Toward an Intelligent Crawling Scheduler for Archiving News Websites Using Reinforcement Learning

Course: CS 6604
Instructor: DR. EDWARD A FOX
Team: Web Archive
Members: Xinyue Wang, Naman Ahuja, Ritesh Bansal, Siddharth Dhar, Nathaniel Llorens
Date: 12/10/2019
Motivation

- The Web is growing rapidly, but so too is information frequently disappearing from the WWW. Preserving the WWW is crucial to record the history of human society.
- How to preserve the web?
  - Crawling and saving
- The existing crawling model in the webarchive communities mainly adopts predefined crawling plans
  - Does not have the flexibility to capture the web dynamics
  - Needs a lot human efforts for designing the rules
- Can we have a smart web crawling scheduler reduce the human efforts efficiently?
Motivation

Web Archive Collector

Crawling Strategies?

WWW
A smart web crawling scheduler

- A scheduler that can automatically **predict** the web content changes and generate corresponding plans to capture the content
  - Web page content change
  - Web site structure change
- The traditional way: predict when the web content would change in the future
  - Use machine learning model to track the changes and make predictions
    - SVM, random forest, logistic regression etc.
  - Then build the scheduler based upon the prediction results
- What if a smart model generate the crawling plans directly?
- We propose to explore **reinforcement learning (RL)**
Traditional Method

- History Observation
  - Target
  - Target
  - Target

- Traditional Model
  - Prediction
    - Target?
    - Target?

- Crawl
  - A system to handle the prediction results and corresponding crawling behaviors
A RL idea
RL model basic

Web History Information

Crawler Brain

Crawling Decisions

Agent

Environment

state $S_t$

reward $R_t$

action $A_t$

$R_{t+1}$

$S_{t+1}$
RL model design - web page change

- Environment
  - A continuous sequence from the data set, which represents the historical records of web pages
- Action
  - Continuous prediction model
  - Sparse prediction model
- Reward
  - Reward the correct actions
  - Punish the wrong actions
Continuous Prediction

Agent

Make decision at each time step and then move forward

Crawl/Not Crawl

Timeline

Observation Space

Continuous Prediction Steps

History Observation

Target

Target

Target

Target

Target

Reward is given after each step

Duplicate

Valid

Duplicate

Valid

Miss
Sparse Prediction

1. Pick number of crawls (N) to be chosen
2. Generate N positions within the range (crawling plan)
Rewarding

- A RL model tend to avoid negative rewarded actions
- A RL model tend to stack more positive rewards as much as possible
- Reward is give after each episode (crawling plan) is generated

<table>
<thead>
<tr>
<th>Result Type</th>
<th>Continuous Prediction</th>
<th>Sparse Prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Duplicate (Crawl)</td>
<td>Negative</td>
<td>Negative</td>
</tr>
<tr>
<td>Valid</td>
<td>Positive</td>
<td>Positive</td>
</tr>
<tr>
<td>Miss</td>
<td>Negative</td>
<td>Negative</td>
</tr>
<tr>
<td>Duplicate (Not Crawl)</td>
<td>Positive</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Data Collection

- Web Archive collection from Archive.org: CNNfocuscrawls
  - From May 2019 to Oct 2019
  - Contains both CNN domain crawls and external links in CNN web sites
  - Crawling Tool: Heritrix
- **Our own focus crawl**
  - From Nov 6th to now
  - Contains only CNN domain crawls with 2 levels depth, crawled every hour
  - Crawling Tool: Web2Warc
- **Issue: gap from Archive.org and our own crawls**
  - We plan to compare the differences of our own crawl and the Archive.org collection to validate the capture accuracy of Archive.org
  - Will need until Archive.org publish the Nov data
Data Preprocessing

