An intelligent web spider for online e-commerce data extraction

Ranjani Murali
Computer Engineering Department
Sardar Vallabhbhai National Institute of Technology
Surat, India
ranjani_murali@india.com

***Abstract*—The growing arenas of e-commerce has its impact on the inflation and CPI. Automated web scrappers or web spiders are softwares that can be used to extract data that is available online. The proposed work here uses a python based web scrapper to extract online price information through automated browsing using Html DOM tree based structure to extract the data rich and relevant data regions of a web page. Further the extracted data is cleaned, integrated into a homogenous form for short term inflation calculation forecasting of vegetables and fruits category and analysis purposes.**

***Keywords*—Web scrapping, web spiders, online prices, Big data, automated data extraction, CPI data, time series data, inflation forecasting, Python.**

I. Introduction (*Heading 1*)

The WWW is ever mutating and growing with its expanse reaching every sphere of society. The size of the World Wide Web, the Indexed Web contains at least 4.48 billion pages as on 01 February, 2018. The Deep web or unindexed web is estimated to be 90 times of the Indexed web. The exploration and collection of data from this vast expanse of information source is known as web crawling or web harvesting and the software is called a web scrapper or a web spider.

These automated or semi-automated software explore web pages and download the content which are relevant to the isometrics. It is a tool for the search engines and other information seekers to gather data for indexing and to enable them to keep their databases up to date [29]. Web crawlers are mostly used by either search engines like Google, Yahoo, Bing etc. or e-commerce giants like Amazon, Flipkart to have comparative pricing. This automated software crawls the web for context search specific data in case of e-commerce oriented application and gathers web page content for analysis. The types of crawlers include focused, simple and distributed where focused crawler only crawls to extract specific information from web pages, a simple crawler extracts all possible web pages without looking for specific data from the given set of seed links and a distributed crawler executed crawling via multiple server requests and is more efficient as discussed in [55].

Most of the information on the web today is in the form of HTML (Hypertext Markup Language) documents which are viewed by humans with a browser. Given that the format of HTML documents is designed for presentation purposes, not automated extraction, and the fact that most of the HTML content on the web is ill-formed, extracting data from such documents can be compared to the task of extracting structure from unstructured documents [27].

Extracting structured data from Web sites requires solving five problems:

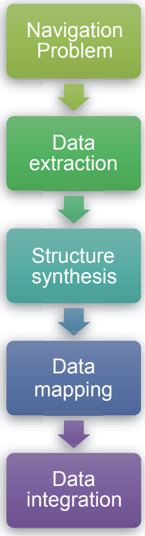


Figure 1: Web crawling process [27]

The navigation problem constitutes the function of getting web pages and navigating to the most relevant ones. After navigating to the most relevant data extraction of the unstructured web pages is encountered. This unstructured or semi-structured data is then analysed for structure synthesis. This analysis follows by mapping of the data to get relevant data components out of the unstructured collection and to correlate it for most relevant information. Finally the extracted data is integrated and completely collected together for further analysis.

This research work aims to extract online product information for CPI and Inflation related studies. The increase in the number of e-commerce outlets has started to influence and reflect the inflation index as discussed in [2]. A method to scrap this data and analyse the prices of fruits and vegetables has been proposed in this paper with an intent to forecast short term inflation. The following report is organised with the next section discussing relevant work done in the sphere and further discusses the framework employed to extract the data

978-1-5386-5657-0/18/$31.00 ©2018 IEEE

and analysis procedure followed by the results and conclusion derived by the designed system modelled after [21].

II. LITERATURE SURVEY

Several research work has been pursued in the area of web crawling with use of techniques ranging from use of DOM (Document Object Model) based HTML tree structure [42] [59] to use of advanced techniques of Artificial Intelligence and Machine Learning [4] [7] [22] [36]. Use of tree based parsing has also been implemented to improve the search efficiency of the web scrapper. Initial web page is taken as input from the user. This web page is automatically parsed for all possible links to create the initial set of web pages. Each tree i.e. link is taken as the DOM tree where the automated parsing and construction of HTML DOM tree is done in [59].

A partial dynamic and static web scrapping methodology is implemented in [42] to automate the process. Distributed method is explored to increase the execution potential in [47]. A reinforcement method is employed to explore only those nodes of the tree which have higher relevance via a swarm intelligence technique of Ant colony optimization in [22]. Using best first search and Genetic algorithm techniques in [9] optimization of searching and extraction of relevant data in the HTML DOM tree data structure is done. A graphical methodology based implementation has also been proved effective in the case described in [46] where using graph based data structure and usage of the patented page rank algorithm of Google has been used to explore the web data scrapping methodology. Use of alternative xpath method has been effective for extraction of necessary relevant data in [27] where the URL ordering has helped increase the efficiency of data extraction.

