Materials you are responsible for

- All lecture material
- Assignment 1
- Readings, papers
- Topics discussed during class
- Videos

Exam

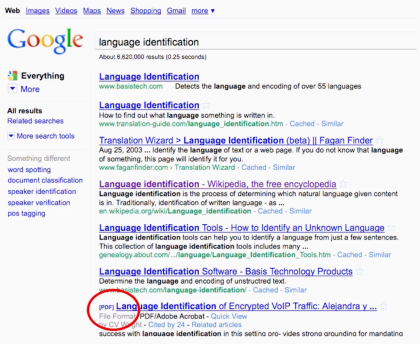
- Closed book, closed note
- Format
  - Write in answers
  - **No** multiple choice
- 10 questions across 4 pages with 1 extra sheet
- Will be administered via D2L
- You sign into WebEx camera on for first 80m and take exam. Regular class after.

Proliferation of content types available

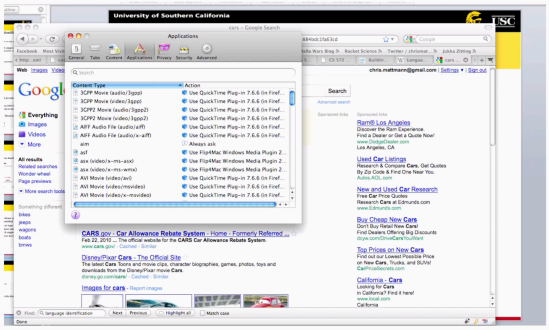
- By some accounts, 16K to 51K content types*
- What to do with content types?
  - Parse them
    - How?
    - Extract their text and structure
  - Index their metadata
    - In an indexing technology like Lucene, Solr, or Compass, or in Google Appliance
  - Identify what language they belong to
    - Ngrams

*http://filext.com/

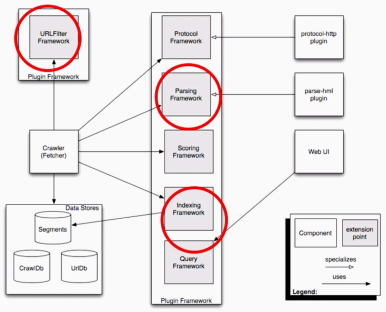
Importance of content types



Importance of content type detection

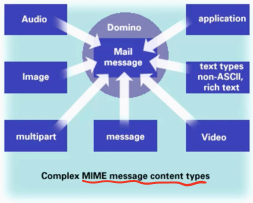


Search Engine Architecture



Goals

- Identify and classify file types
  - MIME detection
    - Glob pattern
      - *.txt
      - *.pdf
    - URL
      - http://…pdf
      - ftp://myfile.t
    - Magic bytes
    - Combination the above me



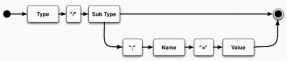
- Classification means reaction can be targeted so you can figure how to open this thing

MIME Types Defined

- RFC 822 https://www.ietf.org/rfc/rfc0822.txt
  - Standard for the Format of ARPA Internet Text Messages - 1982
  - Envelope (Headers incl Email/Date/Time) + Contents
- RFC 2045 https://www.ietf.org/rfc/rfc2045.txt
  - 822 inadequate for including other file types in Email
  - Adds MIME-Version, Content-Type, Content-Transfer-Encoding and e.g., Content-ID, Content-Description
  - Included Charset definition (non US-ASCII)
  - Initial definition of MIME taxonomy

MIME Types Defined (part 2)

- RFC 2045
  - Content-Type:
    - Type / subtype
    - Types: message, multi-part, text, image, audio, video, application, extension-token
    - Parameters
      - Attribute=value
      - E.g., encoding=UTF-8
      - E.g., charset=us-ascii



- Parameters are case-insensitive
- 7 Initial Types defined, with mechanism to add new types defined by IETF

MIME Types Defined (part 3)

