University of Southern California

Viterbi School of Engineering

CSCI 599: Content Detection and Analysis for Big Data

Instructor: Dr. Chris Mattmann

Assignment 3: Evaluating the Content Analysis Text Retrieval Conference (TREC) Polar Dynamic Domain Dataset

TEAM 22

GitHub repository: *https://github.com/harshfatepuria/Evaluation-of-Content-Analysis-on-TREC-Polat-DD-Dataset*

Github.io website: *http://harshfatepuria.github.io*

(All the visualizations with interactive capabilities available on this website)

Date: 05/03/2016

Report submitted by:

Harsh Fatepuria, *fatepuri@usc.edu*
Warut Roadrungwasinkul, *roadrung@usc.edu*
Rahul Agrawal, *rahulagr@usc.edu*
(Graduate Students, Department of Computer Science)

I. Examining the Polar Common Crawl Dataset

To examine properties of Polar Common Crawl Dataset, we develop a parser, CCADetailParser, to extract various properties of CBOR documents in the dataset as metadata, including

1. Requested URL and keywords extracted from the URL
2. Response HTTP status
3. Detected media type
4. Parser chain (the "X-Parsed-By" metadata field)
5. File size
6. Size of text retrieved
7. Size of metadata retrieved
8. Named entities recognized from the text using CoreNLP and 3 classes mode

II. Classification path from Request to Content

1. Using CCADetailParserRunner, we run this parser with a portion of the dataset (‘*572-team6-acadis-plain*’ and ‘*572-team41-ade*’ directories). We use the extracted information to study the relationship between URL keyword and extracted named entities. We develop visualizations of this relationship using D3. For readability, we will show 10 most frequent named entities of each class from 30 most frequent normalized keywords (stemming and stopwords removed using python *stemming* and *stop_words* library) and also from 30 normalized keywords that can extract highest amount of named entities.
2. Named entities present on the arrived at page: Person, Location and Organization

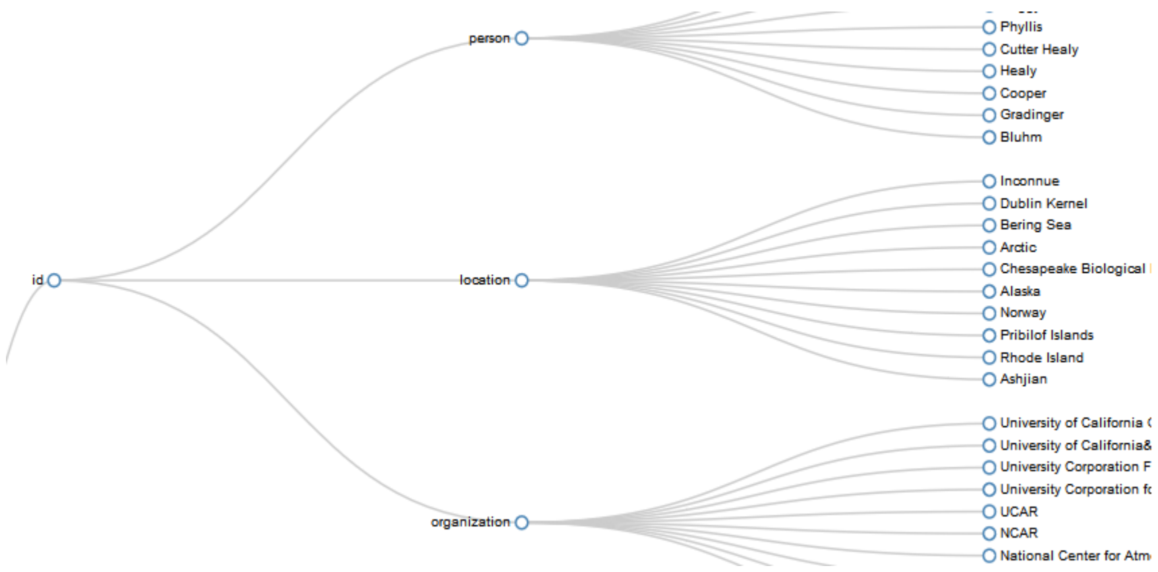


Fig 1: Showing 10 most frequent named entities of each class from 30 most frequent normalized keywords

3. Yes the crawler found the most relevant web pages. The request URL and the contents of the retrieved pages, and the named entities extracted had stark resemblance.

