Leveraging Parquet Files for Efficient Web Archive Collection Analytics

Sawood Alam, Internet Archive
Mark Phillips, University of North Texas
April 24, 2024

Collection Summarization Overview

What is this space?

What is a collection in this context?

What is a typical workflow for summarization?

- WARC -> CDX[J] -> Summary of CDX

Overview of common fields in CDX[J]

Why work with index instead of raw WARC?

- Size difference (storage)
- Computational complexity
- Ability to share indexes when content can’t be shared

Collection Summarization: What is this space?

We are often asked questions about our web archives that are best answered with aggregations of information.

Because of the nature of our web archives, working directly from WARC files poses challenges

- Size, Storage methods, compute resources needed

Even when we can work with WARCs we often can’t share the data directly

Researchers find working directly with WARC data challenging

Many useful questions about our web archives can be answered with our CDX indexes

Collection Summarization: What is a Collection?

As we talk more about collections within web archives today, keep in mind that it is a pretty loose term.

For researchers it might be a useful subset by date, filetype, domain name or top level domain.

Collections can be chunked into smaller collections depending on the summarization need.

Collection Summarization: Typical Workflow *(simplified)*

WARC files containing the archived resources have indexes created for each WARC

These indexes called CDX files and usually contain 9 or 11 fields

- **urlkey** (N): the URL of the captured web object, without the protocol (http://) or the leading www and in SURT format.
- **timestamp** (b): timestamp in the form YYYYMMDDhhmmss. The time represents the point at which the web object was captured, measured in GMT, as recorded in the CDX index file.
- **original** (a): the URL of the captured web object, including the protocol (http://) and the leading www, if applicable, extracted from the CDX index file.
- **mimetype** (m): the IANA media type as recorded in the CDX.
- **statuscode** (s): the HTTP response code received from the server at the time of capture, e.g., 200, 404.
- **digest** (k): a unique, cryptographic hash of the web object's payload at the time of the crawl. This provides a distinct fingerprint for the object; it is a Base32 encoded SHA-1 hash, derived from the CDX index file.
- **redirect** (r): likely blank or recorded with a "-"
- **metatags** (M): likely blank or recorded with a "-"
- **file_size** (S): the size of the web object, in bytes, derived from the CDX index file
- **offset** (V): the location of the resource in the compressed Web Archive (WARC) file which stores the full archived object
- **WARC** filename (g) - name of the compressed Web Archive (WARC) file which stores the full archived object

https://www.loc.gov/preservation/digital/formats/fdd/fdd000590.shtml

Collection Summarization: Typical Workflow *(simplified)*

CDXJ is used commonly for web archive indexes

https://nlevitt.github.io/warc-specifications/specifications/cdx-format/openwayback-cdxj/

This format provides for a way of storing additional arbitrary metadata in a JSON block for each record.

```
com,example)/ 20170730223850 {"url": "http://example.com/", "mime": "text/html", "status": "200", "digest": "G7HRM7BGOKSKMSXZAHMUQTTV53QOFSMK", "length": "1219", "offset": "771", "filename": "example-20170730223917.warc.gz"}
```

Temporal and Spatial Landscape of Archival Analysis

Long Duration

Single Webpage

Webpage Collection

- TMVis
- Wayback Machine Changes
- **TrendMachine**

- MementoMap
- CDX Summary
- Archives Unleashed Toolkit

- Memento Damage
- Archival ACID Test
- Reconstructive

- Warrick
- Wayback Machine Downloader
- Video Archiving Insights

Short Duration

Existing tools for Collection Analysis

Archives Unleashed Toolkit

https://archivesunleashed.org/aut/

Archives Research Compute Hub

https://github.com/internetarchive/arch

CDX Summary

https://github.com/internetarchive/cdx-summary

Summarize CDX

https://github.com/ymaurer/cdx-summarize

Common Crawl Tools/Workflow

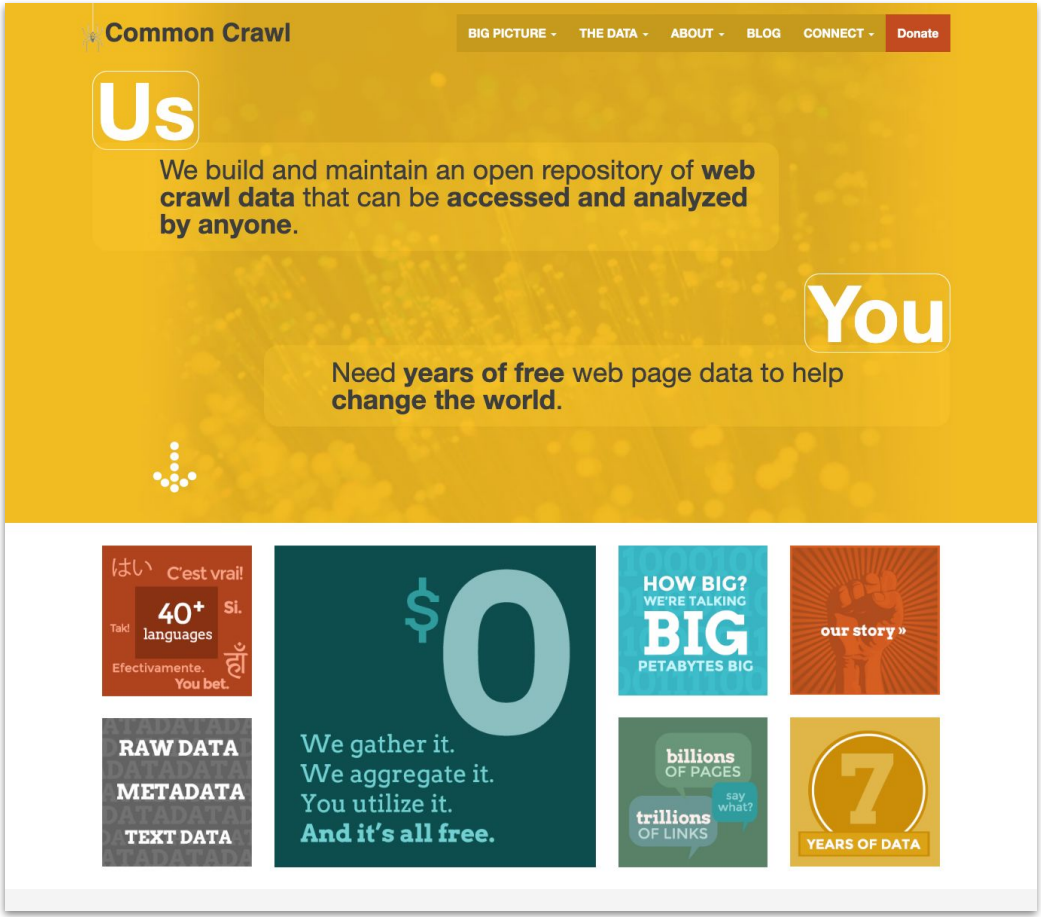
https://commoncrawl.org/blog/index-to-warc-files-and-urls-in-columnar-format

