CSCI 599 Assignment 2

Team 9

28 March, 2016

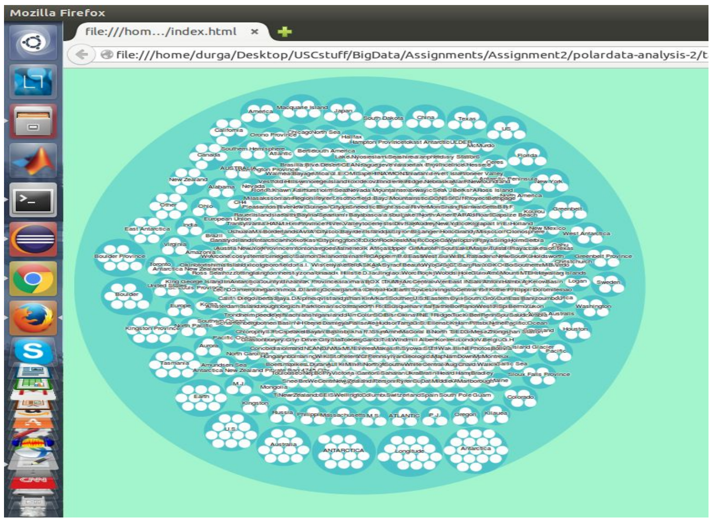
Part 3: Tag Ratio Algorithm with Named-Entity Recognition for Measurements

The tag ratio algorithm helps us extract useful text from HTML- like documents by identifying lines in the text that contain information. To enable applying tag ratio to all mime types and extract measurement information, we do the following steps:

1. Convert the file content to a XHTML file stream using Tika's AutoDetectParser
2. Preprocess the content by removing style, script and comments sections
3. Calculate no.of characters/no.of tags for each line and if this ratio is above 1, we consider it to have information.
4. The content from each file is now given to CoreNLP's 7-model parser to get locations
5. It is also passed through a Regex Parser built for identifying measurements that is built using RegexNERParser under tika-parsers. The regex used for measurement is:

```
MEASUREMENT=
(((\d)+(\.)?(\d)+)((\s)+)((?i)((square)|(cube))?((mm)|(cm)|(km)|(mile)|(mile)|(celcius)|(farenheit)|(c)|(f)|(kelvin)|(k)|(inch)|(inches)|(meter)|(metre)|(foot)|(feet)|(litre)|(gallon)|(pound)|(l)|(ml)|(minute)|(second)|(millisecond)|(hour)|(day)|(year)|(month)|(week)|(amp)|(v))(s?))((\s)+))
```

The location and measurements values are correlated using the visualization below. It is an aggregate depiction of location-wise measurements we found in the polar dataset.


Mozilla Firefox
file:///hom.../index.html
file:///home/durga/Desktop/USCstuff/BigData/Assignments/Assignment2/polardata-analysis-2/t

Part 4: DOI Generation using URL Shortener method

We iterate over all the files and assign DOI to each of them. To ensure that its unique, we maintain the file and its DOI in a hashmap. To generate a DOI we form a string 'a' of length 10 from the character set [a-zA-Z0-9]. We use polar.usc.edu as the prefix, the final DOI generated is of the form polar.usc.edu/str(a).

We then generate a json file which holds the filename and its assigned DOI, which is used in the further steps.

As an example, for the file F633C197BD5DF8151163808E4A5E7D4DD6D4EAEA5279929E536DB282B95702AC the DOI generate is polar.usc.edu\/wRqNuLliZj.

Part 5: Grobid Journal Parser, scholar.py

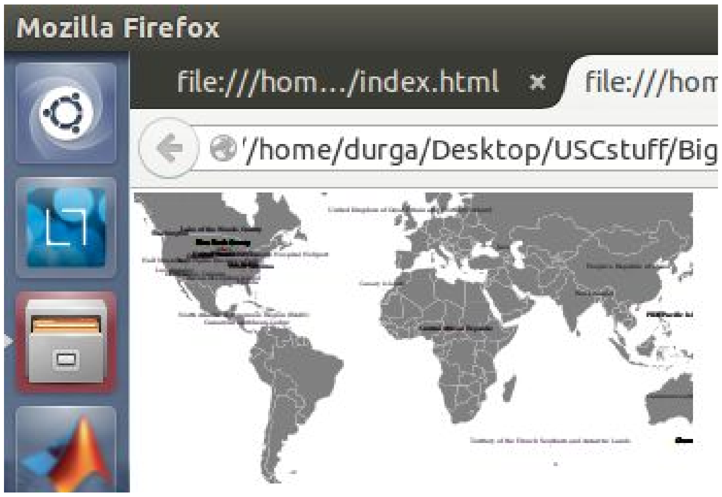
PDF files found in polar data set is parsed using through Grobid Journal parser.
From the parsed data we extract tei annotation and author information.
TEI annotation output:
:https://github.com/durgaravi/polardata-analysis-2/blob/master/grobid_scholar/tei_extract.txt
The extracted author information is passed to google scholar api. The returned result is transformed in the form of json output.
Json output
:https://github.com/durgaravi/polardata-analysis-2/blob/master/grobid_scholar/scholar_output.json