i.	To run the Parser: **java** ccaparser.CCADetailParserRunner *baseFolder resultFolder markerFile*
ii.	To create data for visualization: **python** urlKeywordToNer.py *ccaDataBaseFolder*

Snippet 1: Running the Parser for path from Request to Content

4. From the extracted data, we develop a script to summarize file size statistics of each file type. D3 visualization of this information is also provided.

i.	To create data for visualization: **python** sizeSummary.py *ccaDataBaseFolder*

Snippet 2: Generating visualization

III. File size diversity of Common Crawl (CCA) dataset by MIME type

1. The file size ratio of the Solr Index to the original file sizes were computed and averaged over 100 files from each MIME type. D3 visualization depicting the same is shown below:

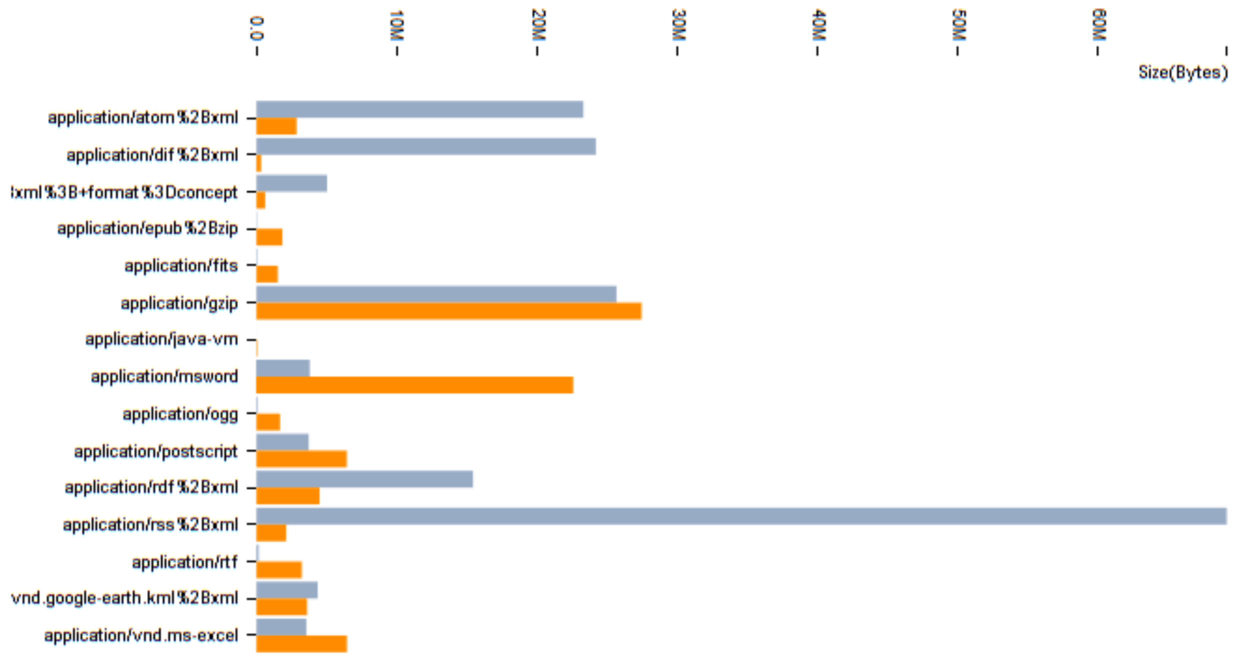


Fig 2: File Compression Ratio. (Orange line represents the actual file size, and the Blue line represents the size of its Solr Index)

```
i.    To create data for visualization: python sizeAnalysisSolr.py listOfAllFilesDirectory ccaDataBaseFolder
```

Snippet 3: Generating visualization data for Size Ratio Analysis

IV. Parser Call Chain

1. The CCADetailParser can be used with actual file by itself but it will not extract request information. We run the parser with entire Polar fulldump dataset using FileDetailParserRunner. D3 Interactive Sunburst can be used to visualize the parser hierarchy against other information. We develop the visualizations of parser hierarchy against number of document per file type, ratio of metadata retrieved to file size per file type, and ratio of text retrieved to file size per file type.

```
To run the Parser: java ccaparser.FileDetailParserRunner baseFolder resultFolder markerFile
To create data for visualization: python parserChain.py ccaDataBaseFolder fullDumpDataFolder
```