Common Crawl

https://commoncrawl.org

"Common Crawl is a 501(c)(3) non-profit organization dedicated to providing a copy of the internet to internet researchers, companies and individuals at no cost for the purpose of research and analysis."

- Monthly large (~300TB) crawls of the web
- Uses Nutch for crawling
- Stores data in WARC files
- Openly shares their data via AWS Open Data Sponsorship Program



Common Crawl Data

WARC files - content of crawls

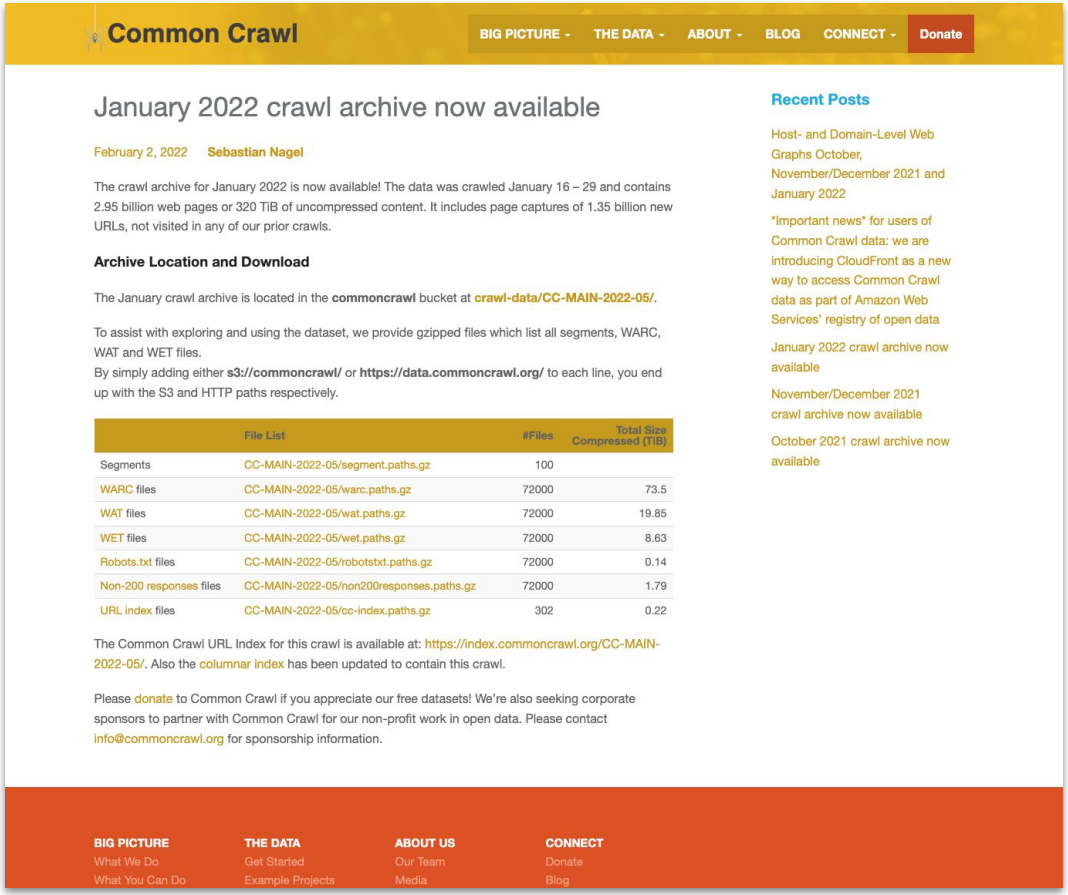
WAT - Extracted metadata from WARC files

WET - Extracted text from WARC files

(WAT and WET limited to HTML and TXT)

CDX Index - ZipNum format

Parquet Index - based on CDX Index



Common Crawl

BIG PICTURE · THE DATA · ABOUT · BLOG · CONNECT · Donate

January 2022 crawl archive now available

February 2, 2022 Sebastian Nagel

The crawl archive for January 2022 is now available! The data was crawled January 16 – 29 and contains 2.95 billion web pages or 320 TiB of uncompressed content. It includes page captures of 1.35 billion new URLs, not visited in any of our prior crawls.

Archive Location and Download

The January crawl archive is located in the **commoncrawl** bucket at **crawl-data/CC-MAIN-2022-05/**.

To assist with exploring and using the dataset, we provide gzipped files which list all segments, WARC, WAT and WET files.
By simply adding either **s3://commoncrawl/** or **https://data.commoncrawl.org/** to each line, you end up with the S3 and HTTP paths respectively.

