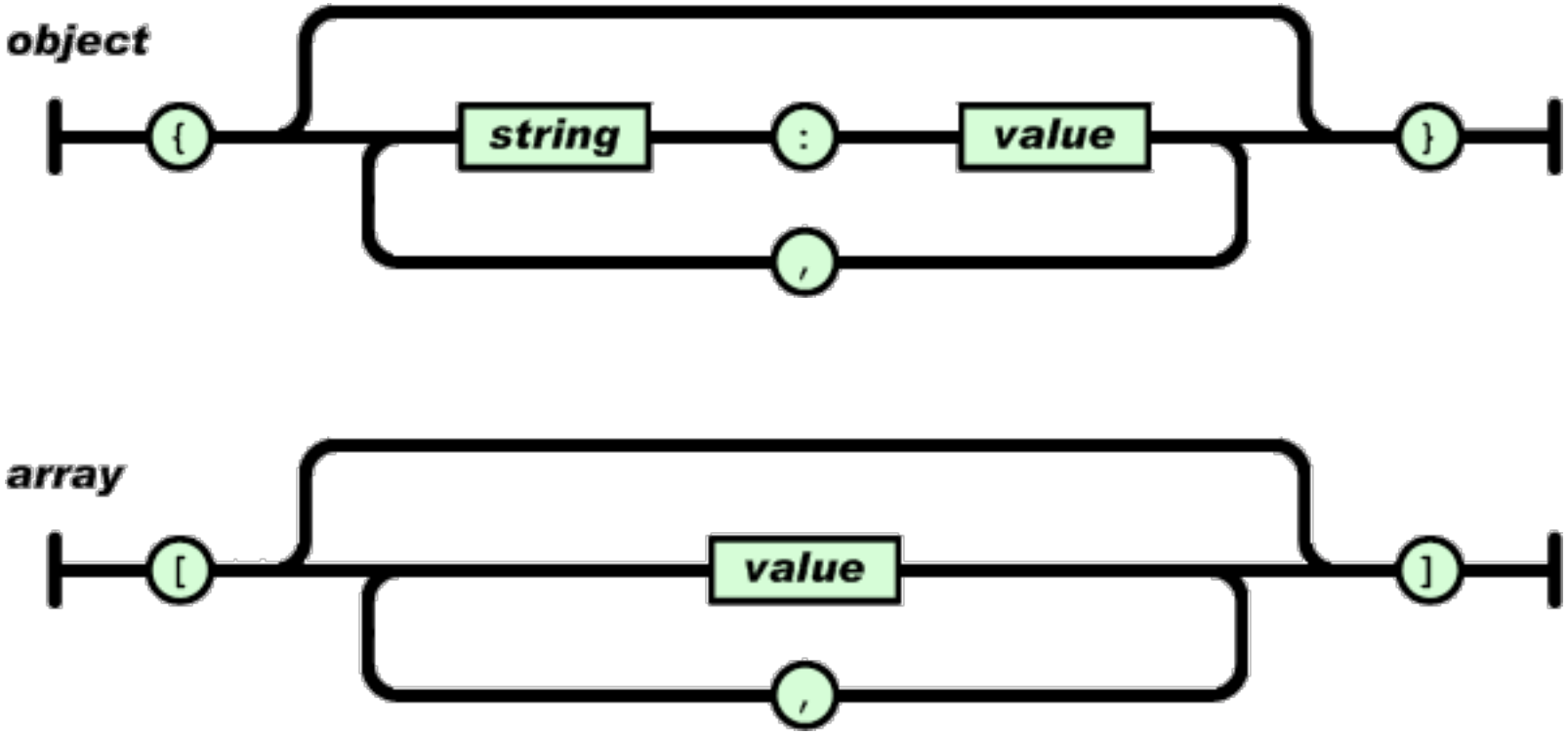
- JSON
- Environment Setup
- Data Acquisition
- Natural Language Parsing
- Problems

JSON

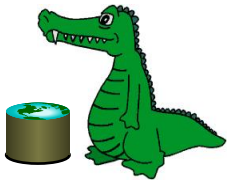
- JavaScript Object Notation
  - Text representation of Objects
- Easy for human
  - Read
  - Write
- Easy for machine
  - Generate
  - Parse