Snippet 4: Running the Parser for Parser Chain Call

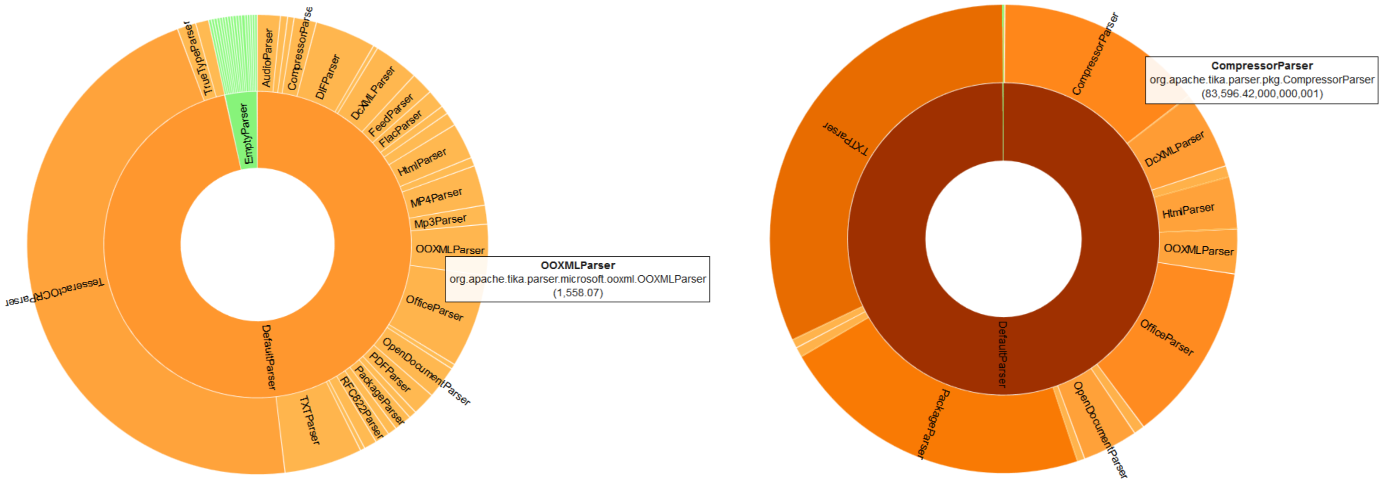


Fig 3: Amount of Text (left) and Metadata (right) retrieved per MIME type v/s parser chain hierarchy

2. Plot of the amount of Text retrieved per MIME type v/s Parser and the amount of Metadata retrieved per MIME type v/s parser chain hierarchy is shown above.

V. Language Identification and diversity

1. We will compare Tika language detection feature with Optimaize language detector library. We run this with entire Polar fulldump dataset and produce D3 visualization of the result. The visualization show that around 300k files are detected to be *lt-Lithuanian* by Tika but Optimaize only detect 42 files. Also, around 300k files are detected to be *unknown* by Optimaize but Tika only detect 8 files. We think this might be an outlier related to detecting empty file. We try to run the language detection again by filtering out the files that have too few text retrieved (<1000 characters). The result shows that number of documents detected to be *lt-Lithuanian* by Tika is significantly lower to 20k and number of documents detected to be *unknown* by Optimaize is lower to 7k. *That means amount of text will play an important role for language detection accuracy.*

```
To run the language detector: java language.LanguageDetectRunner baseFolder resultFolder minSize
To create data (csv) for visualization: python languageCompare.py languageDetectedFolder
```