	File List	#Files	Total Size Compressed (TiB)
Segments	CC-MAIN-2022-05/segment.paths.gz	100	
WARC files	CC-MAIN-2022-05/warc.paths.gz	72000	73.5
WAT files	CC-MAIN-2022-05/wat.paths.gz	72000	19.85
WET files	CC-MAIN-2022-05/wet.paths.gz	72000	8.63
Robots.txt files	CC-MAIN-2022-05/robotstxt.paths.gz	72000	0.14
Non-200 responses files	CC-MAIN-2022-05/non200responses.paths.gz	72000	1.79
URL index files	CC-MAIN-2022-05/cc-index.paths.gz	302	0.22

The Common Crawl URL Index for this crawl is available at: https://index.commoncrawl.org/CC-MAIN-2022-05/. Also the columnar index has been updated to contain this crawl.

Please donate to Common Crawl if you appreciate our free datasets! We're also seeking corporate sponsors to partner with Common Crawl for our non-profit work in open data. Please contact info@commoncrawl.org for sponsorship information.

Recent Posts

Host- and Domain-Level Web Graphs October, November/December 2021 and January 2022

Important news for users of Common Crawl data: we are introducing CloudFront as a new way to access Common Crawl data as part of Amazon Web Services' registry of open data

January 2022 crawl archive now available

November/December 2021 crawl archive now available

October 2021 crawl archive now available

BIG PICTURE: What We Do, What You Can Do
THE DATA: Get Started, Example Projects
ABOUT US: Our Team, Media
CONNECT: Donate, Blog

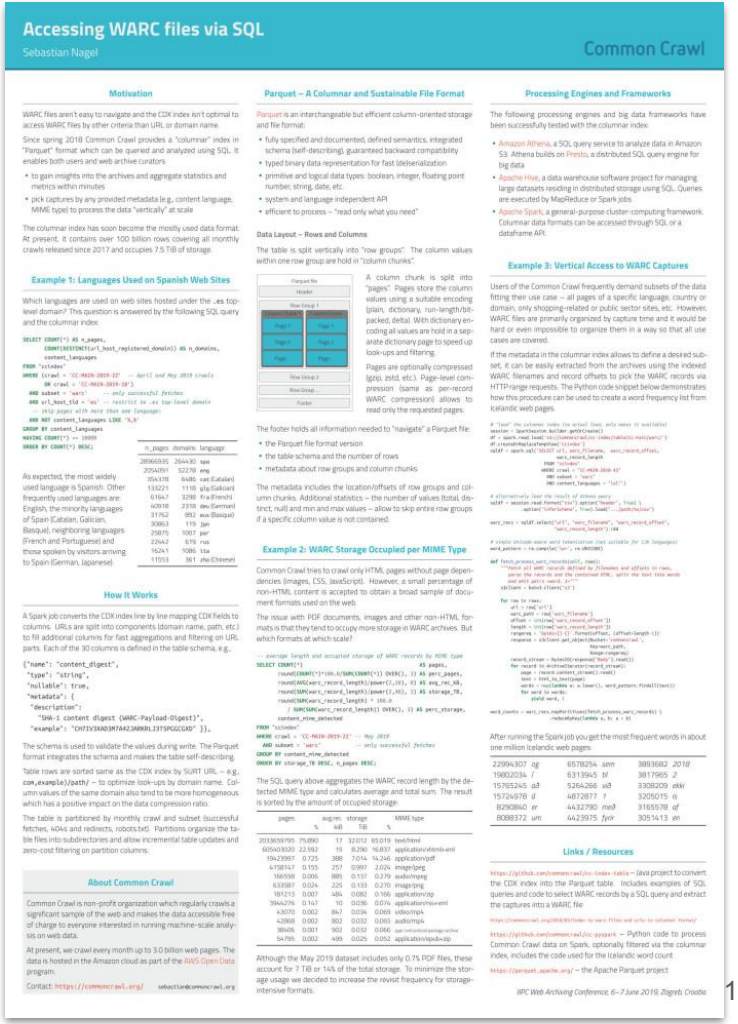
Accessing WARC files via SQL

Work presented by Sebastian Nagel of Common Crawl in Zagreb

Outlined tools and workflow for running queries via the “columnar” index to archives

Provides examples of queries to explore and answer questions directly from the Parquet index for Common Crawl datasets

https://digital.library.unt.edu/ark:/67531/metadc1608961/



Accessing WARC files via SQL

Sebastian Nagel

Common Crawl

Motivation

WARC files aren't easy to navigate and the CDX index isn't optimal to access WARC files by other criteria than URL or domain name.

Since spring 2018 Common Crawl provides a “columnar” index in “Parquet” format which can be queried and analyzed using SQL. It enables both users and web archive curators

- to gain insights into the archives and aggregate statistics and metrics within minutes
- pick captures by any provided metadata (e.g., content language, MIME type) to process the data “vertically” at scale

The columnar index has soon become the mostly used data format. At present, it contains over 100 billion rows covering all monthly crawls released since 2017 and occupies 7.5 TiB of storage.

Example 1: Languages Used on Spanish Web Sites

Which languages are used on web sites hosted under the .es top-level domain? This question is answered by the following SQL query and the columnar index:

```
SELECT COUNT(*) AS n_pages,
       COUNT(DISTINCT(url_host_registered_domain)) AS n_domains,
       content_languages
FROM "ccindex"
WHERE (crawl = 'CC-MAIN-2019-22' -- April and May 2019 crawls
       OR crawl = 'CC-MAIN-2019-18')
  AND subset = 'warc'      -- only successful fetches
  AND url_host_tld = 'es' -- restrict to .es top-level domain
  -- skip pages with more than one language:
  AND NOT content_languages LIKE '%,%'
GROUP BY content_languages
HAVING COUNT(*) >= 10000
ORDER BY COUNT(*) DESC;
```

n_pages	domains	language
28966935	264430	spa
2054091	52279	eng
354378	6486	cat (Catalan)
133221	1110	glg (Galician)
61647	3298	fra (French)
40918	2310	deu (German)
31762	852	eus (Basque)
30863	119	jpn
25875	1007	por
22642	878	rus
16241	1086	ita
11953	361	zho (Chinese)

As expected, the most widely used language is Spanish. Other frequently used languages are English, the minority languages of Spain (Catalan, Galician, Basque), neighboring languages (French and Portuguese) and those spoken by visitors arriving to Spain (German, Japanese).

