

Web Search: what's different?

- Extracting content from web-pages
 - synonymy in XML tags
 - tags embedded in content
 - Document Object Model
 - Finn's plateau method
- Detecting duplicated content
- Link structure: PageRank and HITS
- Amount of data and query types

Text Technologies

Web Search 1

Content Extraction and Duplicate Detection

Victor Lavrenko

Some Figures © Addison Wesley, 2008

Copyright © 2011, Victor Lavrenko

Copyright © Victor Lavrenko, 2010

Tags Across Sources

- Tags and formats will vary across sources

RSS: `<pubDate> Mon Sep 29 00:41:50 BST 2008 </pubDate>`

Twitter: `<published>2010-05-24T21:27:52Z</published>`
`<updated>2010-05-24T21:27:52Z</updated>`

AM: `<xn:publicationTime>2007-02-22T00:00:00-05:00</xn:publicationTime>`
`<xn:receivedTime>2007-02-22T00:00:12-05:00</xn:receivedTime>`

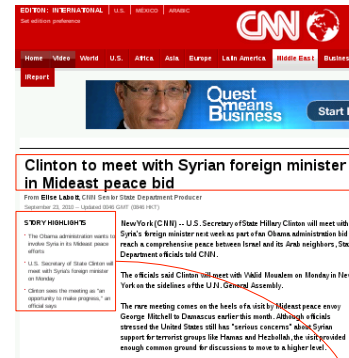
- Need a separate parser for every source
- Sometimes parsing XML is not enough:

```
<xn:vendorData>MRKTWIRE:Source=InvestSource, Inc.</xn:vendorData>
<xn:vendorData>MRKTWIRE:Geographic Code=RE/HUNTINGTON_BEACH</xn:vendorData>
```

dummy tag real tag masquerades as content

Copyright © 2011, Victor Lavrenko

Extracting Content from Web-pages



```
<style type="text/css">
.cnnFBRecBtn
{width:336px;float:right;margin:10px 0;clear:both;}

.cnnFBRecBtnBot
{width:420px;margin:30px 0 15px 186px;}

.cnn_strytcntrlft
{clear:both;}
</style>

</div>
<div class="cnnFBRecBtn" id="cnnStryRcmdBtn"><div>

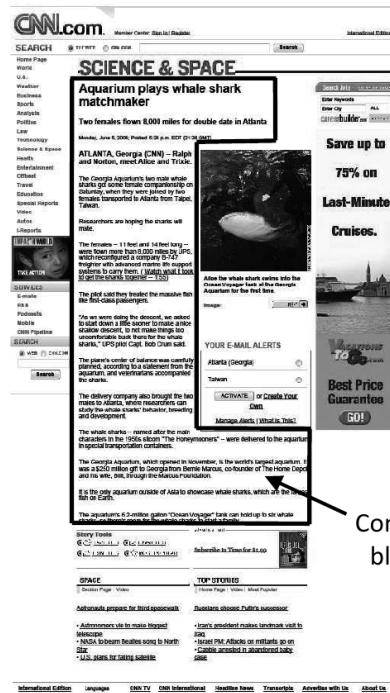
<!-- google_ad_section_start --><!--
startclickprintinclude--> <h1>Clinton to meet with
Syrian foreign minister in Mideast peace bid</h1><!--
startclickprinttextinclude--> <!--endclickprinttextinclude--><div
class="cnn_stryathrtmp"><div class="cnnByline">From
<b>Elise Labott</b></div><div class="cnnStryRcmdBtn"><div
class="cnn_strytmstmp"><script
type="text/javascript">if(location.hostname.indexOf(
'edition.' ) > -1) {document.write('September 23, 2010 --
Updated 0046 GMT (0846 HKT)');} else
{document.write('September 22, 2010 8:46 p.m.
EDT');}</script></div></div><!--endclickprintinclude--
--><!-- google_ad_section_end --><div
class="cnn_strytcntrlft"><!-- google_ad_section_start --
--><!-- CONTENT --><!--startclickprintinclude--> <script
language="JavaScript" type="text/javascript">var
clickExpire = ".1";</script> <div
class="cnn_strlyftcntrl"><div
class="cnn_strlylcntr"><div><b>STORY
HIGHLIGHTS</b></div><ul class="cnn_bulletbin
cnnStryHghlight"><!-- google_ad_section_start --><li>The
Chinese administration wants to
involve Syria in its Mideast peace
efforts.
U.S. Secretary of State Clinton will
meet with Syria's foreign minister
on Monday.
Clinton says the meeting is "an
opportunity to make progress" in
official talks.
The rare meeting comes on the heels of a visit by Mideast peace envoy
George Mitchell to Damascus earlier this month. Although officials
assured the United States still has "serious concerns" about Syria's
support for terrorist groups like Hamas and Hezbollah, the visit provided
enough common ground for discussions to move to a higher level.
</li></ul></div></div></div></div></div>
```

