Ling575 Summarization System

D2: Process a docSet

Rachel Hantz, Yi-Chien Lin, Yian Wang, Chenxi Li, Tashi Tsering

Overview

- Set up Anaconda on Patas
- preprocess the input data
  - Parse the input data using library such as xml.dom.minidom, xml.etree.ElementTree and lxml.etree
  - Tokenize sentences using sent_tokenize/word_tokenize
- Write up/Slides

WorkSplit

- Coding: Yian (with help from Rachel and Yi-Chien)
- Environment setup: Yi-Chien
- Report: Chenxi
- Slides: Tashi
- Presentation: Rachel

Demo

xml.dom.minidom

- A minimal implementation of the Document Object Model interface
- Intended to be simpler and significantly smaller than the full DOM.
- xml.dom.minidom.parse() can take either a file name or a file-like object
- parse() function return a document.

```
from xml.dom.minidom import parse

dom1 = parse('c:\\temp\\mydata.xml')  # parse an XML file by name
datasource = open('c:\\temp\\mydata.xml')
dom2 = parse(datasource)  # parse an open file
```

Preprocess

Process

- Import xml.dom.minidom to parse the path of the file
- Extracts the document ID under 'docsetA'
- Create a directory under output for each 'docsetA'

```
def process(path):
    # read the .xml file using minidom parser
    mode = path.split('/')[6]
    DOMTree = xml.dom.minidom.parse(path)
    collection = DOMTree.documentElement
    topics = collection.getElementsByTagName("topic")
    for topic in topics:
        docsetAs = topic.getElementsByTagName("docsetA")
        for docSetA in docsetAs:
            directory = docSetA.getAttribute("id")  #Get the id for docSetA
```

```
directory = docSetA.getAttribute("id")
path = '../outputs/' + mode + '/' + directory
if not os.path.exists(path):
    os.makedirs(path)
```

xml.etree.ElementTree

- A simple and efficient API for parsing and creating XML data
- ElementTree represents the whole XML document as a tree
- Element represents a single node in this tree
- Parse() function return a tree.
- Root is an element and has children nodes
- Can access specific child nodes by index

```
import xml.etree.ElementTree as ET
tree = ET.parse('data.xml')
root = tree.getroot()
root[child][1][0].tag == 'P'
TEXT = root[child][1].text.split('\n\n')
```

lxml.etree

- A new Python binding for C library libxml2 and libxslt
- Support parsing for both XML and HTML
- Full xpath support
- Faster than ElementTree in most cases
- Recover set to True to parse through broken XML
- parse() return an ElementTree object

```
from lxml import etree
f = open('tmp.txt', 'r')
parser = etree.XMLParser(recover=True)
tree = etree.parse(f, parser)
root = tree.getroot()
DOCs = root.findall('DOC')
```

Preprocess

Tokenization

- Locate the article according the doc_id
- Generate a tree from the article using xml.etree.ElementTree
    - Exception handling needed since there are different format
- Use nltk.sent_tokenize and nltk.word_tokenize to split paragraph and tokenize sentences

```
for file in allFile:
    if (doc_id in file):
        tree = xml.etree.ElementTree.parse(dir+subdir+'/'+file)
```

```
para = p.text.strip('\n')
sent_text = nltk.sent_tokenize(para)
for sentence in sent_text:
    tokens = word_tokenize(sentence)
```

Result

- In the output file, each sentence is listed in single line
- Headline and date-time will be listed in the start if it is documented in the article
- A blank line is used to separate two paragraphs.

```
</TEXT>
</DOC>
<DOC id="AFP_ENG_20050119.0019" type="story" >
<HEADLINE>
Indian, Pakistan military to discuss alleged Kashmir ceasefire
violation
</HEADLINE>
<DATELINE>
NEW DELHI, Jan 19
</DATELINE>
<TEXT>
<P>
Indian and Pakistani military commanders...
</P>
<P>
"The Pakistani side has denied firing. Let's see."
</P>
```

```
headline: Indian, Pakistan military to discuss alleged Kashmir ceasefire
violation

date-time: NEW DELHI, Jan 19

Indian and Pakistani military commanders...

`` The Pakistani side has denied firing .
Let 's see . ''
```

Problems/Solutions

Problem: Some documents in the corpora are not rooted.

Solution: created a temporary root node by inserting <tag> at the start and append <\tag> in the end of the document.

Problem: Code failed to execute for some group members.

Solution: Set up Anaconda on Patas to ensure members in the group have controlled environment.

Reference

https://docs.python.org/3/library/xml.dom.html#dom-nodelist-objects

https://www.runoob.com/python/python-xml.html

https://docs.python.org/3/library/xml.etree.elementtree.html?highlight=etreea

https://lxml.de/api/lxml.etree._ElementTree-class.html