How It Works

A Spark job converts the CDX index line by line mapping CDX fields to columns. URLs are split into components (domain name, path, etc.) to fill additional columns for fast aggregations and filtering on URL parts. Each of the 30 columns is defined in the table schema, e.g.,

```
{"name": "content_digest",
 "type": "string",
 "nullable": true,
 "metadata": {
   "description":
     "SHA-1 content digest (WARC-Payload-Digest)",
   "example": "VH7IV3KAOJM7A4ZJAMKBLI3TSPGGCGXD" }},
```

The schema is used to validate the values during write. The Parquet format integrates the schema and makes the table self-describing.

Table rows are sorted same as the CDX index by SURT URL – e.g., `com,example)/path/` – to optimize look-ups by domain name. Column values of the same domain also tend to be more homogeneous which has a positive impact on the data compression ratio.

The table is partitioned by monthly crawl and subset (successful fetches, 404s and redirects, robots.txt). Partitions organize the table files into subdirectories and allow incremental table updates and zero-cost filtering on partition columns.

About Common Crawl

Common Crawl is non-profit organization which regularly crawls a significant sample of the web and makes the data accessible free of charge to everyone interested in running machine-scale analysis on web data.

At present, we crawl every month up to 3.0 billion web pages. The data is hosted in the Amazon cloud as part of the AWS Open Data program.

Contact: https://commoncrawl.org/ sebastian@commoncrawl.org

Parquet – A Columnar and Sustainable File Format

Parquet is an interchangeable but efficient column-oriented storage and file format:

- fully specified and documented, defined semantics, integrated schema (self-describing), guaranteed backward compatibility
- typed binary data representation for fast (de)serialization
- primitive and logical data types: boolean, integer, floating point number, string, date, etc.
- system and language independent API
- efficient to process – “read only what you need”

Data Layout – Rows and Columns

The table is split vertically into “row groups”. The column values within one row group are hold in “column chunks”.

A column chunk is split into “pages”. Pages store the column values using a suitable encoding (plain, dictionary, run-length/bit-packed, delta). With dictionary encoding all values are hold in a separate dictionary page to speed up look-ups and filtering.

Pages are optionally compressed (gzip, zstd, etc.). Page-level compression (same as per-record WARC compression) allows to read only the requested pages.

The footer holds all information needed to “navigate” a Parquet file:

- the Parquet file format version
- the table schema and the number of rows
- metadata about row groups and column chunks

The metadata includes the location/offsets of row groups and column chunks. Additional statistics – the number of values (total, distinct, null) and min and max values – allow to skip entire row groups if a specific column value is not contained.

Example 2: WARC Storage Occupied per MIME Type

Common Crawl tries to crawl only HTML pages without page dependencies (images, CSS, JavaScript). However, a small percentage of non-HTML content is accepted to obtain a broad sample of document formats used on the web.

The issue with PDF documents, images and other non-HTML formats is that they tend to occupy more storage in WARC archives. But which formats at which scale?

```
-- average length and occupied storage of WARC records by MIME type
SELECT COUNT(*)                                                  AS pages,
       ROUND(COUNT(*)*100.0/SUM(COUNT(*)) OVER(), 3) AS perc_pages,
       ROUND(AVG(warc_record_length),power(2,20), 0) AS avg_rec_kB,
       ROUND(SUM(warc_record_length)/power(2,40), 3) AS storage_TB,
       ROUND(SUM(warc_record_length) * 100.0
         / SUM(SUM(warc_record_length)) OVER(), 3) AS perc_storage,
       content_mime_detected
FROM "ccindex"
WHERE crawl = 'CC-MAIN-2019-22' -- May 2019
  AND subset = 'warc'           -- only successful fetches
GROUP BY content_mime_detected
ORDER BY storage_TB DESC, n_pages DESC;
```

The SQL query above aggregates the WARC record length by the detected MIME type and calculates average and total sum. The result is sorted by the amount of occupied storage.

pages	%	avg rec size kB	storage TB	%	MIME type
2033659795	75.890	17	32.012	65.019	text/html
605403020	22.592	15	8.290	16.837	application/atom+xml
19423997	0.725	388	7.014	14.246	application/pdf
6758167	0.155	257	0.997	2.024	image/jpeg
166598	0.006	889	0.137	0.279	audio/mpeg
633587	0.024	225	0.133	0.270	image/png
181213	0.007	484	0.082	0.166	application/zip
3944276	0.147	10	0.036	0.074	application/rss+xml
43070	0.002	847	0.034	0.069	video/mp4
42868	0.002	802	0.032	0.065	audio/mp4
38406	0.001	902	0.032	0.066	application/vnd.google-earth.kmz
54795	0.002	499	0.025	0.052	application/vnd+zip

Although the May 2019 dataset includes only 0.7% PDF files, these account for 7 TiB or 14% of the total storage. To minimize the storage usage we decided to increase the revisit frequency for storage-intensive formats.