- No semantic tags
 - Tags = how the page looks
 - Highly varied across sources
- What if we remove all tags?

Copyright © 2011, Victor Lavrenko

Getting the text

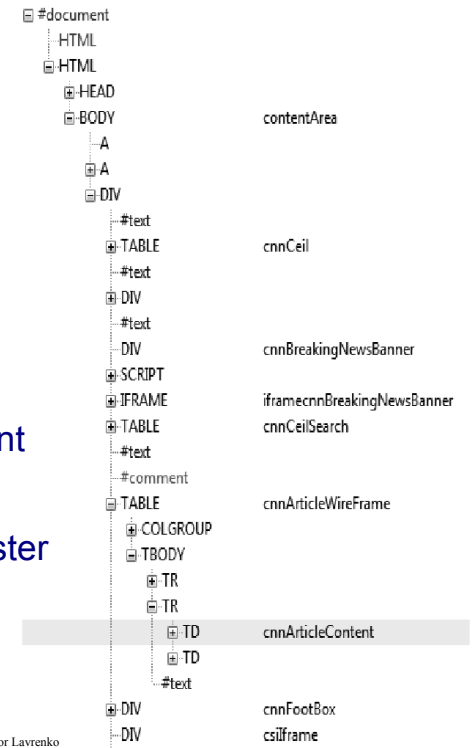
- Web pages contain junk
 - navigation controls
 - interactive / multimedia
 - advertisements
 - copyright notices
- Want to extract content
 - block of text containing description of the page
 - what you want to search



Copyright © 2011, Victor Lavrenko

Getting the text (2)

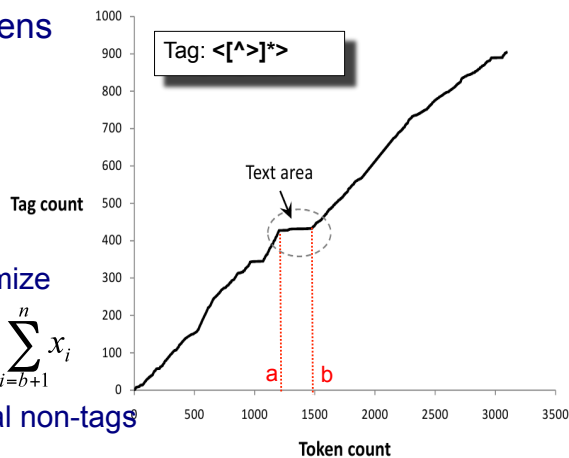
- Use DOM structure
 - document object model
 - logical layout of page
- Explicit markup
 - some sites will tag content
 - rare and site-specific
 - breaks with new webmaster
- Or learn statistically
 - brittle, needs re-training