- The raw web archive files are in WARC format
- WARC -> Parquet
  - WARC is not flexible to use
  - WARC is inefficient to process
- We use Spark framework to batch converting WARC to Parquet efficiently
WARC and Parquet
<table>
<thead>
<tr>
<th>key</th>
<th>surtUrl</th>
<th>timestamp</th>
<th>originalUrl</th>
<th>minisize</th>
<th>digest</th>
<th>redirect</th>
<th>surtUrlMeta</th>
<th>contentLength</th>
<th>offset</th>
<th>filename</th>
<th>allheader</th>
<th>payload</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://info,7ba">http://info,7ba</a>,...</td>
<td><a href="http://info,7ba">http://info,7ba</a>,...</td>
<td>20180520</td>
<td><a href="http://0,7ba.info">http://0,7ba.info</a>,...</td>
<td>text/html</td>
<td>200</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>49471</td>
<td>855</td>
<td>IC-MAIN-201805201</td>
<td>application/xhtml</td>
</tr>
<tr>
<td><a href="http://Com,021cc">http://Com,021cc</a>...,</td>
<td><a href="http://Com,021cc">http://Com,021cc</a>...,</td>
<td>20180520</td>
<td><a href="http://021cctv.co">http://021cctv.co</a>,...</td>
<td>text/html</td>
<td>200</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>3613</td>
<td>10262</td>
<td>IC-MAIN-201805201</td>
<td>...text/html</td>
</tr>
<tr>
<td><a href="http://Cjp,nikki">http://Cjp,nikki</a>...,</td>
<td><a href="http://Cjp,nikki">http://Cjp,nikki</a>...,</td>
<td>20180520</td>
<td><a href="http://05a37y.nik">http://05a37y.nik</a>...</td>
<td>text/html</td>
<td>200</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>58476</td>
<td>12832</td>
<td>IC-MAIN-201805201</td>
<td>...text/html</td>
</tr>
<tr>
<td><a href="http://Cjp,with">http://Cjp,with</a>...,</td>
<td><a href="http://Cjp,with">http://Cjp,with</a>...,</td>
<td>20180520</td>
<td><a href="http://06.withtub">http://06.withtub</a>...</td>
<td>text/html</td>
<td>200</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>23773</td>
<td>23958</td>
<td>IC-MAIN-201805201</td>
<td>...text/html</td>
</tr>
<tr>
<td><a href="http://Cjp,akl">http://Cjp,akl</a>...,</td>
<td><a href="http://Cjp,akl">http://Cjp,akl</a>...,</td>
<td>20180520</td>
<td><a href="http://00a4.cou">http://00a4.cou</a>,...</td>
<td>text/html</td>
<td>200</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>77291</td>
<td>31857</td>
<td>IC-MAIN-201805201</td>
<td>...text/html</td>
</tr>
</tbody>
</table>
Find Web Content Change

- Two types of web content change:
  - Web page change
  - Website structure change
- Get the dataset labeled
  - Find unique web copies for a web page from the web archive
  - Find unique website treemap from the web archive
Baselines for predicting web content change - WebPage

Data Preprocessing

Parquet Files

PySpark

{ timestamp: 20191107160812, payload: <!DOCTYPE html> }

Text Similarity
HTML DOM Similarity
HTML Style Similarity

Facebook Prophet

SVM
Random Forest

<table>
<thead>
<tr>
<th>ds</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.0</td>
</tr>
<tr>
<td>1</td>
<td>2.0</td>
</tr>
<tr>
<td>2</td>
<td>3.0</td>
</tr>
<tr>
<td>3</td>
<td>0.0</td>
</tr>
<tr>
<td>4</td>
<td>1.0</td>
</tr>
</tbody>
</table>

array([1., 2., 3., 4., 1.]),
array([2., 3., 4., 1., 3.]),
array([3., 4., 1., 3., 5.]),
array([4., 1., 3., 5., 2.])
- The SVM model is able to solve nonlinear estimation problems; successful in time series forecasting.
- Random Forests generally don’t fit very well for time series trends and seasonalities.

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
<th>Precision</th>
<th>Recall</th>
<th>F1 Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Support Vector Machine</td>
<td>0.769</td>
<td>0.74</td>
<td>0.77</td>
<td>0.75</td>
</tr>
<tr>
<td>Random Forest</td>
<td>0.615</td>
<td>0.60</td>
<td>0.62</td>
<td>0.61</td>
</tr>
</tbody>
</table>
Predicting web page change - Facebook Prophet
Predicting web page change - Facebook Prophet
Predicting web page change - Facebook Prophet
Constructing unique web site treemap from the web archive

1. **Page level** prediction will help decide to **crawl particular** web page.