- RFC 2046 http://www.hjp.at/doc/rfc/rfc2046.html
  - Augments 2045 with definitions of MIME taxonomy types and initial sub-types
  - Plain, unrecognized sub-types defined
  - Octet-Stream sub-type of application
  - Discrete Types (atomic)
    - Application, image, video, audio, text
  - Composite Types
    - Message, multi-part
  - Extension Types
    - Formal way of adding new x-types, without requiring IANA

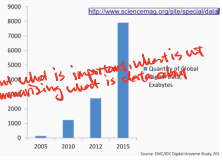
Big Data Definitions - NASA

Big Data: Big Data describes the situation in which collection, storage and analysis of data exceeds the capability and capacity of conventional methods or software systems. This state necessitates new architectural approaches in data management, artificial intelligence, statistics and visualization that change the paradigm by which data is collected, stored, processed, transmitted and analyzed, often through scalable cyberinfrastructures and algorithms.

- NASA OCT

U.S. National Science Foundation

- The Big Picture
  - Astronomy, Earth science, planetary science, life/physical science all *drowning in data*
  - Fundamental technologies and emerging techniques in archiving and data science
    - Largely center around *open source* communities and related systems
- Research challenges (adapted from NSF)
  - *More data is being collected than we can store*
  - *Many data sets are too large to download*
  - *Many data sets are too poorly organized to be useful*
  - *Many data sets are heterogeneous in type, structure*
  - *Data utility is limited by our ability to use it*
- Big Data Content Detection and Analysis
  - *Research methods for integrating intelligent algorithms for data triage, subsetting, summarization*
  - *Construct technologies for smart data movement*
  - *Evaluate cloud computing for storage/processing*
  - *Construct data/metadata translators "Babel Fish"*



National Institutes of Standards & Technology (NIST) – Big Data WG

- http://bigdatawg.nist.gov/home.php
- Big Data refers to the inability of traditional data architectures to efficiently handle the new datasets. Characteristics of Big Data that force new architectures are:
  - Volume (i.e., the size of the dataset);
  - Variety (i.e., data from multiple repositories, domains, or types);
  - Velocity (i.e., rate of flow); and
  - Variability (i.e., the change in other characteristics).

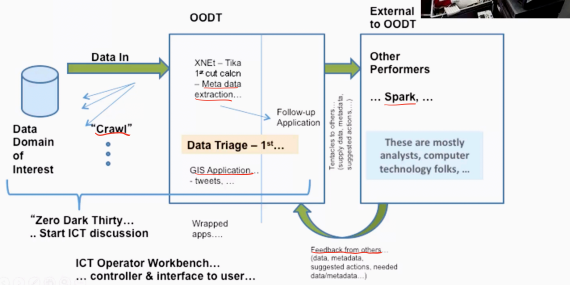
More data collected than we can store

- Data is collected faster than we can store it
- Rationale
  - Instrument resolutions – cameras take bigger and better pictures nowadays (10+ megapixels on your phone, compared to in specialized instruments)
  - Data taken everywhere – "everyone is a sensor"
    - Cameras driving this
- Some examples

Many datasets are too poorly organized to be useful

- Data *Forensics*
- Sometimes for scientific analysis
  - NASA Lunar Recovery Project
    - http://www.nasa.gov/topics/moonmars/features/LOIRP/
  - International Environmental Data Rescue Organization (IEDRO)http://www.theatlantic.com/technology/archive/2014/08/the-quest-to-scan-millions-of-weather-records/378962/
- Sometimes for defense and military
  - Seal Team 6 – cache of disk and data bricks/forensics

"Zero Dark Thirty"



Many datasets are heterogeneous in size and structure

- Images
- Videos
- Audio
- PDF, documents
- Science data
- Code

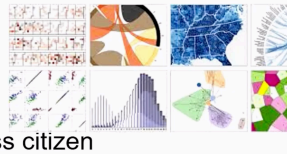
Similarly, you know.