Processing Engines and Frameworks

The following processing engines and big data frameworks have been successfully tested with the columnar index:

- Amazon Athena, a SQL query service to analyze data in Amazon S3. Athena builds on Presto, a distributed SQL query engine for big data
- Apache Hive, a data warehouse software project for managing large datasets residing in distributed storage using SQL. Queries are executed by MapReduce or Spark jobs
- Apache Spark, a general-purpose cluster-computing framework. Columnar data formats can be accessed through SQL or a dataframe API.

Example 3: Vertical Access to WARC Captures

Users of the Common Crawl frequently demand subsets of the data fitting their use case – all pages of a specific language, country or domain, only shopping-related or public sector sites, etc. However, WARC files are primarily organized by capture time and it would be hard or even impossible to organize them in a way so that all use cases are covered.

If the metadata in the columnar index allows to define a desired subset, it can be easily extracted from the archives using the indexed WARC filenames and record offsets to pick the WARC records via HTTP range requests. The Python code snippet below demonstrates how this procedure can be used to create a word frequency list from Icelandic web pages.

```
# load the columnar index (no actual load, only makes it available)
session = SparkSession.builder.getOrCreate()
df = spark.read.load('s3a://commoncrawl/cc-index/table/cc-main/warc/')
df.createOrReplaceTempView('ccindex')
sql df = spark.sql('SELECT url, warc_filename, warc_record_offset,
                    warc_record_length
                     FROM "ccindex"
                     WHERE crawl = 'CC-MAIN-2018-43'
                       AND subset = 'warc'
                       AND content_languages = 'isl')

# alternatively load the result of Athena query:
sql df = session.read.format('csv').option('header', True).
                 option('inferSchema', True).load('...path/to/csv')

warc_recs = sql df.select('url', 'warc_filename', 'warc_record_offset',
                          'warc_record_length').rdd

# simple unicode word and tokenization (not suitable for LCR languages)
word_pattern = re.compile('\w+', re.UNICODE)

def fetch_process_warc_records(rows):
    '''Fetch all WARC records defined by filenames and offsets in rows,
       parse the records and the contained HTML, split the text into words
       and emit pairs (word, 1)'''
    s3client = boto3.client('s3')

    for row in rows:
        url = row['url']
        warc_filename = row['warc_filename']
        offset = int(row['warc_record_offset'])
        length = int(row['warc_record_length'])
        rangeheq = 'bytes=%d-%d'%(offset,offset+length-1)
        response = s3client.get_object(Bucket='commoncrawl',
                                       Key=warc_path,
                                       Range=rangeheq)
        record_stream = BytesIO(response["Body"].read())
        for record in ArchiveIterator(record_stream):
            page = record.content_stream().read()
            text = html_to_text(page)
            words = map(lambda w: w.lower(), word_pattern.findall(text))
            for word in words:
                yield word, 1

word_counts = warc_recs.mapPartitions(fetch_process_warc_records) \
                       .reduceByKey(lambda a, b: a + b)
```

After running the Spark job you get the most frequent words in about one million Icelandic web pages

22994307	og	6578254	sem	3893682	2018
19802034	í	6313945	til	3817965	2
15765245	að	5264266	við	3308209	ekki
15724978	á	4872877	í	3205075	is
8290860	er	4432790	með	3165578	af
8088372	um	4423975	fyrir	3051413	en

Links / Resources

https://github.com/commoncrawl/cc-index-table – Java project to convert the CDX index into the Parquet table. Includes examples of SQL queries and code to select WARC records by a SQL query and extract the captures into a WARC file

https://commoncrawl.org/2018/03/index-to-warc-files-and-urls-in-columnar-format/

https://github.com/commoncrawl/cc-pyspark – Python code to process Common Crawl data on Spark, optionally filtered via the columnar index, includes the code used for the Icelandic word count

https://parquet.apache.org/ – the Apache Parquet project

IIPC Web Archiving Conference, 6–7 June 2018, Zagreb, Croatia

End of Term Web Archive

End of Term Web Archive

- Collaborative web archiving activity in the United States since 2008
- Goal to document the transition in the Executive Branch of the Federal web before and after each election cycle
- Serves as a longitudinal snapshot of Federal .gov and public .mil web every four years
- Partners volunteer time, crawling, and storage resources for the project
- Public access provided by the Internet Archives' Wayback Machine
- https://eotarchive.org

EOT Crawling Partners

	2004*	2008	2012	2016	2020
Archive Team (AT)				Crawl	
California Digital Library (CDL)		Crawl			
Internet Archive (IA)		Crawl	Crawl	Crawl	Crawl
Library of Congress (LOC)		Crawl	Crawl	Crawl	
National Archives and Records Administration (NARA)	Crawl				
University of North Texas (UNT)		Crawl	Crawl	Crawl	Crawl

* Technically pre-EOT

Datasets to date

Crawl	WARC Files	WARC Size	WAT Size	WET Size	CDX Size	META Size
EOT-2004	58,977	7TB	108GB	18MB	6GB	36GB
EOT-2008	125,704	15TB	447GB	108GB	9GB	68GB
EOT-2012	78,509	41TB	885GB	217GB	12GB	82GB
EOT-2016	194,683	139TB	2TB	331GB	25GB	178GB
EOT-2020	239,811	266TB	9TB	3TB	84GB	713GB
Total	638,707	468TB	12TB	4TB	136GB	1TB

Where to get the datasets

https://eotarchive.org/data/

End of Term Web Archive

Background Partners Datasets

Datasets

End of Term Datasets

The End of Term project is working with the Amazon Web Services' Open Data Sponsorship Program to host a copy of the 2004, 2008, 2012, 2016, and 2020 End of Term Datasets.

The work of inventorying, staging and moving the data into AWS is still ongoing and more information will be provided here in the future.

Currently we have these datasets partially available for use.

Dataset	WARC #	WARC Size Compressed
EOT-2020	239811	266.04 TB
EOT-2016	194683	139.3 TB
EOT-2012	78509	41.42 TB
EOT-2008	125704	15.32 TB
EOT-2004	58977	6.42 TB

Dataset Overview

Download with HTTP or S3

Path files contain full paths to each file in dataset.