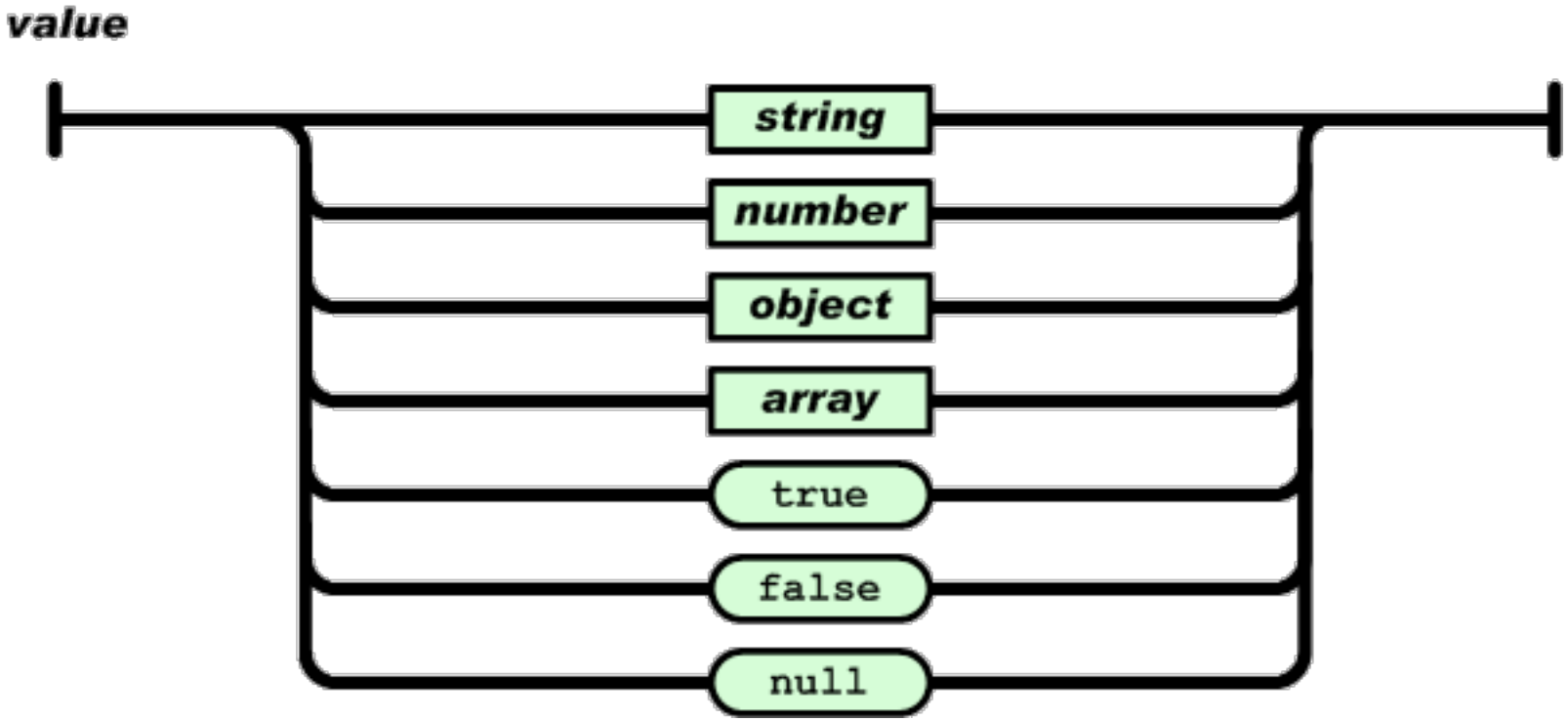
JSON



Source: http://json.org/



JSON



Source: http://json.org/

JSON - Example

```
{ "firstName": "John",  "lastName": "Smith",
   "isAlive": true, "age": 25,
   "address": { "streetAddress": "21 2nd Street",
                "city": "New York", "state": "NY",
                "postalCode": "10021-3100" },
   "phoneNumbers": [
        { "type": "home", "number": "212 555-1234" },
        { "type": "office", "number": "646 555-4567" } ],
   "children": [],
   "spouse": null }
```