Heterogeneous data sets

- File oriented
- Streaming data
- Databases
- NoSQL
- We will come back to this!

Data utility is limited by our ability to use the data

- Ability to search and interact with information
- Self describing data
  - Linked open data, RDF, Microformats, etc.
- *Metadata* as a first class citizen
  - *Data about data*
- Understanding the right ways to present information



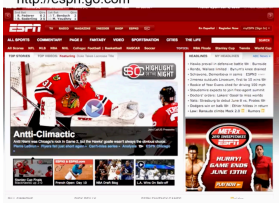
The Five V's

- Volume – more data collected than we can store
  - Total Volume at inception; at end of the project
  - Implications on Storage, on Services, on Architectures
- Velocity – too large to download and more than we can store
  - How fast is it coming
  - Matters – I/O – how fast the disk can write the data
  - Triaging the data
- Variety – too unorganized to be useful – heterogeneous in size/structure
  - How many data and metadata types – how are they described?
- Veracity – limited by our ability to use the data
  - Confidence in the data – how big is the error bar? Is this a guess?
- Value – limited by our ability to use the data
  - What decisions can be made on this data, and how easily is it to turn this data into knowledge and value?

What are Web Duplicates?

- The same page, referenced by different URLs

http://espn.go.com



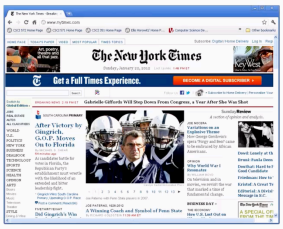
http://www.espn.com

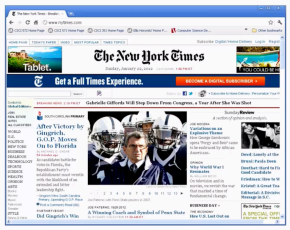
•What are the differences?
•URL host (virtual hosts), sometimes protocol, sometimes page name, etc.

Duplicate/Near-Duplicate Detection

- There are two forms of page duplication
  - *Duplicate*: Exact match;
    - Solution: compute fingerprints or use cryptographic hashing
    - SHA-1 and MD5 are the two most popular cryptographic hashing methods
  - *Near-Duplicate*: Approximate match
    - Solution: compute the syntactic similarity with an edit-distance measure, and
    - Use a similarity threshold to detect near-duplicates
      - e.g., Similarity > 80% => Documents are "near duplicates"

Slightly different results





Two copies of www.nytimes.com snapshot within a few seconds of each other
The pages are essentially identical except for the ads;

Why is it important to detect web mirrors in dedup

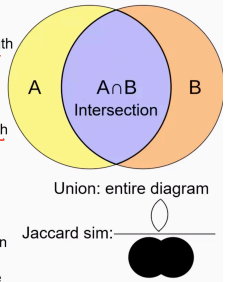
- Smart crawling
  - Avoid duplication
  - Allow fetching from the fastest or freshest server
- Better connectivity analysis
  - Combine in-links from the multiple mirror sites to get an accurate PageRank
  - Avoid double counting out-links
- Add redundancy in result listings
  - "If that fails you can try: <mirror>/samepath"

General paradigm for detecting web duplicates

1. Define a function f that captures the contents of each document in a number
   - E.g. hash function, signature, fingerprint
2. Create the pair <f(doc$_i$), ID of doc$_i$> for all doc$_i$
3. Sort the pairs
4. Documents that have the same *f* value or an *f* value within a small threshold are believed to be duplicates

Jaccard Similarity

- The Jaccard similarity of sets A and B, J(A,B), is
- | A intersect B | / |A union B |
- A *k-shingle* of a document is any substring of length *k* found in the document
  - We ignore multiple blanks, tabs, newlines
- Choosing an appropriate value for *k*
  - If *k* is too small, then most sequences of length *k* will occur in most documents
  - *k* should be picked large enough that the probability of any given shingle appearing in any document is low
  - E.g. suppose in email only letters and white space occur (this is a simplified example); then there are 27^5 = 14,348,907 possible 5-shingles; since emails are generally small, the value k=5 should work well