Download path files and then iterate over all lines in file to retrieve full dataset

Take the parts you need

If you have questions reach out.

mark.phillips@unt.edu

sawood@archive.org

End of Term 2020 Dataset

End of Term 2020 Dataset

The End of Term 2020 Dataset represents data collected by two collecting institutions. These institutions were the Internet Archive (IA) and the University of North Texas Libraries (UNT). The data is part of the initiative called the End of Term Presidential Web Archive.

Archive Location and Download

The 2020 End of Term archive is located on the **eotarchive** bucket at EOT-2020.

To assist with exploring and using the dataset, we provide gzipped files which list all segments, WARC, WAT, WET, and CDX files.

By adding either **s3://eotarchive/** or **https://eotarchive.s3.amazonaws.com/** to each line, you end up with the s3 and HTTP paths respectively.

File	List	#Files	Total Size Compressed
Segments	EOT-2020/segment.paths.gz	26	
WARC files	EOT-2020/warc.paths.gz	239811	266.04 TB
WAT files	EOT-2020/wat.paths.gz	239811	9.15 TB
WET files	EOT-2020/wet.paths.gz	239811	2.6 TB
META files	EOT-2020/meta.paths.gz	239811	712.66 GB
CDX files	EOT-2020/cdx.paths.gz	239811	83.66 GB
URL Index files	EOT-2020/eot-index.paths.gz	49	74.4 GB

Tools Used

Small 5-node Local Hadoop Cluster (250TB) & mrjob

WAT/WET

https://github.com/commoncrawl/ia-web-commons

https://github.com/commoncrawl/ia-hadoop-tools

CDXJ

https://github.com/webrecorder/cdxj-indexer

WARC Metadata Sidecar

https://github.com/unt-libraries/warc-metadata-sidecar

Zipnum

https://github.com/commoncrawl/webarchive-indexing

Parquet

https://github.com/commoncrawl/cc-index-table

```
D DESCRIBE SELECT * FROM read_parquet('*.parquet');
```

column_name varchar	column_type varchar	null varchar	key varchar	default varchar	extra varchar
url_surtkey	VARCHAR	YES			
url	VARCHAR	YES			
url_host_name	VARCHAR	YES			
url_host_tld	VARCHAR	YES			
url_host_2nd_last_part	VARCHAR	YES			
url_host_3rd_last_part	VARCHAR	YES			
url_host_4th_last_part	VARCHAR	YES			
url_host_5th_last_part	VARCHAR	YES			
url_host_registry_suffix	VARCHAR	YES			
url_host_registered_domain	VARCHAR	YES			
url_host_private_suffix	VARCHAR	YES			
url_host_private_domain	VARCHAR	YES			
url_host_name_reversed	VARCHAR	YES			
url_protocol	VARCHAR	YES			
url_port	INTEGER	YES			
url_path	VARCHAR	YES			
url_query	VARCHAR	YES			
fetch_time	TIMESTAMP	YES			
fetch_status	SMALLINT	YES			
content_digest	VARCHAR	YES			
content_mime_type	VARCHAR	YES			
content_mime_detected	VARCHAR	YES			
content_charset	VARCHAR	YES			
content_languages	VARCHAR	YES			
content_puid	VARCHAR	YES			
warc_filename	VARCHAR	YES			
warc_record_offset	BIGINT	YES			
warc_record_length	BIGINT	YES			
warc_segment	VARCHAR	YES			
crawl	VARCHAR	YES			
subset	VARCHAR	YES			

31 rows 6 columns

D

Adding new fields to CDXJ

WARC Metadata Sidecars

Python tool for content-based characterization of WARC files.

https://github.com/unt-libraries/warc-metadata-sidecar

Language identification - Compact Language Detector 2 (CLD2)

Format Identification - Fido & python-magic

Encoding Identification - chardet

Soft404 detection - soft-404

Writes output to WARC Metadata record.

Language Identification

- Compact Language Detector 2 (CLD2)
- Python bindings for CLD2 using - pycld2
- Language identification is performed on html and txt files
- Returns up to three languages present in each file
- Information from CLD2 includes
  - reliability
  - language name
  - two digit language code
  - text-coverage
  - score
- Future implementations we will look at pycld3 which is neural network model for language identification
- https://github.com/aboSamoor/pycld2

Format Identification

- Format Identification is performed on all content payloads in WARC
- Fido used for MIME type and preservation identifiers
  - Uses the PRONOM format registry
  - Returns PRONOM identifiers
  - Example: Preservation-Identifier: fmt/99
- python-magic used for MIME type identification
  - Python interface to the libmagic file type identification library.
  - Similar to the Unix command `file`
  - Example: Identified-Payload-Type: {"fido": "text/html", "python-magic": "text/html"}
- Output of both Fido and python-magic are stored in metadata sidecar
  - Fido is generally more specific about formats (xhtml vs. html)
  - python-magic has a more general output
- Fido - https://github.com/openpreserve/fido
- python-magic - https://github.com/ahupp/python-magic

Encoding Detection

- chardet - Universal Character Encoding Detector
- Encoding detection is performed on html and txt files
- Python implementation
- Port of the auto-detection code in Mozilla
  - Example: Charset-Detected: {"encoding": "ascii", "confidence": 1.0}
- https://github.com/chardet/chardet

Soft404 Detection

- Soft 404 detection for HTML pages
- A “soft” 404 page is a page that is served with 200 status, but is really a page that says that content is not available.
- Model trained on over 117,000 pages of content from a wide set of languages
- Returns probability of HTML being a Soft 404
  - Example: Soft-404-Detected: 0.022243212227210058
- https://github.com/TeamHG-Memex/soft404