Source: https://en.wikipedia.org/wiki/JSON/

Environment Setup

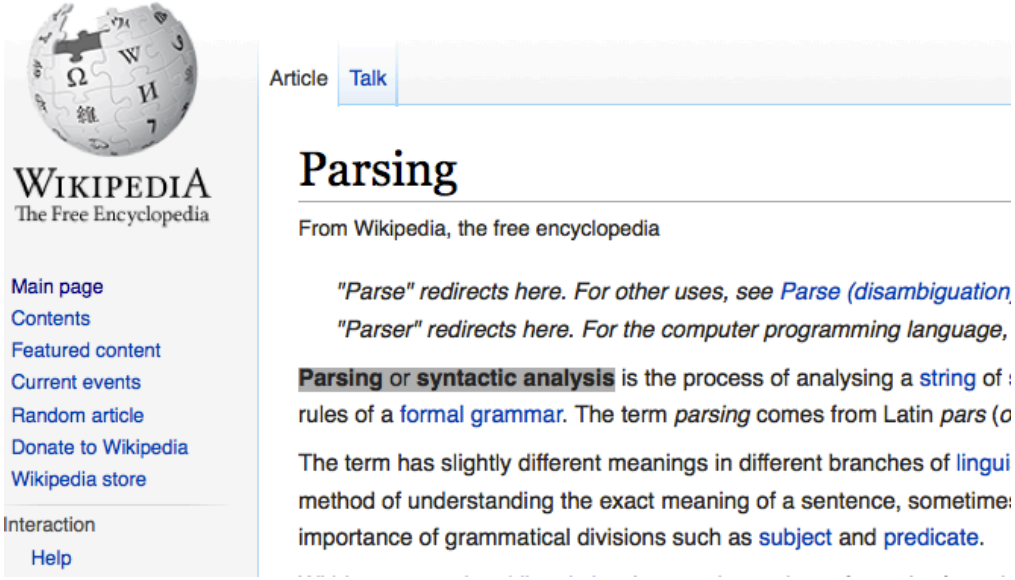
- Mediawiki Parser
  - Python package: “mwparserfromhell”
    - sudo pip install mwparserfromhell
- Requests Module
  - Python package: “requests”
    - sudo pip install requests
- Stanford Parser
  - Download from Canvas files
  - Follow installing guide from Canvas Pages

Data Acquisition

- Mediawiki RESTful API
  - Documentation here
  - Options to request
    - format=json to receive JSON data
    - action=query to query Wikipedia content
    - titles=string to specify a list of page titles to search for
    - prop=revision to return the latest revision
    - rvprop=content to return the full page content

Data Acquisition - Code

```
import requests
title='parsing'
response = requests.get(
                    "http://en.wikipedia.org/w/api.php?
                    format=json&action=query&titles="+str(title)
                    +"&prop=revisions&rvprop=content")
```




```
{"batchcomplete":"","query":{"normalized":
[{"from":"parsing","to":"Parsing"}],"pages":{"310015":
{"pageid":310015,"ns":0,"title":"Parsing","revisions":
[{"contentformat":"text/x-
wiki","contentmodel":"wikitext","*":"
{{Redirect|Parse}}\n{{redirect|Parser|the computer
programming language|Parser (CGI language)}}\n
\n'''Parsing''' or '''syntactic analysis''' is the
process of analysing a [[String (computer
science)|string]] of [[Symbol (programming)|symbols]],
either in [[natural language]] or in [[computer
languages]], conforming to the rules of a [[formal
grammar]].  The term ''parsing'' comes from Latin
''pars'' (''orationis''), meaning [[Part of speech|part
(of speech)]].<ref>{{cite web
|url=http://www.bartleby.com/61/33/P0083300.html
|title=Bartleby.com homepage |accessdate=28 November
2010}}</ref><ref name=\"dictionary.com\">{{cite web
|url=http://dictionary.reference.com/search?q=parse&
x=0&y=0 |title=parse
|publisher=dictionary.reference.com |accessdate=27
November 2010}}</ref>\n\nThe term has slightly
different meanings in different branches of
```

Data Acquisition - Code

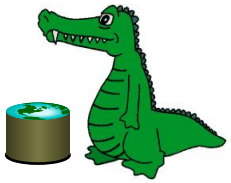
```
import json
jsondata = response.json()
def pretty(jdata):
        str = json.dumps(jdata,
                                sort_keys=True,
                indent=4)
        return str.decode('string_escape')

def saveas(sdata, fname):
        f = open(fname,'w')
        f.write(sdata)
        f.close()

saveas(pretty(jsondata), '/home/datascience/labs/lab4/'+title
+'.json')
```

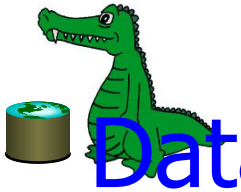
Data Acquisition - Code

```
#Explore JSON
type(jsondata)
jsondata.keys()
jsondata['query']
type(jsondata['query'])
jsondata['query'].keys()
jsondata['query']['pages'].values()[0]
....
# content = ???
```



Data Acquisition – Wikimedia parse

```
import mwparserfromhell as mwph
#Parse using wikipedia format
wikicode = mwph.parse(content)
#Some Filters
wikicode.filter_comments()
wikicode.filter_headings()
#Get rid of format
text = wikicode.strip_code()
```



