Name : Sharayu Sharad Mantri NetId : ssm171330

CRAWLING: E-Commerce for Food

Requirements for crawling :

- Java 8
- Apache Nutch 1.15
- Solr -7.3.1
- Selenium-2.42.2
- Firefox- V29
- Python

I used Apache Nutch 1.15 with Selenium 2.42.2 and firefox browser V29 to crawl all the web-pages. I gave around 50 seeds in seed list. These 50 seeds where injected in Apache Nutch Crawl db. Following was the flow of the crawl script that i ran to generate 45000 pages.

Why Use Nutch ?

- Production ready Web Crawler, Scalable , Tried and Tested
- Fine grained Configurations
- Relying on Apache hadoop data structure, batch processing
- MultiThreaded.
- Allows Custom Implementation for parse, index and scoring.
- Pluggable Indexing (solr, mongodb, elastic search, etc)
- Automation on checking broken links
- Handle Duplication
- User friendly ,Url Filters, Normalization

Crawl Jobs Sequence :

1. **Inject** (Seed list of size 50 urls)
2. Repeat for N Cycles :
    a. **Generate** : Select URLS from Crawl DB for fetching.
    b. **Fetch** : fetch URLS from fetch list.
    c. **Parse** : extract content metadata and links
    d. **UpdateDB**: Crawl Db status, score and signature
        i. Add new urls inlined or at the end of one crawler run.
    e. **InvertLinks**
        i. Map Anchor Text to document the link points to.
        ii. HashSet<Inlink (URL and anchor)>
    f. **DeDuplicate** Documents by Signature. (MD5 Hash is used)
    g. **Indexing** (Output of Invertlinks, given input to solr for indexing)
    h. **WebGraph** (creates web graph of the urls crawled, inlinks, outlinks and nodes)

Name : Sharayu Sharad Mantri NetId : ssm171330

Configurations for Nutch :

- Basic configuration of Java, selenium, solr and nutch.
- Ecommerce Specific configurations are as follows:
- Filters - conf/regex-urlfilter.txt
    - Commented -[?*!@=],Since amazon urls contained ? in everything
    - Skipped urls with
        - Corporate.walmart.com
        - help.walmart.com
        - And incorrect urls or seeds
    - Skip urls with suffix -(?i)\.(gif|jpg|png|ico|css|sit|eps|wmf|zip|ppt|mpg|xls|gz|rpm|tgz|mov|exe|jpeg|bmp|js)
- conf/nutch-site.xml
    - Allow Http, Https and do not allow ftp, sftp, etc
    - Ignore Outlinks to same domain : false
    - Ignore External Links (By Domain) : true
    - Threads per queue : Max number of threads allowed to access a queue 1
    - Add to Plugin.includes : solr, indexing, selenium
    - Add configurations for selenium
    - Minimum fetcher delay before hitting server of same domain : 2sec
    - Redirect Allowed : 1 (1 redirect allowed for amazon)
    - Generate Max Count : 50 (Max urls in single fetch list or single domain)
    - Fetcher Throughput threshold pages : 2 (Threshold of minimum pages per second. If the fetcher downloads less pages per second than the configured threshold, then fetcher stops)

Passing Collection for Index Creations :

1. Crawldb of apache nutch has all the data that has been crawled.
2. A small python script was run all the segments , and that script executed readseg api to get data from all the segments.
3. This api when called with parameters of to get title, url and content , created a dump for each segment.
4. A Python script was ran on this data to create, individual record files with Recno, Title, Url, Outlinks and Page Content which was then shared with Indexing person(Siddhant).

Stats of Data Crawled :

- Nutch provides an api called readdb with stats as argument, which gives the stats of the data crawled.
- Following is the screenshot of the output of stats of crawldb:
- This shows
    - Db_unfetched : 266256
        - These many urls are present in crawldb and are yet to crawled..
    - Db_fetcher : 41575
        - These many urls are crawled and data is fetched

- Db_gone : 1054
  - These many urls where deleted from crawled pages since they were 404 or had other errors with no data.
- Db_duplicate : 370
  - Around 370 duplicate urls where found and deleted from crawl data.
- Urls have assigned min score of 0.0 and max score of 3.93.