Script python script for the above functionality :
https://github.com/durgaravi/polardata-analysis-2/blob/master/grobid_scholar/grobid.py

Part 6: GeoTopic Parser

On placing the en-ner-location.bin in the classpath of tika-parsers and running the lucene-geo-gazeteer, the GeoParser class in Tika is made available for service. On parsing the files as InputStream using GeoParser.parse, the metadata gets additional fields Geographic_NAME, Geographic_LATITUDE, Geographic_LONGITUDE.

We build a CSV for D3 visualization and a JSON to load to Solr with a mapping to the DOI id generated in step 4. The visualization we use is a map histogram that highlights the locations in polar data and onclick directs to a google search on the location.



Part 7: Sweet ontology Parser

We first selected 5 owl files from the set of sweet ontology files. The owl files which we selected are Matter.owl, PhyProcess.owl, PhyState.owl, MtrWater.owl and Mineral.owl. We then trained a model to detect these concepts using Stanford NLP CRFClassifier.

We then created a tika parser similar to the existing CoreCLPRecogniser, named SweetOntologyParser which loads the classifier and then detects the sweet concepts from the file. We created a json to hold the files corresponding to each sweet ontology concept, and

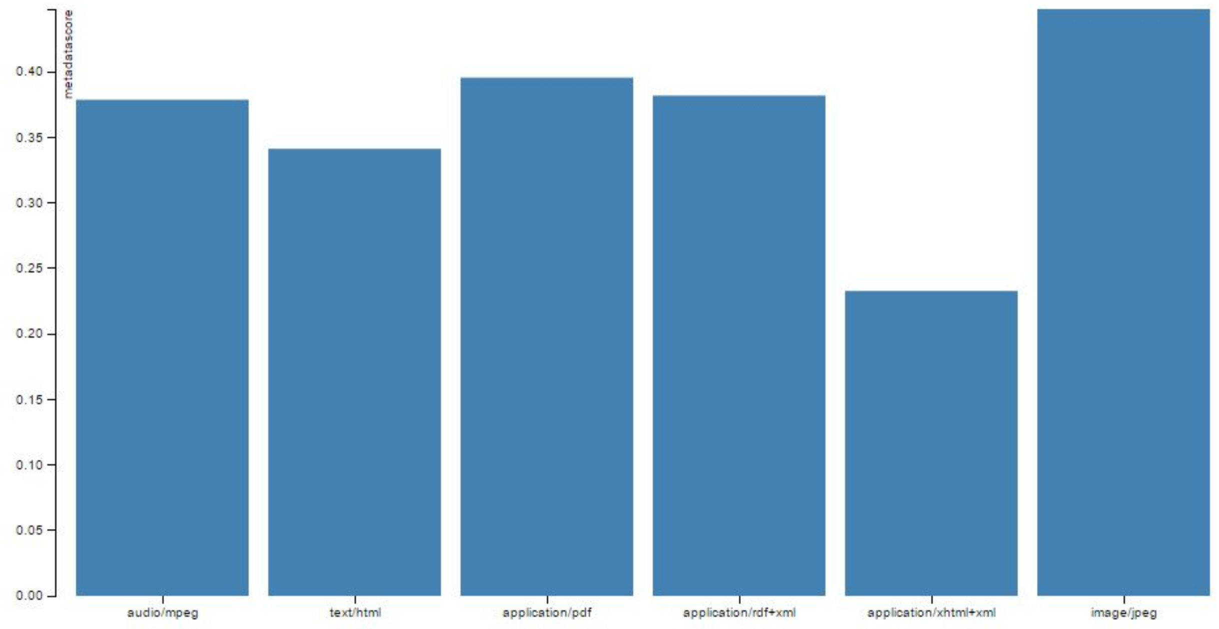
Part 8: Metadata scoring

Metadata of the files are extracted. Metadata attributes are checked against dublin core attributes. For every attribute present score of 1 is assigned for metadata score of the file. Once the metadata score is calculated across all files for a particular format, scores are normalized. Once metadata score is normalized average score is calculated for that particular file.
Program :
https://github.com/durgaravi/polardata-analysis-2/blob/master/MetaDataScore/MetaDataScore.java