Data Acquisition – Before and After

Raw Data

```
                "revisions" : [
                   {
                      "*": "{{Redirect|Parse}}
{{redirect|Parser|the computer programming
language|Parser (CGI language)}}

'''Parsing''' or '''syntactic analysis''' is the
process of analysing a [[String (computer
science)|string]] of [[Symbol
(programming)|symbols]], either in [[natural
language]] or in [[computer languages]], conforming
to the rules of a [[formal grammar]].  The term
''parsing'' comes from Latin ''pars''
(''orationis''), meaning [[Part of speech|part (of
speech)]].<ref>{{cite web
|url=http://www.bartleby.com/61/33/P0083300.html
|title=Bartleby.com homepage |accessdate=28
November 2010}}</ref><ref name="dictionary.com">
{{cite web |url=http://dictionary.reference.com
/search?q=parse&x=0&y=0 |title=parse
|publisher=dictionary.reference.com |accessdate=27
November 2010}}</ref>

The term has slightly different meanings in
different branches of [[linguistics]] and
[[computer science]].  Traditional sentence parsing
```

Clean Data

"Parsing or syntactic analysis is the process of analysing a string of symbols, either in natural language or in computer languages, conforming to the rules of a formal grammar. The term parsing comes from Latin pars (orationis), meaning part (of speech).

The term has slightly different meanings in different branches of linguistics and computer science. Traditional sentence parsing is often performed as a method of understanding the exact meaning of a sentence, sometimes with the aid of devices such as sentence diagrams. It usually emphasizes the importance of grammatical divisions such as subject and predicate.

Within computational linguistics the term is used to refer to the formal analysis by a computer of a sentence or other string of

Natural Language Parsing

- From terminal
  - lexparser-gui.sh
- Load parser file
  - /opt/StanfordParser/stanford-parser-3.4.1-models.jar
- Select parser
  - englishPCFG.ser.gz
- Parse some text!!!

NLP – Content Analysis

```
from lxml import etree

parser = etree.XMLParser(recover=True)

tree = etree.parse('/home/datascience/
labs/lab4/cat.xml',parser)
```

MLP – Examine Tree

```
root=tree.getroot()
root.tag
len(root)
root[0].tag
root[1][0][0].attrib['value']
#Sentence node
s=root[6][0][0][0]
s.attrib['value']
```

NLP – Exploring the Tree

```
#Get Children
s[:]
#Node types are hidden inside value
map(lambda (x): x.attrib['value'], s[:])
```

NLP – Exploring the Tree

- Etree – findall()
  - Flexible Syntax to locate elements
    - Type or attribute
- Slash (“/”) defines a child node
- Double slash (“//”) defines any descendant
- “node[@value=‘…’]” specify a node with the given attribute value

NLP – Exploring the tree

```
agent = s.findall("./node[@value='NP']//
node[@value='NN']//leaf[@value='cat']")

verb = s.findall("./node[@value='VP']//
node[@value='VBZ']//leaf[@value='is']")
```

- Finds all the nodes starting with an 'NP' child of s, and having a 'NN' node above a leaf with 'cat' value

NLP – Altogether

```
def printnode(node):
        for i in node.findall(".//leaf"):
                print(" " + i.attrib['value']), print('')
def testnode(node, agent, action):
        aa = node.findall("./node[@value='NP']//
node[@value='NN']//leaf[@value='"+agent+"']")
        bb = node.findall("./node[@value='VP']//
leaf[@value='"+action+"']")
        if (len(aa) > 0 and len(bb) > 0):
                printnode(node)
def agentact(node, agent, action):
        testnode(node, agent, action)
        snodes = node.findall(".//node[@value='S']")
        for snode in snodes:
                testnode(snode, agent, action)
```

NLP – Let’s try!

agentact(s, title, 'is')

map(lambda (nn): agentact(nn[0][0][0], title, 'is'), root) []

Problems

1. Write code to extract the actual content of the current version of a Wikipedia page.
2. Load the first sentence of the "Parse" wikipedia article using the method provided. Did it parse correctly? Explain.
3. Modify the given testnode function such that more facts about cats can be extracted.
4. Extract facts about this people's wikipedia pages
   - Jim Parsons
   - Barack Obama

Challenge Problem

1. Can you write code to automatically extract the following facts about a given person's wikipedia page?
   1. Place of birth
   2. Spouse
   3. Schools attended

   Test your code using Barack Obama's wikipedia page

   Hint: you can write different fuctions for each relation.