- All this data (41575) pages and links where shared with indexing system.

```
score quantile 0.01:    0.0
score quantile 0.05:    1.064905674752481E-7
score quantile 0.1:     1.1995147910948883E-7
score quantile 0.2:     2.1097414347397037E-7
score quantile 0.25:    1.1853253202589567E-6
score quantile 0.3:     1.3368090776566532E-6
score quantile 0.4:     1.5347666532571018E-6
score quantile 0.5:     2.8420344486322026E-6
score quantile 0.6:     9.730642766356543E-6
score quantile 0.7:     3.516350836678299E-5
score quantile 0.75:    4.556326965067523E-5
score quantile 0.8:     6.547591930823925E-5
score quantile 0.9:     2.3490156412976794E-4
score quantile 0.95:    0.0012198811928319517
score quantile 0.99:    0.035203814346948185
min score:      0.0
avg score:      0.0024160545605997137
max score:      3.933323383331299
status 1 (db_unfetched):        266256
status 2 (db_fetched):  41575
status 3 (db_gone):     1054
status 4 (db_redir_temp):       1179
status 5 (db_redir_perm):       1473
status 6 (db_notmodified):      69
status 7 (db_duplicate):        370
CrawlDb statistics: done
```

Statistics

42,000 urls crawled in 9 days. (12 hours a day).
Started with 450 urls in hour . (Jobs ran with parameters - top 200 urls and 25 threads)
Optimization made it 650 urls in an hour. (jobs ran with parameter - top 700 urls and 60 threads)

Handling Duplicate Data :

DeDup api of nutch is run in every crawl job, that handles duplicate urls fetched and remove then from crawl db. DeDup uses MD5 Signature to compare fingerprint of data fetched with existing data. If several entries share the same signature, the one with the highest score is kept. If the scores are the same, then the fetch time is used to determine which one to keep with the most recent one being kept. If their fetch times are the same we keep the one with the shortest URL. The entries which are not kept have their status changed to STATUS_DB_DUPLICATE,

this is then used by the Cleaning and Indexing jobs to delete the corresponding documents in the backends. Hence, Dedup commands run after every dp update to handle duplicate data.

Seed List :

I have used around 50 seeds for my crawler (70 different domains). It contained all the ecommerce websites for food. These are all highly rated websites for food e commerce which contains all groceries and beverages .

- https://grocery.walmart.com/
- https://www.naturesbasket.co.in/
- http://www.shopfoodex.com/
- http://www.mysupermarket.co.uk/
- https://www.supermarketitaly.com/ and many more

Along with US based websites, I also gave some international or UK based sights so that we can have variety in data.

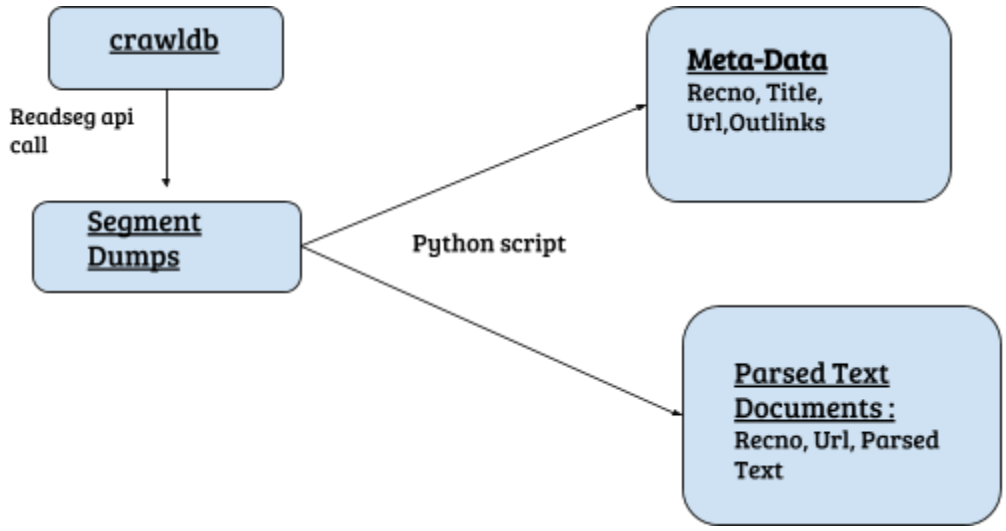
- https://www.ocado.com/
- https://naturebox.com
- https://thrivemarket.com/c/food
- https://www.asianfoodgrocer.com/
- https://www.internationalfoodshop.com/
- https://www.holamexico.de and many more

Along with ecommerce pages that sell all groceries, I also added few seeds(websites) that covered in specific type of food like meat, steak, ice-cream, wine, chocolates, cakes, mexican food, indian food , organic food, etc

- https://www.mexgrocer.com/
- https://minitacoshells.com
- http://ecomealorganic.com
- http://store.patelbros.com
- https://www.americanwestbeef.com
- https://www.benjerry.com/
- https://www.mymms.com and many more

Since, all these websites had different structures, I added generic schema to get title and entire data in content while parsing the page.

