**Task 14 - Extra credit:**

From the list of features mentioned here: <http://wiki.apache.org/tika/> , the ffmpeg with tika content extraction has been chosen for this assignment. The ffmpeg is a framework which can decode, encode, transcode, mux ,demux, stream, filter and play pretty much anything that humans and machines have created. The ffmpeg parser with tika can be used for extracting additional information of the file.

a. Why did you chose the Content Extractions?

This was chosen as we had considerable amount of audio and video files which had poor metadata information.

b. What additional knowledge did you gain from the features?

On enabling this parser and then running tika, additional metadata information for certain files could be seen. A program for running the tika metadata extraction has been written for this purpose. The details regarding the same have been included in the project folder FFMPEG:

*Run\_ffmpeg.java* -> The program to call tika and get extracted metadata and write json output.

*Tika\_with\_ffmpeg.json* -> the output after running tika with ffmpeg on the set of files which were audio and media type (a program was written for the purpose of segregating the files according to their mimetype after which the above java program is run).

*Tika\_without\_ffmpeg.json* -> the output after running tika without ffmpeg on the same set of files as above.

Exceptions and Warnings faced:

Illegal IOException from org.apache.tika.parser.video.FLVParser,

WARNING: PooledTimeSeries not installed! ( The pooled Time series video descriptor with tika was not used here , that is why).

**Task 9 - Solr:**

The extracted metadata and other information extracted (by performing Task 3-8) of documents has been indexed.

1. We chose Solr as our indexing technology.
2. The documents have been segregated into folders according to their mimetypes.All the extracted and additional information for all the documents have been bundled into a json under the name of the specific mimetype. This is for handling indexing in an organized fashion and helps in keeping track of what types are indexed during the process.A program for indexing has been developed where each <mimetype>\_output.json is fed to this and the the indexing is performed by calling solr using a curl command of the following fashion:

curl 'http://localhost:8983/solr/update?commit=true' --data-binary @text\_plain\_output.json

-H 'Content-type:application/json'

1. A solr schema has been developed with specific fields which have been identified - the ones extracted from tika like content-type, resourceName, etc… and few others which are the metadata which were adding as result of performing tasks 3-8 like location, author, affiliations, doi, etc… A visualization of the schema and fields is in the file:

visualizations/*index\_solr.html*. The nesting shown for fields is just for visualization purpose.

**Task 3 - TTR+Measurements:**

1. A tag ratio content handler was created to get the areas of text with important information. The tag ratio algorithm used here is based on the fact that the area of the document where more text is contained between an opening and closing tag has relevant information. We ignore the script, style, comment, head and link tags throughout in the document.
2. TTR algorithm - The body content of the document is passed to a function which tokenizes and sets bit array to 1 or 0 depending on whether the token being parsed is a text or tag. This bit array is then converted to integer array by making current indices’ value as the sum of current element and the previous element. The goal is to find the sub-arrays with maximum character tokens. Every element in the inner loop (j=i+1) is compared to every element in the outer loop (i), i and j representing the indices of the integer array. In the inner loop, the current max chunk along with the i and j values of the chunk is updated by adding the tagAfter(a[n]-a[j]),tagBefore(a[i]) and tagBetween((j-i) - (a[j]-a[i]) values and comparing it with the recent max chunk. The final chunk got after the outer loop iteration is complete gives the relevant content of the document.
3. Measurement extraction - The measurement extraction content handler takes the above output, passes it through an exhaustive regex containing all possible SI units of measurement from which the unit value(number), the unit, the parent category and the subcategory of the units are matched and extracted by performing regex named group matches.
4. All the extracted measurement objects are returned as output from this function.

**Task 4 - DOI+ShortUrl:**

1. After gaining knowledge regarding DOI formats from lecture, the DOIGenerator Content Handler was formed. It uses the apache commons codec and generates the base64 encoded string from title, resourceName, author, description, url of a document’s metadata.
2. The base64 encoded string generated in above step is passed to the url shortener (YOURLS API) to obtain the final DOI format url: polar.usc.edu/10.2432/<encodedstr>.

The Java program to run the above content handlers along with tika was written and is present in the project folder TTR\_DOI\_Call. The program takes in a directory, recursively reads each file in the directory and does the following : Auto detect and get parsed xhtml output from tika -> Call TTRContentHandler and get relevant content -> Call MeasurementContentHandler and get list of measurement objects -> Call DOIGeneratorContentHandler and get base64 encoded string -> Pass it to YourlsShortener and get the short url -> Form json arrays for measurements,doi add to json object along with content-type and title and append to json file. Finally, Obtain json containing metadata of all the files. The directory passed contains the categorized polar dataset according to the mimetypes and one json output is formed for each mimtype: <mime>\_output.json.

Related files in the project folder:

*TTRContentHandler,MeasurementExtractingContentHandler,* *DOIGeneratorContentHandler* -> Tika-master (tika-master\tika-parsers\src\main\java\org\apache\tika\parser\ttr)

*Run\_Tika, YourlsShortener* -> TTR\_DOI\_Call\src

Visualization: A measurement visualization has been created representing the category, sub cateogries of SI units along with the number of files the measurement types are present in.