Average Normalized Score for
audio/mpeg , text/html, application/pdf, application/rdf+xml, application/xhtml+xml,

Link:
https://github.com/durgaravi/polardata-analysis-2/blob/master/MetaDataScore/data.tsv

Visualization :



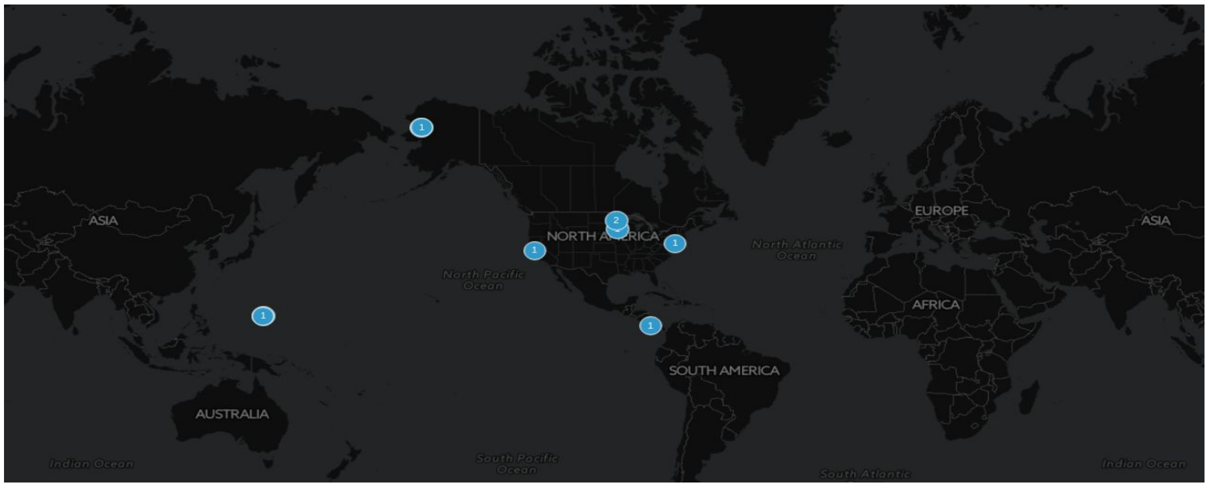
Part 9: Solr upload

1. Posting and indexing files: We create a collection polarcollection2 in our solr instance and use SolrJ library to post files via Java that uses Tika to extract metadata field values and store into tika. To load with DOI id, we set the literal.id field.
2. The NER Measurements, Geotopic data, Sweet features and publication data which has the DOI id along with it is used to read and load using SolrPy module in Python. This code updates the documents with additional metadata fields.

Part 10: GeoMemex Parser

The GeoMemex parser requires us to run lucene-geo-gazeteer on port 8765 as well as the tika-server on port 8001 which is done by placing TikaServerCLI class on the current classpath and using the -p option to use a custom port.

For the data, we use geotopic.csv generated in GeoTopic parser step (task 6). This csv has location and lat-long data that is identified by geo memex parser. It is placed in geoparser_app/static/uploaded_files so that a new collection in Solr is created with the location data.
On running the goe memex parser on port 8000 and accessing the server through http://localhost:8000, we get a map with location counts as shown below:



Part 11: Tika Similarity
We chose value-similarity.py to run our metadata maps by modifying the dictionary that has metadata values from tika-parser to our metadata. We compare the field values of differnet files and cluster the files based on our metadata score. The result files can be found here:
https://github.com/durgaravi/polardata-analysis-2/tree/master/Ravichandran_Durga_CSCI599_HW_CE_MET_NER/documentation

Part 12: Visualization

1. Named entity Recognition: location-measurements
2. Geotopic Parser: Location histogram
3. PieChart: Sweet ontologies
4. Metadata score: Bar chart on mimetypes vs metadata score
5. Named entity Recognition: organization-measurements
6. Grobid: tree visualization

Part 13: GeoMemex Parsing with solr indexing

On adding our Solr index with http://localhost:8983/solr/uploaded_files, our solr data indexes are used to visualize data and the search facility highlights on the map in dark blue as shown below:

NORTH AMERICA
EUROPE
ASIA
AFRICA
SOUTH AMERICA
AUSTRALIA
ANTARCTICA

Part 14: FFMPEG Parser

We selected the FFMPEG parser over the available parsers. This parser provides video specific information

```
prasadns14@ubuntu:~/Downloads$ java -classpath tika-app-1.12.jar org.apache.tika.cli.TikaCLI -m 1D900F9F3F1DEB99CF9A7071DB843198D2C3A9617EED4A90E2D65AFD13C72742
Content-Length: 62530
Content-Type: video/mpeg
X-Parsed-By: org.apache.tika.parser.DefaultParser
X-Parsed-By: org.apache.tika.parser.external.CompositeExternalParser
X-Parsed-By: org.apache.tika.parser.external.ExternalParser
resourceName:
1D900F9F3F1DEB99CF9A7071DB843198D2C3A9617EED4A90E2D65AFD13C72742
xmpDM:duration: 00:00:00.13
xmpDM:fileDataRate: 3748 kb/s
xmpDM:videoCompressor: mpeg1video
```

This extra information gives insights on the quality of the files, the video and audio compressor used which can help us in determining the suitable applications to play these video files. The FFMPEG parser returns these extra metadata only for mp4 and mpeg formats and has no support for the older formats.

Useful features in polar dataset

1. Location: The location data in polar dataset is very rich and predominantly spread in North America. It is useful to understand that most work in the area is done in the United States/ North America region.
2. Publications: The PDF files are publications and interesting insights on related files based on author and content rather than mimetype.

3. Texts: Using
4. Dates: NER parsing on dates show about the timeframe along which the files have been generated

NER tag ratio: how tag ratio was useful and how it helped
Tag ratio was useful in trimming unwanted tag contents from the XML tika-parsed handler string and helped extract body text for all file types. The lines with higher tag ratio indicates high content availability and thus the size of text to be processed minimised and helped better named entity recognition.

Did NER and SWEET terminology mapping work well – was the NER unable to identify SWEET categories and concepts?
We created a custom CRF model, for the sweet ontology concepts and then used the CRF classifier to detect the sweet concepts from the files. The model was able to detect the sweet concepts to an accuracy of 60-70%. By using more data for model training we can achieve better efficiency.

Did the D3 interactive visualizations help you understand the data?
The visualizations were very insightful.

1. **Named entity Recognition: location-measurements:** This visualization gives us an insight on the kind of measurements we see per location. It also gives us an idea of the locations in the data set.
2. **Geotopic Parser:** A location that appears large number of times is highlighted indicating a prominent location. Also, the latitude longitude prominenet range can be found.
3. **PieChart: Sweet ontologies:** The pie chart gives a list of file ids that are associated with the higher level ontology.
4. **Metadata score: Bar chart on mimetypes vs metadata score:** This visualization helps us to understand how mimetypes and their metadata score are related and which mimetypes have large scores. In the current dataset, image/jpeg got a high metadata score of 0.4.
5. **Named entity Recognition: organization-measurements:** This visualization gives us an insight on the kind of measurements we see per organization. It also gives us an idea of the locations in the data set.

Were particular features that you extracted such as the geo-locations more effective in producing clusters?
With our metadata features, the tika similarity made higher number of clusters. Previously, using Jaccard similarity, there were 7 clusters formed. With the current

metadata having location as metadata, we have 14 clusters with related files clustered together. All related pdf files were seen to be clustered effectively with this.

Were particular cluster techniques e.g., k-means, more meaningful than hierarchical clustering? What about distance metrics – which ones were more effective (Jaccard, Edit Distance, etc.) Why?

We used k-means clustering since the data has numerical metadata values that can be used for clustered as compared to hierarchical clustering. With k-means, it was easy to specify ner, location and sweet concepts attributes as data and on applying Jaccard distance with this, the results were fast.
Jaccard similarity is a more suitable measure since edit and cosine distances are string based comparison measures and the metadata values we hava can be considered as attributes rather than text. Also, since Jaccard distance compares file with all data files rather than one against another, the process is made faster.

Was your metadata quality score something that you could leverage to find richly curated records and ultimately is it something that could be leveraged to point users to the more meaningful polar data? Results of metadata score filtering??
From the result it can be observed that normalized metadata score can be used point towards a certain file type. For example if the normalized metadata score of a file is in the range of 0.35 - 0.4 its most probably indicative that its audio/mpeg, application/pdf application/rdf+xml and above 0.4 image/jpeg file format.

Were you able to find related scientific publications, and did the authors you found both inside the dataset and using Google Scholar have a high degree of overlap with the existing Polar dataset?
The result varies. For files where authors are present there are common result.
But when it comes to multiple Authors present in a document the degree of overlap in the result mostly decreases.