Copyright © 2011, Victor Lavrenko

Getting the text: tag plateau

- Idea: text portion contains fewer HTML tags
 - plot #tags vs. #tokens
 - look for a plateau
- Formally
 - $x_i = 1 \dots$ a tag
 - $0 \dots$ non-tag
 - find a,b that maximize
 - $c = \text{total tags} / \text{total non-tags}$



Copyright © 2011, Victor Lavrenko

Tag plateau algorithm

- Naïve implementation: $O(n^3)$... hours per page
- Incremental: $O(n^2)$

$$\sum_{i=1}^{a-1} x_i + c \sum_{i=a}^b (1 - x_i) + \sum_{i=b+1}^n x_i$$

```

for a = 1..n:
  L += X[a]
  ...
  for i = a+1..n:
    R -= X[i]
    M += 1-X[i]
    ...
            
```
- Dynamic programming: $O(n)$
 - under 10ms per page if implemented correctly
- Limitations:
 - single "content" section surrounded by "junk"
 - alternative: slopes in k-token slices of HTML

Copyright © 2011, Victor Lavrenko

Web Search: what's different?

- Extracting content from web-pages
- Detecting duplicated content
 - exact duplicate detection: Adler32
 - near-duplicate detection: SimHash
 - statistical error bounds for LSH
- Link structure: PageRank and HITS
- Amount of data and query types

Copyright © Victor Lavrenko, 2010

Detecting duplicates Naively

- Compare every document against all others



```
for b = 1..n:  
  for a = 1..b-1:  
    if doc[a] == doc[b]: b is a duplicate of a  
    else: b is not a duplicate
```

- Computational complexity: $O(n^2d)$ [impractical]
 - n ... number of documents in a set ($10^6 - 10^9$)
 - d ... size of each document (10^3 words / 10^4 bytes)

Copyright © Victor Lavrenko, 2010

Detecting duplicates

- About 30% of web-pages are duplicates (2003)
 - mirrors, plagiarism, spam
 - also happens in news-feeds, speeches, essays
- Important to detect and remove
 - exact duplicate:
 - the content is identical bit-for-bit
 - non-content elements may differ
 - near-duplicate:
 - very similar but not exact: e.g 90% of content is the same
 - reworded a few sentences, or used different keywords

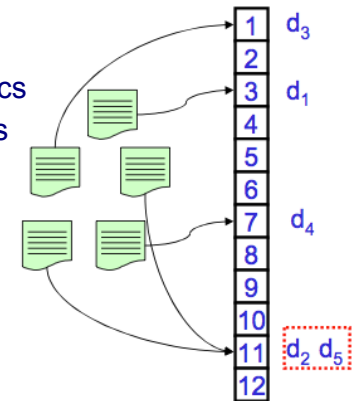
Copyright © 2011, Victor Lavrenko

Duplicates via Fingerprinting

- Idea: make duplicates fall into same bucket
- For each new document:

- compute its fingerprint
 - large number, which is:
 - always same for two identical docs
 - never same for two different docs
- hash [fingerprint]
 - collision \rightarrow duplicate

- Complexity: $O(nd)$



Copyright © Victor Lavrenko, 2010

Fingerprinting: Adler32

- Generate a unique fingerprint for each doc

- change 1 bit in a doc → get a different print
- maintain low chance of false collision
- remove non-content (punctuation, tags)
- add up byte values modulo a prime

```

adler32(str) {
  A = B = 0; m = 65521;
  while (c = *str++) A += (B += c);
  return ((A % m) << 16) | (B % m); }
    
```

	a	c	a	b	
ASCII	97	32	99	97	98
modulo 13					
sum B	97	129	228	325	423
sum A	97	226	454	779	1202

"0110 0111"

- Designed for data integrity applications:

- “magnify” small changes in input
 - transmission error, bad bit in RAM, bad sector on disk

Copyright © 2011, Victor Lavrenko

Detecting near-duplicates

- Locality-sensitive hashing (LSH)
 - similar checksums for similar documents
- Simhash fingerprint:
 - collect words, assign weights (e.g. frequencies)
 - compute a unique **b**-bit binary hash for every word
 - convert 0 → -1 and multiply by word weight
 - add by columns, set to 1 if $\sum > 0$, 0 otherwise
- high bit overlap in fingerprints → near duplicate
 - still need to compare fingerprints

Copyright © 2011, Victor Lavrenko

Intuition for Adler32 / MD5 / etc.

