Youtube Video Demonstration

https://www.youtube.com/playlist?list=PLfsGiW3iUfri1L7eCM-N1FYHjh1IsFTZZ

1. Is your MIME detection good? Define “good”.

In this assignment, Tika AutoDetectDetector was used, unlike more reliable content-based file fingerprint method followed in the first assignment.

We used Tika MIME detection several times in the last assignment. We think it’s pretty useful, since there are a lot of different files of dozens of types. We detected MIME type first, and then performed different analysis tasks on the files of different types in both CCA and raw Polar datasets.

We believe Tika MIME detection did good as we didn’t encounter any major parser exceptions, which may be raised if we use wrong parser on mime-type which it doesn’t support

==

2. Are your parsers extracting the right text? Define “right”.

Following are exceptions encountered captured in parserCallChainError.json for varied media types, with the filenames

image\/png parse error
TIKA-198: Illegal IOException from org.apache.tika.parser.video.FLVParser@3406472c
Unexpected RuntimeException from org.gagravarr.tika.TheoraParser@4e31276e

Following is a summary of Parser Call Chain statistics

mime_type	**: Parser**	**: Content**	**: MetaData**
application/fits	: gdal.GDALParser	: mostly 0	: some trivial metadata
application/msword	: microsoft.OfficeParser	: 40% max	: useful Author,Title,Date fields
application/atom+xml	: feed.FeedParser	: 60% max	: more than 90% data with useful description info
application/gzip	: pkg.CompressorParser	: mostly 0	: some trivial metadata
application/java-vm	: asm.ClassParser	: 2% max	: some trivial metadata with resourceName fields
application/rtf	: rtf.RTFParser	: 40% <1KB fileSize 2%max >1KB fileSize	
application/rdf+xml	: xml.DcXMLParser	: 70% max	: 30% in some instances
application/rss+xml	: feed.FeedParser	: 50% max	: more than 90% data with useful description info
****application/postscript	: parser.EmptyParser	: 0 bytes	: some trivial metadata

4_output (CAN BE IMPROVED IF WE CAN DEVELOP/INTEGRATE A PARSER WHICH FETCH LYRICS AND ADD TO METADATA)

audio/x-ms-wma : parser.EmptyParser : 0 bytes : some trivial metadata
audio/x-flac : tika.FlacParser : 0.5% max : GOOD but LESS info including album artist, title
audio/x-wav : audio.AudioParser : 0% : some trivial metadata
audio/mpeg : mp3.Mp3Parser : 1% : uses XML Dynamic Media model, and some hardware specs, like audio sample rate

5_output

image/png : image.ImageParser : 0% :>100% metadata useful including Palette, Image Orientation, RGB, Pixel Aspect Ratio
image/vnd.adobe.photoshop: image.PSDParser : 0% :less metadata includes color mode, image length and width
image/png : image.ImageParser : 0% :>100% metadata

6_output (CAN BE IMPROVED IF WE CAN DEVELOP/INTEGRATE A PARSER WHICH FETCH SubTitles AND ADD TO METADATA)

video/x-m4v : mp4.MP4Parser : 0% : some trivial metadata
video/x-ms-wmv : parser.EmptyParser : 0% : some trivial metadata
video/mpeg : parser.EmptyParser : 0% : some trivial metadata
application/mp4 : mp4.MP4Parser : 0% : some useful metadata date fields
video/x-ms-asf : parser.EmptyParser : 0% : some trivial metadata
video/quick-time : mp4.MP4Parser : 0% : some useful metadata date fields
video/x-msvideo : parser.EmptyParser : 0% : some trivial metadata
video/mp4 : mp4.MP4Parser : 0% : some trivial metadata

7_pdf_output

application/pdf : pdf.PDFParser : 75% avg : 55% good amount of metadata

==

3. Are we selecting the right parser? Define “right”.

We believe a "right" parser is one which can extract useful knowledge from data represented in resource.
Rather than picking one parser, we believe a Collective Parser which combines the knowledge(in terms of text and metadata) from family of related parsers is the best parser.
Though Tika supports Composite Design Pattern with CompositeParser, we feel a sort of Collective Design pattern would be able to derive best knowledge from electronic resource.

We cannot decide if we are selecting the right parser, without comparing results from a family of related parsers

Hence we propose Tika to have an ability to provide an interface where users can compare output from set of parsers.

This feature may be a feather in Tika's cap, with its powerful unified parser interface.

Tika can also be standard platform for people developing parsers for different media types, to compare their parsers with existing state of open source parsers.Tika- A BenchMark Tool
==

4. Is your Metadata appropriate? What’s missing? You can use your Metadata score
generated from assignment #2 here, and also your results from this assignment.

Metadata from assignment #2 represents Summarized info to an extent

(sweet ontology concepts, measurement mentions, grobid bibliographic info for pdfs, related publications fetched from scholar API)

These Metadata can be instrumental in representing summarized info of a resource and improve search engine performance, by representing user with useful metadata,which user can leverage to decide whether to use e-resource or not. For most of the files, it is appropriate according to dubin core, a model for metadata,it can show appropriate materials (a description field, title or version exist in metadata), it supports interoperability, allow for long-term management (creating index for all files), identifies terms and conditions. We use the metadata score to filter the dataset and set a threshold for them, after filtering, meaningless files with low metadata score or empty files will be abandoned, we just use the remaining files to do analysis.
==

5. How well is my language detection performing? Comment based on the diversity of the
languages derived in this assignment. Are there mixed languages? Did it affect your accuracy?

We leveraged Tika-default language detection using n-grams approach without using isReasonablyCertain flag which checks if the distance between language profile of input text and that of different languages < 0.022D

We used Universal Declaration of Human Rights (UDHR) sample documents, and our approach, which detected the right language though distance measure was below the threshold value

{"ru":["rus.pdf"],"en":["eng.pdf"],"fr":["frn.pdf"],"es":["spn.pdf"]}

When we used OptimaizeLangDetector the results were not good, all the 3/4 reference documents rus.pdf,frn.pdf,spn.pdf were classified as th: Thai [refer OptimazeResults.txt of our submission] and only eng.pdf was detected as english

Hence we used our approach and captured a broad diversity 28 languages and 1 unknown category.

However diversity obtained has to be compared with other approaches to determine accuracy
==

6. Do your Named Entities make sense?

Based on NER extraction, we can add more meaningful metadata and update metadata fields for the file

Sure, it makes sense. We tried four different methods such as opennlp, nltk, corenlp and grobid quantities.

For the first three name entity methods, I get seven types of entities: person, location, date, time, money, organization, percentage.

We first use tika to extract the file, then text is got.

Use the extracted text as the input to these three ner methods. for opennlp, text are divided into sentences, then get tokens

to run the algorithm. for nltk, all type of entities by recognizer, for corenlp, the indices of entities and type are returned.

For grobid quantities, we used GET/POST to get the measurement_numbers, units, measurements and normalized_measurements.

After running them, we checked the outputs. Though some meaningless things returned, most of them are extracted correctly. However, sometimes corenlp is a little confusing, because it return the indices of a particular type, if some entities of the same type appear continuously, it’s difficult to split them, because some entities may consist of more than one word.

We used D3 to do comparison on first three methods.

===