Machine learning has been used to train the data with few examples of data extracted from semi-automated method to help the system recognize the correct portion of the input html code and extract the data relevant data intelligently. Machine learning techniques are implemented to enable efficient forest creation and DOM Tree based HTML data extraction in [59].

A technique to limit the level of parsing is used by lower bounds for tree edit distance algorithm in [42]. The synthetic intelligence induced is used to determine the level of link exploration and web page generation and helps in pruning of the dataset at the optimum level to avoid local re-reference and irrelevant parsing. Swarm intelligence techniques is used in [22] to implement forest pruning and optimisation along with lower bound specification. Genetic algorithm is used with an initial regular expression and using mutation techniques and cross over techniques new individuals (regular expressions) are generated to get the exact data necessary in [9].

After creation of a set of useful web pages each HTML DOM tree can be explored to extract most relevant data. Here the relevancy of each web page is established using some fitness function. Machine learning or text analysis techniques is explored to find relevant information from the web pages via relevancy of text analysis via keyword approach, content analysis or regular expression building [9] where machine learning techniques like SVM, Naïve Bayes [18] and Neural Networks method are used for the extracted data from HTML web pages in [7] [35].

Analysis of text gained from can be done by either a fitness function in genetic algorithm. This function would need another layer of machine learning technique to decide the relevance of the data gained. The fitness function is a reinforcement learning machine learning module in [30]. Hence each iteration of the system to extract the relevant leaf node data will get strengthened via reinforcement techniques and the relevant data can extracted at correct depth.

III. BRIEF DESCRIPTION OF WEB SCRAPPER SYSTEM

The system for the proposed web scrapper module consists of four main components whose process flow is illustrated by figure 2. The system has been developed using Python 3 language for web scrapping. The functionality of the automated software has been segregated as modules of web scrapping, data extraction, data cleaning and integration system and an analysis module. The components and methodology are described below.

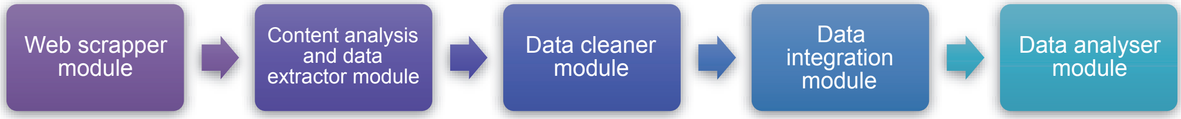


Figure 2: System description

Languages used: Python 3, R

Tools used: Selector gadget was used to analyse the webpage structure and generate Xpath rules to locate the data rich regions and help the system to capture the relevant data. Browser plugins were used to enhance and enable the web page capture extraction and human browsing behaviour emulation.

Python libraries used: Selenium module has been used to automate web browsing by automatically launching web drivers emulating dynamic events like scrolling, button click etc. CSV module has been used to perform automated read write operations in comma separated(CSV) files. Beautifulsoup4 has been used for parsing the html code and to obtain HTML CSS elements from the webpage. Waybackpack has been used as the python API to interact with the web archive database and to extract the archived past versions of the websites containing the online prices existing previously.

Web data extracted are of Vegetables and Fruits prices, Beverages and Drink prices & Pulses and Grain prices. The extracted web pages are parallel processed to extract most relevant data which are then integrated in three separate CSV files of each aforementioned categories which are then used for further analysis for inflation forecasting and prediction purposes.

The four integral components analyse the data in a linear format where several pages are processed in a parallel mode. Each of the components are described in the sections below.

A. Web scrapper module

The web scrapping module here is a focused web crawler with intent to gather data specifically about domain specific data in this case it is fruits, beverages, pulses and vegetables prices. The web scrapper module takes the initial web page link as input and parses through a set of most relevant links applying different techniques to overcome web pages complexity.

Some of the customised functions employed to imitate *human web browsing behaviour pattern* are:

- Infinite scrolling: It is applied to access the entire web content which does not appear immediately after loading the web page link. This techniques ensures that the whole web page content is visible and accessible through the browser.
- Automated navigation: It is employed to navigate and access special sections of the webpage which are not immediately loaded when the browser opens the web page. Only partial content display is followed by several web pages which require the user to click on displayed buttons to load the consecutive data.
- Focus: To detect automated crawlers or spiders certain sites employ a check to detect if the browser is in focus in the displayed monitor. Hence this technique is employed to emulate in focus behaviour of browsers.
- Form filling: Certain web pages require the user to give details of their location before the relevant data page is displayed. This automated form filling results in the relevant data being displayed and accessible through the browser.
- Scheduler: Extraction of web pages needed to be contiguous and consistent. Hence the automated system schedules the web extraction at the same to acquire the data chronologically.