General properties of distance measures

- Distance measure must satisfy 4 properties
  1. No negative distances
  2. D(x,y) = 0 iff x=y
  3. D(x,y) = d(y,x) symmetric
  4. D(x,y) <= d(x,z) + d(z,y) triangle inequality

There are several distance measures that can play a role in locating duplicate and near-duplicate documents

- *Euclidean distance* – d([x_1…x_n], [y_1,…,y_n]) = sqrt(Sum(x_i-y_i)^2) i=1…n
- *Jaccard distance* – d(x,y) = 1 – SIM(x,y) or 1 minus the ratio of the sizes of the intersection and union of sets x and y
- *Cosine distance* – the cosine distance between two points (two *n* element vectors) is the angle that the vectors to those points make; in the range 0 to 180 degrees
- *Edit distance* – the distance between two strings is the smallest number of insertions and deletions of single characters that will convert one string into the other
- *Hamming distance* – between two vectors is the number of components in which they differ (usually used on boolean vectors)

Why automated file detection

- Virus Scanning
  - Many virus scanning software tools rely on the ability to identify which files may contain viruses in them, and so the ability to automatically detect executable files is key
- Firewalls / Networking / Intrusion Detection Systems
  - At the network level, allowing e.g., SMTP communication on port 25 with an EXE attachment or Python file is important to detect
- File Forensics
  - When handed a bunch of disks with data on them, being ability accurately classify that data for legal purposes, etc., is key
- Scientific Data / Processing / Big Data
  - Sorting level 1 data (raw telemetry) from process geo-referenced science data (level 2)

File Formats

- "A file format is a standard way that information is encoded for storage in a computer file. It specifies how bits are used to encode information in a digital storage medium. File formats may be either proprietary or free and may be either unpublished or open."
- https://en.wikipedia.org/wiki/File_format

Basic approaches

- 1. File Extension based – can easily be spoofed. Microsoft's Operating Systems typically have used this approach almost exclusively. OS maintains table of associations and must be manually updated by user.
- 2. /etc/magic file – Special file in UNIX oriented systems that look at first 16 bits of each file and then associates these as a magic number with each file. Table is maintained in /etc/magic file
- 3. Container File Formats – encapsulate other files

Approaches for automatic detection

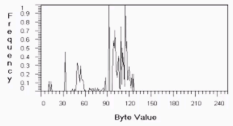
- Basic (Content-Independent)
  - File extension
  - File Directory Structure
  - File Naming Convention
- Content-Based Algorithms
  - Byte Frequency Analysis
  - Byte Frequency Correlation Analysis
  - File Header Trailer Analysis
  - /etc/magic

Learning from the actual File Data

- Previous approaches were somewhat limited
  - Rely on filenames which are easily spoofed and/or changed
  - Rely on tables that are difficult to discern and must be maintained and updated (Magic)
  - Rely on new proprietary "wrapper formats" (Containers)
- Key Insight: What if we simply examined the data inside many existing files and tried to develop a "fingerprint" a la Similarity metrics to identify files => Content-based approaches

Content-Based File Type Detection

- McDaniel & Heydari, 2003
  - Investigated three approaches
  - Byte Frequency Analysis
  - Byte Frequency Correlation
  - File Header Trailer (FHT)

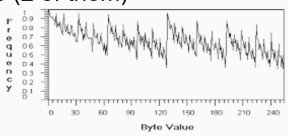


- Some key targets
  - Accuracy – approach should be highly accurate (overcome filenames) and difficult to spoof (security/firewalls/viruses)
  - Automatic – lots of files, big data, need to run fast
    - Fingerprints should be small and easily comparable
  - Flexibility – don't want to have to maintain tables