Snippet 5: Running the LanguageDetectRunner

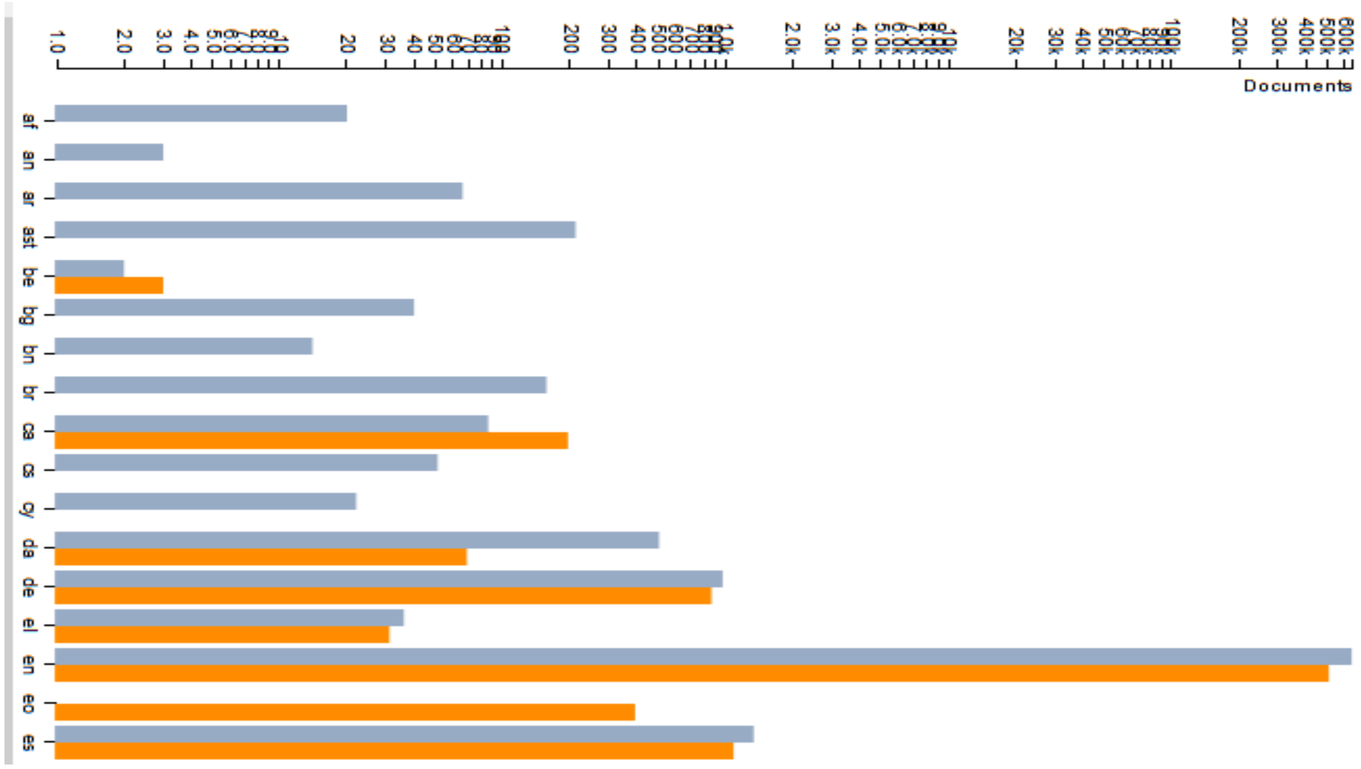


Fig 4: Language Detection. Orange line-Tika Language Detection and Blue line- Optimaize Language Detected (y-axis) v/s Number of Files (x-axis)

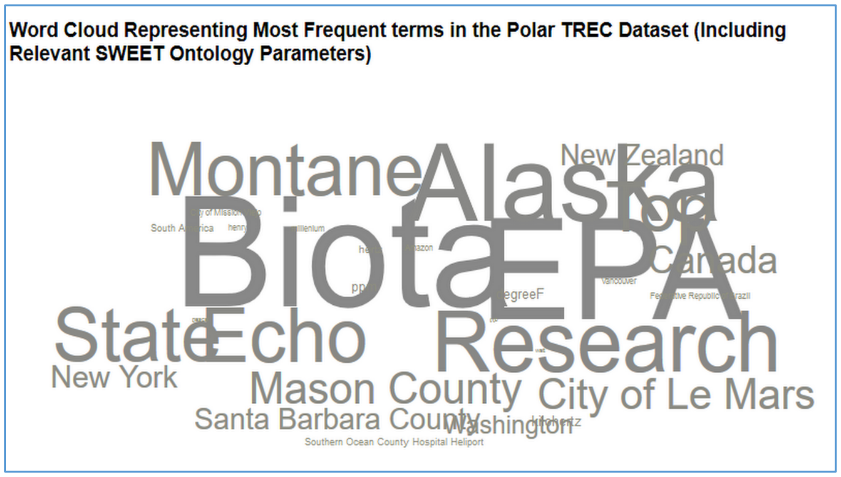
2. To study documents with mixed language, we will try to split extracted text into chunks (maximum at 10 chunks for a file) then use Tika language identifier to detect language of each chunk. We then compare detected language from full text to the majority of detected language of each chunks. If they are not the same, it will possibly indicate the file contains multiple language. We then make a visualization of the rate which language detected from parts and whole file are not the same. Though the rate is not much, it will be an evidence that some files in the dataset contains multiple language.

```
To run the mixed language detector: java language.MixedLanguageDetectRunner baseFolder resultFolder
```