- Data = sequence of “units” (bytes, words, ...)
- each unit flips a subset of bits: $0 \leftrightarrow 1$
- bits scattered over the fingerprint, depend on unit
- flips depend on surrounding bits (so “UU” ≠ “”))
- P (False Negative) = 0
 - identical units → identical flips → identical result
- P (False Positive) = P (different inputs → same flips)
 - different unit → k flipped bits
 - FP = P (other units flip exactly those k bits back to original)
 - best possible case: no “regularities” in bit flips
 - each of the $2^{\#bits}$ fingerprints equally likely result given input
- Bad fit for near duplicates: “magnifies” small changes

Copyright © 2011, Victor Lavrenko

Tropical fish include fish found in tropical environments around the world, including both freshwater and salt water species.

(a) Original text

tropical 2 fish 2 include 1 found 1 environments 1 around 1 world 1 including 1 both 1 freshwater 1 salt 1 water 1 species 1

(b) Words with weights

tropical	2x0	1100001	fish	2x1	10101011	include	11100110
found	00	0111110	environments	00	1011101	around	10001011
world	00	101010	including	11	000000	both	10101110
freshwater	00	1111111	salt	10	110101	water	00100101
species	11	101110					

(c) 8 bit hash values

1 -5 9 -9 3 1 3 3

(d) Vector V formed by summing weights

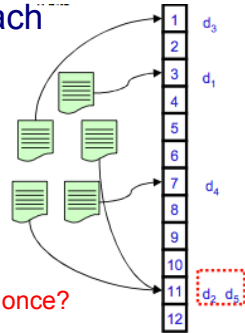
1 0 1 0 1 1 1 1

(e) 8-bit fingerprint formed from V

Detection: comparing fingerprints

- Have I seen a similar fingerprint before?
- Naïve algorithm: $O(n^2)$
 - compare every fingerprint to every other fingerprint
- Use hashing: $O(n)$
 - **b-bit fingerprint** \rightarrow **L groups of k bits each**
 - for each of the L groups:
 - hash using k bits of the group
 - if collision: compare to docs in the same bucket
 - $O(ndL + \text{comparisons})$

10101111
k bits



Why do we need L different groups? Can we use all bits at once?

Copyright © 2011, Victor Lavrenko

LSH Detection Errors

- Assume: L groups, K bits/group,
 - positives: pair of docs that are really near-duplicates
 - negatives: pair of docs that are really different
 - $p = P$ (bit is same for two real near-duplicates)
 - $q = P$ (bit is same for two different documents)
- P (False Negative Error) = $(1-p^K)^L$
 - none of the L groups matches on all K bits
- P (False Positive Error) = $1 - (1-q^K)^L$
 - at least one of the L groups matches on all K bits
- Both can be made arbitrarily small

Copyright © Victor Lavrenko, 2010

Summary

- Content extraction: DOM and Finn's method
- Duplicates: spam, plagiarism, 30% of the web
- Exact duplicates: byte-for-byte identical
 - $O(n^2)$ approach: compare every A to every B
 - $O(n)$: fingerprint \rightarrow hash, compare collisions
 - Adler32, MD5, etc: very sensitive to small changes
- Near duplicates: several words / sentences changed
 - simhash: locality-sensitive fingerprint
 - more words changed \rightarrow more bits in fingerprint change
 - K,L affect chance of detection after insertion / deletion
 - repeat L times with different K-bit fingerprints

Copyright © Victor Lavrenko, 2010