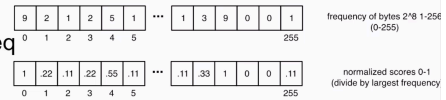
Byte Frequency Distribution (BFD)

- GIF file (2 of them)

average distribution → fingerprint



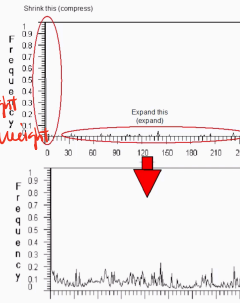
- Use 8-bit (byte) signatures and divide them into bins
  Count freq in each



Companding process

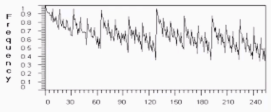
- Companding amplifies low signal and compresses high signal
- Results after companding of the GIF file
- Since original numbers were normalized 0..1 the result companding will be too

got more weight



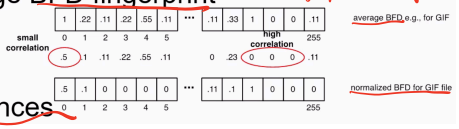
BFD Correlation

- GIF files



correlation high → difference low → all the frequencies align well

- Average BFD fingerprint



- Corr. differences

BFD "correlation" strength

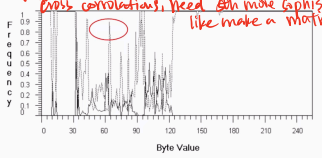
- If a byte value occurs with some regular frequency f in a file type then it is an important feature by definition for file identification
- Compute "correlation strength" between a type (e.g., GIF's) BFD and a particular file type
- Difference the BFD from an input file and its overall BFD fingerprint
  - Compute a "similarity" or "correlation" between an input file and its BFD

Byte Frequency Cross Correlation (BFC)

the fact that seeing one thing means seeing another

- In looking at Byte Frequency Correlation, sometimes it is possible to notice e.g., equal sized spikes between two byte markers (pair-wise) in a file type
- Example
  - HTML files, byte values 60 "<" and 62 ">"
- This is called "cross correlation"

frequencies are equal doesn't mean there's a cross correlation, need something more sophisticated. like make a matrix.



What is Metadata?

- "Metadata is structured information that describes, explains, locates, or otherwise makes it easier to retrieve, use, or manage an information resource." –

Understanding Metadata
National Information Standards Organization 2004

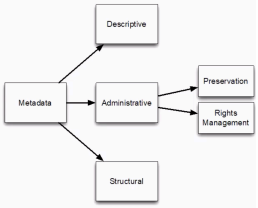
What is Metadata (2)?

- "Metadata is data that provides information about other data. Two types of metadata exist: structural metadata and descriptive metadata. Structural metadata is data about the containers of data. Descriptive metadata uses individual instances of application data or the data content."-

Wikipedia

Three main types of Metadata

- 1. *Descriptive Metadata* – Can be used to perform resource discovery and to identify data.
- 2. *Structural Metadata* – Can be used to define how data "fits together", e.g., what is the ordering of slides in a presentation, or what files are part of this package?
- 3. *Administrative Metadata** – Consists of management information about the data. For example when was it created, what type of file is it, who can access it, etc.



Dublin Core Significance

- 15 elements to describe *any* electronic resource
- Generic yes, powerful yes
- First widespread use of common ISO-11179 to specify metadata elements
  - Range, Controlled Value set
  - Specification for expressing metadata in a registry
  - https://en.wikipedia.org/wiki/ISO/IEC_11179
- Inspired Metadata capture syntax in
  - RDF, OODT Profiles, Schema.org, etc.