Snippet 6: Running the MixedLanguageDetectRunner

VI. Maximal occurring topics in the dataset

1. After running various parsers built during the course of Assignment 2 and this project, including measurement parser, SWEET ontology parser, GeoTopic parser and other metadata extractors, a set of maximally occurring words were obtained from each of the parsers. The results are combined and presented in the form of Word Cloud below:


Word Cloud Representing Most Frequent terms in the Polar TREC Dataset (Including Relevant SWEET Ontology Parameters)
Montane
Alaska
New Zealand
Biota
EPA
Canada
State
Echo
Research
New York
Mason County
City of Le Mars
Santa Barbara County
Washington
Southern Ocean County Hospital Heliport

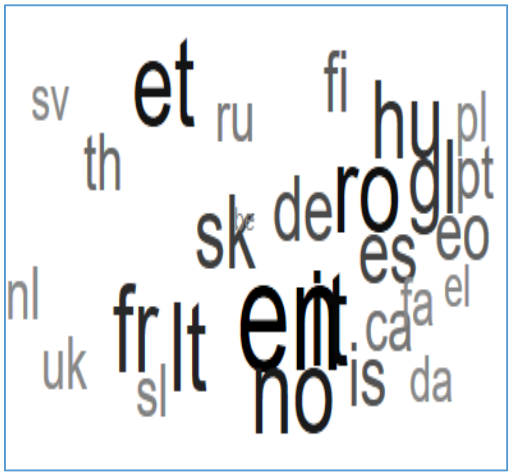


Fig 5: Word Cloud representing the maximal occurring topics including SWEET concepts(left), language diversity (right) in the dataset

2. Observation: Highest number of documents are in English. However, there is a great diversity of languages in the dataset. Also, since this is the Polar Dynamic domain dataset, we can clearly see that there is a large mention of places with huge ice shelves, namely, Alaska, Canada etc.

VII. Named Entity Recognition Agreement:

1. NLTK, CoreNLP, openNLP and Grobid Quantities are NER recognizers used to extract named entities for the data. After setting them up, we employ them in the CompositeNERAgreementParser. The NER agreement list can be obtained by accumulating most common named entities extracted from all 3 recognizers until the size of the list exceeds defined threshold. We can configure the parser whether to apply TTR analysis to the extracted text before doing named entity recognition. We run the parser with first 1500 files of each MIME type from the Polar fulldump dataset. D3 visualization is made to display 200 most common named entities extracted by the recognizers.