Links Data to Indexing and Relevance Model:



- The api **WebGraph** was run during every execution to update the existing webgraph and created inlinks, outlink and nodes data in crawldb.
- A small python script was run all the segments , and that script executed **readseg api** to get data from all the segments.
- This api when called with parameters of to get title, url , content, inlinks and outlinks , created a dump for each segment.
- A Python script was ran on this data to create, individual record files with Recno, Title, Url, Outlinks(crawled) and Page Content which was then shared with Indexing person(Siddhant).

Problems Faced and Solved

- Choosing Version of Apache Nutch :
  - I started with apache nutch 2.X , it has web-api where in we can schedule jobs and monitor. But 2.X is not stable and does not have as many features as 1.X like webgraph api and many more. Hence, I switched to nutch 1.X
- URL Problems :
  - By default nutch sets redirect to Zero and amazon most of the urls redirect. Hence. Made the redirect to 1. (page is allowed to redirect once)
  - Urls with image with image only where avoided by regex
  - Also, urls with ftp and smtp where not allowed to be crawled using regex.
- Domain Problem :
  - I wanted to crawl grocery.walmart.com and restrict crawling www.walmart.com, since it is not food ecommerce page. This was done by adding regex of www.walmart.com to be excluded in regex urlfilter file.
- Apache Nutch is multithreaded, we give number of threads as parameter to crawl script, each thread crawls a unique domain. But so if i launch 20 threads, say 18 threads finish

very quickly since, those domains or queues are small and end quickly. Where as big domains like grocery.walmart.com have huge pages. Hence, the never does not end quickly and only one page per second and crawled by the last active thread.
    - To avoid this, generate max count is set to 50 means maximum number of urls in a fetch list will be 50 and each domain will have at most 50 in queue.
    - And small batches of crawls jobs are run with high number of threads, like 200 crawled by 50 threads, or 500 crawled by 100 threads, so in each run more domains are covered , with limited urls from each domain.
    - Also, instead of running generate jobs with depth parameter, I ran jobs with topN parameter so that , top scoring N urls are selected for be fetched. This is good practice, since they top scoring urls are probably urls in interest.
- Filters : Pages like help.walmart.com, blog pages where being crawled which are not needed, and where not crawled future with regex url filter.
- Since, I had inserted very few seeds, even though I had huge number of threads running, crawling was very slow. I had to increase number of seeds.
- One Major Problem, I faced while scraping ecom website is DYNAMIC CONTENT.
    - Apache nutch by default doesn't scrape dynamic content. Using sitemap was first choice but not all sites use it and using Splash was next best option, but splash being light weight for faster execution wasn't able to load heavy commerce dynamic data. Hence, the last best option was SELENIUM.
    - Apache nutch has plugin utility where, we need to install selenium driver and plug it in nutch-config.xml and selenium is hooked with nutch. Hence, using selenium we were able to load e commerce dynamic content. (but it is really slow)
- Selenium is automates browser. Nutch loads its pages in selenium's browser. So, It's very important to find the compatible version of browser with particular version of selenium. Version mentioned in requirements are compatible versions with nutch 1.X.
- Selenium has an issue of memory leak. So due to this, many times browser gets stuck on a pages and do not terminate. Throwing port locked exception. This can be handled by using selenium hub/node configuration where, hub kills all the nodes which are inactive. But, it has its overhead and doesn't solve the problem completely, Hence I run a command in cron for every 15 mins to kill firefox. This releases the locked ports, to be used by active threads.

Monitoring

- Apache Nutch 1.X doesn't have web-ui , to monitor the jobs and logs error and manage crawling.
- Hence, I created a email script where after every couple of crawl jobs, it reads the logs and send email to me, giving details of the jobs like :
    - Number of jobs ran
    - Number of urls processed.
    - Amount of time each job took.
    - Urls where Error Occurred , along with error
    - Report failed Jobs.

Improvements

- Currently, When job is run, screen is popped with many firefox tabs. Need to run selenium-headless with apache nutch to run selenium browsers in background.
- Need to integrate with Interactive Selenium, since many web pages needs to be logged with zip code or user name and password to crawl the site. (Example : instacart , one of the largest e commerce for food website, was not crawled since, it needed Zip code to enter the site)
- There is no way only crawl Amazon Fresh without crawling Amazon.com. Need to focus more on targeted crawling.

Discussion

In the beginning, we had discussion with whole team regarding what are the tools which we will be using. Once, we started crawling was a very ad-hoc task, in the later week we met to discuss, how is output from the crawling expected so that Indexing was read it and process on it. Once, we were done , we met to integrate the whole data and see the final run is successful.

Conclusion

Apache Nutch is one of most scalable tools and can be easily plugged with many different tool and its easily configurable and can be easily used in distributed mode without having to make any major changes. But some more improvements can be done improve the speed but running it in distributed mode.