Classic Example: Dublin Core

- Started at 1995 workshop sponsored by OCLC and NCSA in Dublin Ohio
- Continuing Development => Dublin Core Metadata Initiative
- Originally 13 core elements, grown to 15 later
  - *Title, Creator, Subject, Description, Publisher, Contributor, Date, Type, Format, Identifier, Source, Language, Relation, Coverage, Rights*

Dublin Core Example

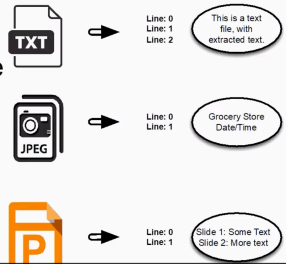
Title="Metadata Demystified"
Creator="Brand, Amy"
Creator="Daly, Frank"
Creator="Meyers, Barbara"
Subject="metadata"
Description="Presents an overview of metadata conventions in publishing."
Publisher="NISO Press"
Publisher="The Sheridan Press"
Date="2003-07"
Type="Text"
Format="application/pdf"
Identifier="http://www.niso.org/standards/resources/Metadata_Demystified.pdf"
Language="en"

Text Encoding Initiative (TEI)

- Develop guidelines for marking up electronic texts including novels, plays and poetry primarily to support human research in the humanities
  - http://www.tei-c.org/index.xml
- TEI specified a *Guidelines for Electronic Text Encoding and Interchange*
  - Specifies a header portion of the resource that consists of metadata about the work
  - TEI defined as SGML syntax and rules and in a Document Type Definition (DTD)
  - *TEI Lite* often used b/c TEI specification is so large

Text Extraction from File Types

- 1400+ file types
- Possibly different parsers for each type
- For each type, it has file header, body/ payload
- How to get text from each file type?



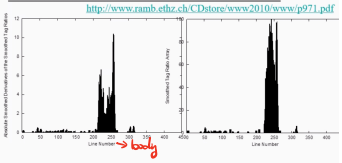
Tag/Annotation Ratios

- From WWW2010
  - Weninger et al.
- Take a web page, remove its scripts, stop words, styles, etc.
- Compute ratio of non annotated text (chars) with annotated (tags) chars
- Lines with more text than tags will have a higher ratio number
  - Can be used to isolate where text is in the HTML file, and to go after those lines

Algorithm 1 Compute Tag Ratios

```
INPUT: D
OUTPUT: T
D ← removeScriptTags(D)
D ← removeRemarkTags(D)
D ← removeStyleTags(D)
for all i ← 1 to |D| do
    x ← nonTagChars(D_i)
    y ← tags(D_i)
    if y = 0 then
        y ← 1
    end if
    T_i ← x/y
end for
```

http://www.ramb.ethz.ch/CDstore/www2010/www/p971.pdf



Basic Approach

- Identification of text
  - File Header – Usually Metadata information
  - Body/Payload of the file – File Specification can detail this – where is this in the file?
- Analysis of Text
  - Structural Analysis – e.g., Tag Ratios (WWW2010), Emphasis (IRI2014)
  - Contextual Analysis
    - Proximity, stylistic (IRI2004)
- Featurize
  - TFIDF (CACM 1975 – Salton), Clustering (Similarity), Metadata Extraction, Named Entity Recognition (NER)
  - BoilerPipe algorithm WSDM 2010

Tag Ratios Algorithms in General

- Also called "histogram clustering"
- Build a text to tag ratio array (TTRArray)
- Use following procedure
  - Remove empty lines and script tags
  - Initialize the TTRArray
  - For each line in document
    - X = non tag ASCII characters
    - Y = # of tags in the line
    - TTRArray[current line] = X if no tags, else TTRArray[current line] = X / Y

BoilerPipe – Main Features

- Text frequency of the whole corpus for common phrase extraction
- Presence of particular tags that enclose a block of text (heading tags, paragraphs, anchors, divs)
- Shallow features such as average word length, sentence length, absolute number of words in a defined segment
- Local context of the text – absolute/relative
- Heuristic features (number of words that start with uppercase, all caps, date and time tokens, link density)
- Density of text blocks

BoilerPipe Algorithm

- Human Annotators help train feature classifier based on extracted features
  - Label as content, or boilerplate
  - SVM learning/classifier