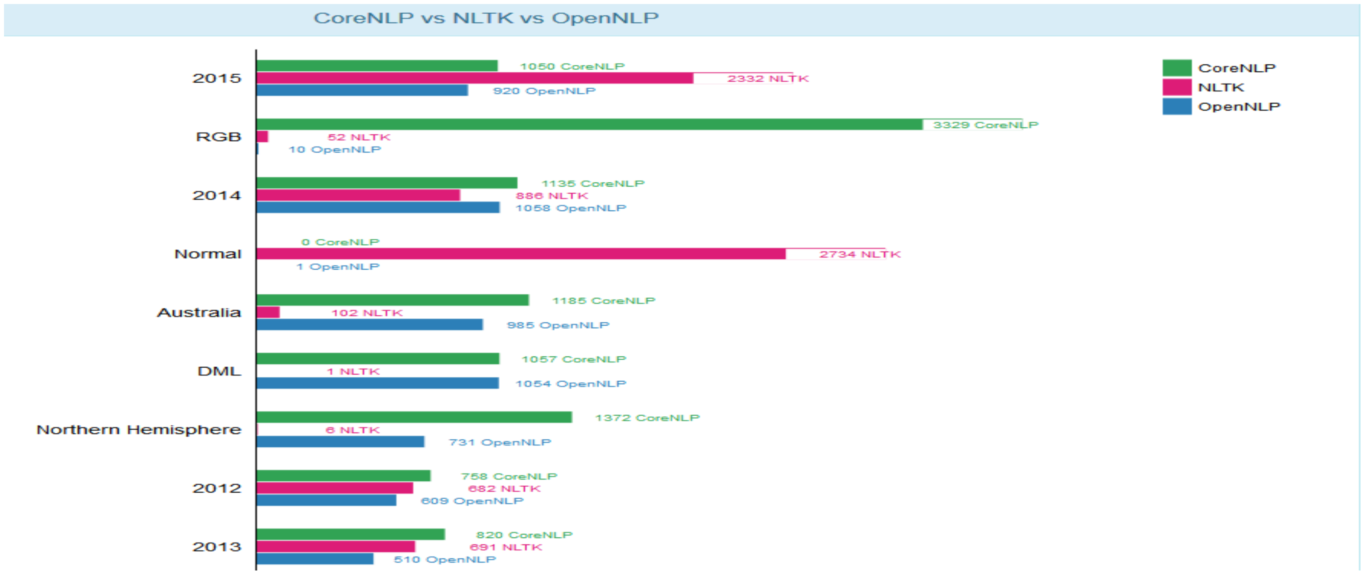


Fig 6: Maximal join agreement between the three NER toolkits.

```
To run the parser: java ner.NERAgreementParserRunner baseFolder resultFolder markerFile typeJsonFolder
To create data for visualization: python nerAgreement.py extrectedNERFolder
```

Snippet 7: Running the NER Agreement parser

Since entities extracted via Grobid Quantities Parser is not much in agreement with the above three toolkits, a separate visualization was developed for extracted measurements.

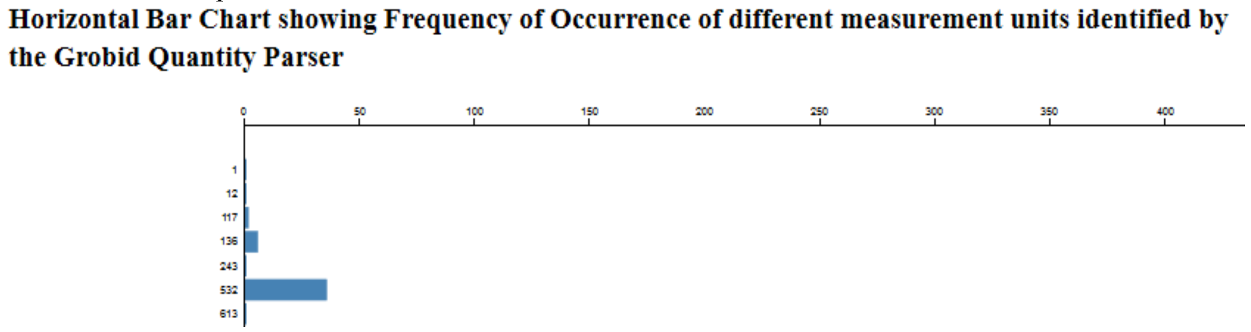


Fig 7: Grobid Quantities- extracted measurements and their counts (please see github.io website for more clarity)

2. [Extra Credit] Grobid Quantities:
   A Tika NER implementation that invokes Grobid Quantities via its REST service was developed. A pull request to merge the implementation is submitted.

```
To run the parser and create data for visualization: java ner.GrobidQuantitiesNER
```

Snippet 8: Running the Grobid Quantity parser implemented by us

3. Our new joint agreement (as described in NERAgreementParserRunner.java file) produces updated NER for the metadata records. We added the new maximal joint agreement NERs to the metadata records in our Solr Index, which can be seen in the attached Solr schema file.

VIII. Spectrum of Measurements

1. A program (measurementParser.py) was developed which takes the various measurement extractor entities- developed as a part of Assignment 2, and the measurements extracted using Grobid Quantities and generates JSON in three formats:
   i. Max, Min and Average for all the different measurement units.
   ii. All measurements clustered based on domain: A dictionary of high level domain and related measurement units was manually and meticulously curated. A sample entry in dictionary looks like:

```
{
"TEMPERATURE":["kelvin", "celsius", "celcius", "fahrenheit", "degreeC", “degreeF” … ]
}
```

   iii. Measurements clustered based on the MIME types: A script was developed to map each file in the dataset to its MIME type. This was generated using a list generated as a part of Assignment 1 which maps each MIME type to the list of file-paths it refers to. Then the measurements extracted for each files were clustered based on the MIME type of that file. A Sample JSON for the D3 visualization is shown below:

```
{
“name”: Measurement Range Clustered on MIME Type”,
"children": [{
        “name”: “application/x-sh”,
        "children": [
              {"name": "hour", "size": 3},
              {"name": "kelvin", "size": 2},
                    {"name": "month", "size": 1}
                     …
              } … ]
            …
}
```