Example WARC Metadata Sidecar Record

```
WARC/1.0
WARC-Date: 2012-10-26T21:59:42Z
WARC-Concurrent-ID: <urn:uuid:da5927b0-4efb-469b-a885-04c7347a0dc6>
WARC-Type: metadata
WARC-Record-ID: <urn:uuid:757626b2-c1fd-4430-abd0-545cdee7cefc>
WARC-Target-URI: http://140.194.76.129/publications/eng-pamphlets/index.html
WARC-Payload-Digest: sha1:SUYN7XTFAOFZ7RB6PQSUDVXZV6DPKAS3
WARC-Block-Digest: sha1:SUYN7XTFAOFZ7RB6PQSUDVXZV6DPKAS3
Content-Type: application/warc-fields
Content-Length: 308

Identified-Payload-Type: {"fido": "text/html", "python-magic": "text/html"}
Preservation-Identifier: fmt/99
Charset-Detected: {"encoding": "ascii", "confidence": 1.0}
Languages-cld2: {"reliable": true, "text-bytes": 16199, "languages": [{"name": "ENGLISH", "code": "en",
"text-covered": 99, "score": 878.0}]}
```

Integrating WARC Metadata Sidecar Files

- sidecar2cdxj.py
  - Creates cdxj output for each WARC Metadata Record in sidecar.
  - Contains all information from content-based identification
- merge_cdxj.py
  - Helps to merge cdxj from primary WARC file and cdxj from WARC Metadata Sidecar
  - Implements project-specific logic on what fields to combine.
    - Example: python-magic MIME over Fido MIME
    - Example: Limit to one, two, or three language codes
  - Uses a combination of URL and timestamp to combine records
    - Future improvement could be to use UUIDs contained WARC records

Sidecar and WARC CDXJ files


```
129,76,194,140)/publications/eng-pamphlets/index.html 20121026215942
{
	"Identified-Payload-Type": {
		"fido": "text/html",
		"python-magic": "text/html"
	},
	"Preservation-Identifier": "fmt/99",
	"Charset-Detected": {
		"encoding": "ascii",
		"confidence": 1.0
	},
	"Languages-cld2": {
		"reliable": true,
		"text-bytes": 16199,
		"languages": [{
			"name": "ENGLISH",
			"code": "en",
			"text-covered": 99,
			"score": 878.0
		}]
	}
}
```


Metadata Sidecar CDXJ


```
129,76,194,140)/publications/eng-pamphlets/index.html 20121026215942
{
	"url":
"http://140.194.76.129/publications/eng-pamphlets/index.html",
	"mime": "text/html",
	"status": "200",
	"digest": "VJQMCBDSG6QIN4LBRRQZH2JWRVJ5JYCF",
	"length": "12637",
	"offset": "587",
	"filename":
"crawl-data/EOT-2012/segments/UNT-001/warc/UNT-20121026215944486-00552-5180~libharvest1.library.unt.edu~8443.warc.gz"
}
```


WARC CDXJ

Merged CDXJ with Sidecar Metadata

129,76,194,140)/publications/eng-pamphlets/index.html 20121026215942
{
"**url**": "http://140.194.76.129/publications/eng-pamphlets/index.html",
"**mime**": "text/html",
"**status**": "200",
"**digest**": "VJQMCBDSG6QIN4LBRRQZH2JWRVJ5JYCF",
"**length**": "12637",
"**offset**": "587",
"**filename**":
"crawl-data/EOT-2012/segments/UNT-001/warc/UNT-20121026215944486-00552-5180~libharvest1.library.unt.edu~8443.warc.gz",
"**mime-detected**": "text/html",
"**puid**": "fmt/99",
"**charset**": "ascii",
"**languages**": "eng"
}

EOT 2012 National Laboratories Collection

Dataset Specifics

Extracted from the 2012 End of Term Crawls

Basic extraction of CDXJ records from a National Laboratory domain

Contains the Energy Department's 17 National Labs

Some of their domains have changed from 2012 to 2024

Total file size of compressed (gzipped) CDXJ file is 216MB

End of Term National Labs Dataset - 2012

edu.stanford.slac	SLAC National Accelerator Laboratory
gov.ameslab	Ames Laboratory
gov.anl	Argonne National Laboratory
gov.bnl	Brookhaven National Laboratory
gov.doe.netl	National Energy Technology Laboratory
gov.doe.srnl	Savannah River National Laboratory
gov.fnal	Fermi National Accelerator Laboratory
gov.inl	Idaho National Laboratory
gov.lanl	Los Alamos National Laboratory
gov.lbl	Lawrence Berkeley National Laboratory
gov.llnl	Lawrence Livermore National Laboratory
gov.nrel	National Renewable Energy Laboratory
gov.ornl	Oak Ridge National Laboratory
gov.pnnl	Pacific Northwest National Laboratory
gov.pppl	Princeton Plasma Physics Laboratory
gov.sandia	Sandia National Laboratory
org.jlab	Thomas Jefferson National Accelerator Facility

CDX Summary Overview of EOT National Lab Dataset

https://github.com/internetarchive/cdx-summary

CDX Overview

Total Captures in CDX	3,924,946
Consecutive Unique URLs	3,410,554
Consecutive Unique Hosts	2,004
Total WARC Records Size	1.1 TB
First Memento Date	Sep 13 2012
Last Memento Date	Mar 31 2013

MIME Type and Status Code Distribution

MIME	2XX	3XX	4XX	5XX	Other	TOTAL
HTML	1,868,220	124,206	199,751	9,226	0	2,201,403
Image	1,061,600	404	1	0	0	1,062,005
CSS	24,916	2	0	1	0	24,919
JavaScript	7,754	1	0	0	0	7,755
JSON	474	0	6	0	0	480
XML	41,122	45	78	153	0	41,398
Text	137,084	30,589	1,776	87	0	169,536
PDF	209,684	0	0	2	0	209,686
Audio	989	0	0	0	0	989
Video	5,023	0	0	0	0	5,023
Revisit	0	0	0	0	43,758	43,758
Other	145,862	5,790	6,339	3	0	157,994
TOTAL	3,502,728	161,037	207,951	9,472	43,758	3,924,946