2. Consider the case when **multiple new web pages** are added, maybe during a particular time and day. We may want to crawl all these pages.

3. **Sitemap prediction** will help decide to crawl **whole website** of not.

4. **Data archived** contains data of **multiple media-type** and **status code**.

5. **Initial preprocessing** is required to build sitemap, ‘text/html’ and ‘200’.

6. **Tree-Based Structure**, ‘hostname’ is ‘root node’, ‘path act’ as ‘nodes’.
Baselines for predicting web structure change - TreeMap

1. Built **training data** based on the window size of **5 hours**.
2. Used **percentage of nodes added** from the last crawled datapoint to evaluate the **labels**.
3. Number of nodes added will act as the input feature.
4. Used SVM and Random Forest models to predict if the website should be crawled.
5. Prediction is if we want “to crawl whole website in the next hour”

<table>
<thead>
<tr>
<th></th>
<th>Accuracy</th>
<th>Precision</th>
<th>Recall</th>
<th>F1-Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>0.71</td>
<td>0.53</td>
<td>0.71</td>
<td>0.61</td>
</tr>
<tr>
<td>RF</td>
<td>0.61</td>
<td>0.59</td>
<td>0.59</td>
<td>0.59</td>
</tr>
</tbody>
</table>
Predicting SiteMap change - Prophet
Predicting SiteMap change - Prophet

![Graph showing trend and daily changes over time]
Constructing unique web site treemap from the web archive (continued)

- The sitemaps generated by our algorithms are important for predicting the change of a site’s tree structure, but they are also useful for visualizing how and where a site is changing.
- Algorithm, when given input of two sitemaps, can show differences between them in a visualization known as a treemap (unrelated example to the right).
- Allows a user to see how a site is changing between snapshots of a site.

Credit: [https://en.wikipedia.org/wiki/Treemapping](https://en.wikipedia.org/wiki/Treemapping)
RL model evaluation - web page change

- Use the same dataset as web page prediction baselines
- 100000 training steps for each model
- 8:2 training, testing ratio
- Two RL learning policies: DQN, PPO
- Two types of rewarding functions: fixed score, scaled score
## RL model evaluation - web page change

### Reward Setting

<table>
<thead>
<tr>
<th>Result Type</th>
<th>Fixed Score</th>
<th>Scaled Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Duplicate (Crawl)</td>
<td>-2</td>
<td>-2</td>
</tr>
<tr>
<td>Valid</td>
<td>15</td>
<td>$20 \times (1 - \frac{D_t}{D_{\text{max}}})$</td>
</tr>
<tr>
<td>Valid (Exact Match)</td>
<td>20</td>
<td>N/A</td>
</tr>
<tr>
<td>Miss</td>
<td>-10</td>
<td>-10</td>
</tr>
<tr>
<td>Duplicate (Not Crawl)</td>
<td>2</td>
<td>N/A</td>
</tr>
</tbody>
</table>

### Learning Policy Used

<table>
<thead>
<tr>
<th></th>
<th>Max Fixed Score</th>
<th>DQN</th>
<th>PPO2</th>
<th>Max Scaled Score</th>
<th>DQN</th>
<th>PPO2</th>
</tr>
</thead>
<tbody>
<tr>
<td>continuous prediction</td>
<td>50</td>
<td>-28.5</td>
<td>27.6</td>
<td>50</td>
<td>-26.8</td>
<td>30.8</td>
</tr>
<tr>
<td>sparse prediction</td>
<td>20</td>
<td>N/A</td>
<td>9.7</td>
<td>20</td>
<td>N/A</td>
<td>10.5</td>
</tr>
</tbody>
</table>
Conclusion and future work

- At current stage, we can not compare the RL model with baselines directly since they are different tasks
  - Further experiments are needed to build a system to work with baselines for crawling plan generation
- We also need more training data with a longer period of various of types
- RL model has the potential to solve our web crawling
- Quantitative evaluation is needed to reflect the actual performance of the model compared with baselines or predefined policies
- Investigate the web site structure change model in the future
- The web page change and web site structure change may also be combined as a single model