Snippet 9: Mapping Domain to Units (ii) and JSON file for D3 generation (iii)

2. Various visualizations for the above generated spectrum of measurements were created:

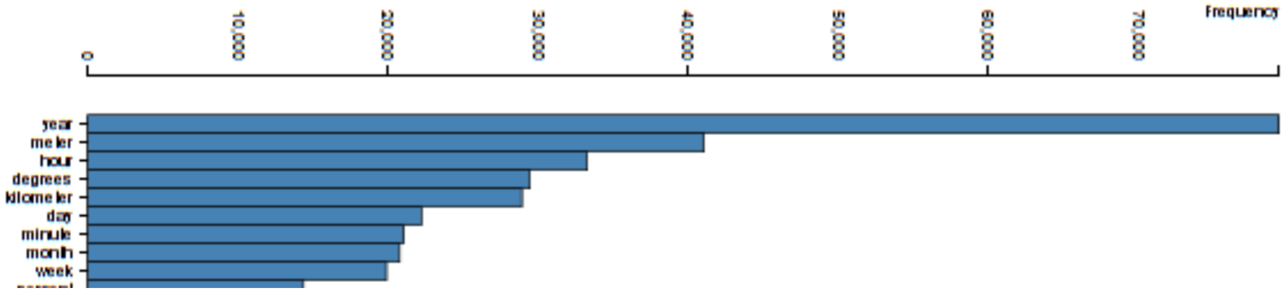


Fig 8: Bar Chart depicting extracted measurement units and their frequency

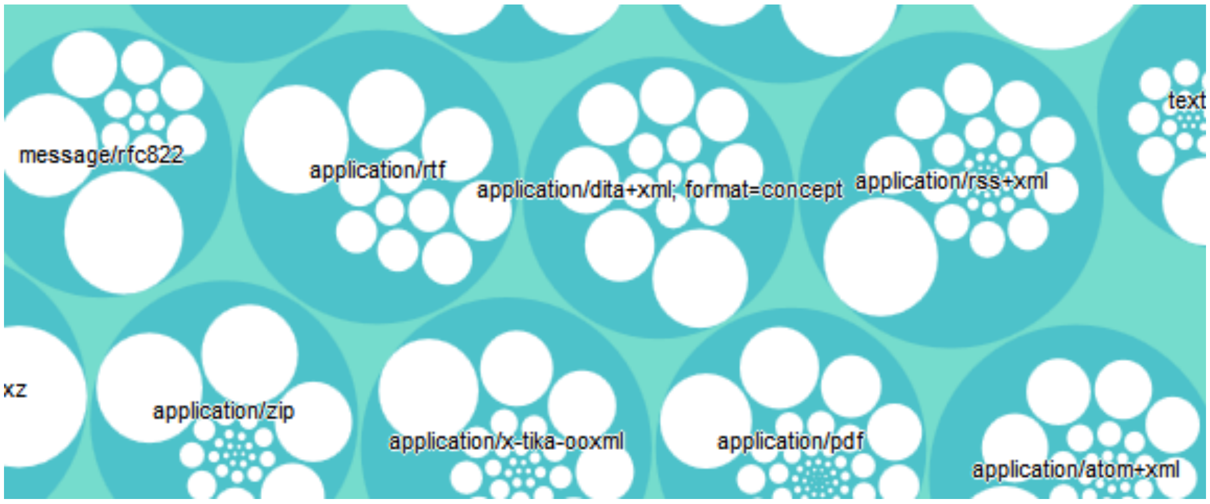


Fig 9: Circle Packing showing Measurement units clustered based on MIME types. Note that the data has been normalized for visualization purposes. See Github.io website for more clarity.

```
To run the parser and create data for visualization: python measurementParser.py
```

Snippet 10: Running the Measurement parser implemented by us to get data for visualizations

3. Observation: In MIME type application/google-earth, the maximally occurring key word is 'degree' referring to longitudinal and latitudinal minutes with an occurrence of over 95%. Overall, in the whole dataset, 'year' is the most used unit, double the count of second most used unit, 'meter'.