Path and Query Segments

Path	Q0	Q1	Q2	Q3	Q4	Other	TOTAL
P0	2,001	4,255	3,298	222	1,072	17	10,865
P1	57,439	20,344	12,431	25,655	11,628	36,251	163,748
P2	249,205	164,127	120,002	193,958	52,590	82,749	862,631
P3	478,970	125,016	46,142	36,594	29,150	24,499	740,371
P4	630,813	55,958	33,164	11,620	14,203	13,801	759,559
Other	1,258,435	51,629	30,233	11,495	14,140	21,840	1,387,772
TOTAL	2,676,863	421,329	245,270	279,544	122,783	179,157	3,924,946

Year and Month Distribution

Year	01	02	03	04	05	06	07	08	09	10	11	12	TOTAL
2012	0	0	0	0	0	0	0	0	433,574	239,888	1,106	1,985	676,553
2013	1,781	1,555,616	1,690,996	0	0	0	0	0	0	0	0	0	3,248,393
TOTAL	1,781	1,555,616	1,690,996	0	0	0	0	0	433,574	239,888	1,106	1,985	3,924,946

Top 10 Out of 2,004 Hosts

Host	Captures
usaxs.xray.aps.anl.gov	178,298
nrel.gov	81,544
sandia.gov	80,985
fnal.gov	79,431
ornl.gov	71,190
inlportal.inl.gov	70,136
bnl.gov	62,693
lanl.gov	50,292
jlab.org	47,624
llnl.gov	46,094
OTHERS (1,994 Hosts)	3,156,659

Why are we talking about this and Parquet?

CDX[J] files are column-based format of select metadata fields from WARC files.

Often scripts and tools iterate over these formats row by row and aggregate output.

Answering similar questions often requires multiple times through the dataset to answer

This doesn’t matter as much with small data but as the datasets grow it becomes problematic

CDX is optimized for archival playback, Parquet can be optimized for analytics

Enter: Columnar (Column-Oriented) storage formats (Parquet/Other Examples)

Tease at Parquet - Why this is cool/useful

Being able to parse data once into a stable format that we can run multiple queries against.

Move web archiving data/formats into standard tools/workflows

Query web archives (cdx data specifically) using SQL or using DataFrames

Quick SQL example to show

Converting CDXJ to Parquet

Start with sorted CDXJ Index

Process URL to create meaningful subfields

Make use of WARC Metadata Sidecar fields

Currently based on Common Crawl's cc-index-table

Custon EOT table for additional rows

https://github.com/commoncrawl/cc-index-table

```
D DESCRIBE SELECT * from EOTNL.parquet;
```

column_name	column_type	null	key	default	extra
url_surtkey	VARCHAR	YES			
url	VARCHAR	YES			
url_host_name	VARCHAR	YES			
url_host_tld	VARCHAR	YES			
url_host_2nd_last_part	VARCHAR	YES			
url_host_3rd_last_part	VARCHAR	YES			
url_host_4th_last_part	VARCHAR	YES			
url_host_5th_last_part	VARCHAR	YES			
url_host_registry_suffix	VARCHAR	YES			
url_host_registered_domain	VARCHAR	YES			
url_host_private_suffix	VARCHAR	YES			
url_host_private_domain	VARCHAR	YES			
url_host_name_reversed	VARCHAR	YES			
url_protocol	VARCHAR	YES			
url_port	INTEGER	YES			
url_path	VARCHAR	YES			
url_query	VARCHAR	YES			
fetch_time	TIMESTAMP	YES			
fetch_status	SMALLINT	YES			
content_digest	VARCHAR	YES			
content_mime_type	VARCHAR	YES			
content_mime_detected	VARCHAR	YES			
content_charset	VARCHAR	YES			
content_languages	VARCHAR	YES			
content_puid	VARCHAR	YES			
warc_filename	VARCHAR	YES			
warc_record_offset	BIGINT	YES			
warc_record_length	BIGINT	YES			
warc_segment	VARCHAR	YES			
crawl	VARCHAR	YES			
subset	VARCHAR	YES			

```
          url_surtkey = gov,anl,alcf,esp)/blog
                  url = http://www.esp.alcf.anl.gov/blog/
            url_host_name = www.esp.alcf.anl.gov
             url_host_tld = gov
    url_host_2nd_last_part = anl
    url_host_3rd_last_part = alcf
    url_host_4th_last_part = esp
    url_host_5th_last_part = www
  url_host_registry_suffix = gov
url_host_registered_domain = anl.gov
   url_host_private_suffix = gov
   url_host_private_domain = anl.gov
    url_host_name_reversed = gov.anl.alcf.esp.www
             url_protocol = http
                 url_port =
                 url_path = /blog/
                url_query =
               fetch_time = 2013-02-25 01:29:36
             fetch_status = 200
           content_digest = 5JDQ4KZFAGOXSVHBVHDLV77VFOBVJ5QS
        content_mime_type = text/html
     content_mime_detected = text/html
          content_charset = utf-8
        content_languages = eng,ile
             content_puid = fmt/96
            warc_filename = crawl-data/EOT-2012/segments/IA-001/warc/EOT-2012-20130225012855292-05948-10476~wbgrp-crawl013.us.archive.org~8443.warc.gz
       warc_record_offset = 39276133
       warc_record_length = 12724
             warc_segment = IA-001
                    crawl = EOT-2012
                   subset = warc
```

All well and good, but what is a Parquet file?