IX. Contributions to http://www.polar.usc.edu and Video Demonstration:

1. We have previously contributed our visualizations to polar.usc.edu in both, Assignment 1 and Assignment 2. We have also added a pull request for the visualizations developed as a part of this assignment.
2. Video demonstrating a screencast walk-through to the tasks is available at: https://youtu.be/lChEX_gn65g

X. D3 Visualizations:

The D3 visualizations can be found in the different directories present in the *result* directory of the submission.

Visualization File Name	Description
UrlKeywordToNer-MostFrequentKeyword.html	Most frequent URL Keywords vs Extracted NER
UrlKeywordToNer-MostNerExtracted.html	URL Keywords that produce most NE vs Extracted NER
SizeRatioSummary.html	Size Ratio of Solr Index to Original File Size (Grouped By MIME Type)
SizeSummary.html	File Size Summary
ParserVsCount.html	Parser Hierarchy vs Document Count by Type
ParserVsMetadataRetrieved.html	Parser Heirarchy vs Size of Metadata Retrieved Ratio by Type
ParserVsTextRetrieved.html	Parser Heirarchy vs Size of Text Retrieved Ratio by Type
ParserVsTTRRetrieved.html	Parser Heirarchy vs Size of TTR Text Retrieved Ratio by Type
LanguageCompareFilter.html	Number of Documents by Detected Language (filtered out short documents)
MixedLanguage.html	Tika Mixed Language Detection
WordCloudD3.html	Word Cloud Representing Most Frequent terms in the Polar TREC Dataset (Including Relevant SWEET Ontology Parameters)
GrobidQuantityTTR.html	Horizontal Bar Chart showing Frequency of Occurrence of different measurement units identified by the Grobid Quantity Parser
NERAgreementFullText.html	CoreNLP vs NLTK vs OpenNLP
Measurement_ByDomain.html	Zoomable Circle Packing D3 Showing vaious Measurement Quantities(Grouped By Domain)
Measurement_ByMIME.html	Zoomable Circle Packing D3 Showing vaious Measurement Quantities(Grouped By MIME Type)
measurementCountBarChart.html	Bar Chart showing Frequency of Occurrence of different measurement units
Measurement_Range.html	Dendogram showing Minimum, Maximum, and Mean values of Measurement Units

XI. Important Observations:

1. Yes, our MIME detection was good. We used Apache Tika's inbuilt parsers to parse the dataset, and categorize the data into MIME types. We also ran Mac Operating System's inbuilt MIME detector using the command: *file --mime-type -b <file_name>*
   Not surprisingly, **almost 99% of the files were detected the same**. There were certain files in the application/octet stream which were not detected by Tika, but were identified by Mac OS as text-plain.
   Out of around 1.7 M files in the dataset, around 200,000 files were detected to be application/octet stream, which amounts to an accuracy of 88% correct detection.

2. Yes, our parsers are extracting the right text. During the course of the three assignments, we built many parsers like measurement extractor, geo topic parsing extractor, used MEMEX geoparser etc. Also, we developed many wrappers for toolkits like openNLP, Core NER, NLTK and Grobid Quantities. These parsers are extracting the correct text, which is evident from the fact that they are in agreement to each other. More details for this agreement can be seen in the visualizations provides.

3. Tika identifies the correct parser most of the time to parse any input file. This provides us with the required and relevant data from each input file. This is evident from the Parser Hierarchy Visualizations created by us.

4. The Metadata identified by Tika is highly appropriate, as seen from the metadata scores obtained in Assignment 2. We try to enrich the metadata using various techniques taught to us in class. All these new metadata parameters were dumped in Solr with respect to each file we worked with. These new metadata fields include geolocations, SWEET Ontology concepts, related publications (for PDF files), and measurement units present.

5. We compared Tika language detection feature with Optimaize language detector library. The visualization made by us show that around 300k files are detected to be *lt-Lithuanian* by Tika but Optimaize only detect 42 files. Also, around 300k files are detected to be *unknown* by Optimaize but Tika only detect 8 files. We think this might be an outlier related to detecting empty file. We try to run the language detection again by filtering out the files that have too few text retrieved (<1000 characters). The result shows that number of documents detected to be *lt-Lithuanian* by Tika is significantly lower to 20k and number of documents detected to be *unknown* by Optimaize is lower to 7k. *That means amount of text will play an important role for language detection accuracy.*

6. The Named Entities extracted using OpenNLP, CoreNLP, NLTK, and GrobidQuantity are in agreement to each other, and hence make sense. The terms extracted are also in sync with the relevant topics of the TREC Dynamic Polar Dataset.