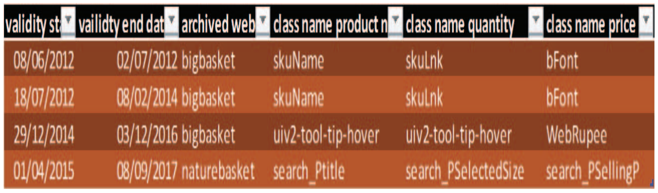
The module extracted the most relevant pages and the subsequent modules completed further processing to obtain clean and analysable data. The data was extracted on a daily basis from live web pages. Several archetypes were employed in the code to make the automated system independent of the web page structure and content alignment along with imitating human web browsing patterns discussed previously.

Another source of archived data was used for extending the project for inflation forecasting as it requires several years past data. To comply with this condition use of past data was necessary hence use web archive which contained older versions of the website were also scrapped to contain data from 2012 to 2018. Webarchive.org website was scrapped to access previous versions of these web pages to acquire the online prices data of previous years.

The web scrapping system used a scheduler code to run the script automatically to extract data from different web sites at the same time each day. The extracted data is then cleaned and integrated with the file containing data of only that website. These files are then integrated with their data timestamp, source and city details for further processing.

B. Content analysis and data extractor module

This module functions as an analyser and data extractor of the collected unstructured web pages. This module recognises the useful data regions and extracts these units specifically. A rule-set CSV file is previously compiled to instruct this module on the data rich regions and the function input parameters to locate the regions from the unstructured web data.



validity st	vailidty end dat	archived web	class name product n	class name quantity	class name price
08/06/2012	02/07/2012	bigbasket	skuName	skuLnk	bFont
18/07/2012	08/02/2014	bigbasket	skuName	skuLnk	bFont
29/12/2014	03/12/2016	bigbasket	uiv2-tool-tip-hover	uiv2-tool-tip-hover	WebRupee
01/04/2015	08/09/2017	naturebasket	search_Ptitle	search_PSelectedSize	search_PSellingP

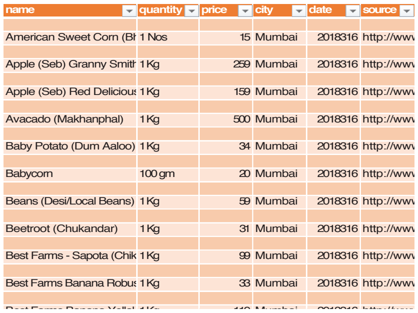
Figure 3 Rule set file snapshot

- Selector Gadget: This tool was used as a browser plugin to analyse the captured web page from the browser and obtain the set of Xpath expressions which serve as rule set for data extraction for this module.
- Rule set: A separate CSV file was taken as input by this module for data rich region detection and data extraction.

The content structure is previously analysed to find data representation pattern to formulate the rules stated in Xpath and CSS element for the python selenium module. These Xpath rules and CSS elements are repeated throughout the website and are used to extract the relevant data regions. These extracted data regions are then stored in variables for further text extraction and categorisation. The extracted cleaned text is then processed by the subsequent modules.

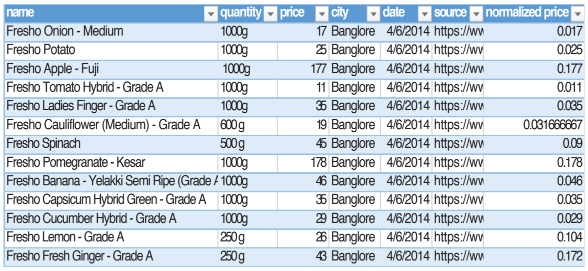
C. Data cleaner

The extracted text from the data regions serves as input for the data cleaner and integrator module. This module produces the cleaned and normalised dataset as output. Figures 5 6 and 7 illustrate the data cleaning process and the input file and output file snapshots. The extracted text is cleaned for irrelevant and useless portions of it. It is then segregated into portions of data and represented in a homogenous form. The difference of units, representation terms inherent due to extraction from different websites are all standardised into a fixed representation and represented in a CSV file. This CSV file is then used by the analyser module.



name	quantity	price	city	date	source
American Sweet Corn (Bh	1 Nos	15	Mumbai	2018316	http://www
Apple (Seb) Granny Smith	1 Kg	259	Mumbai	2018316	http://www
Apple (Seb) Red Delicious	1 Kg	159	Mumbai	2018316	http://www
Avacado (Makhanphal)	1 Kg	500	Mumbai	2018316	http://www
Baby Potato (Dum Aaloo)	1 Kg	34	Mumbai	2018316	http://www
Babycorn	100 gm	20	Mumbai	2018316	http://www
Beans (Desi/Local Beans)	1 Kg	59	Mumbai	2018316	http://www
Beetroot (Chukandar)	1 Kg	31	Mumbai	2018316	http://www
Best Farms - Sapota (Chik	1 Kg	99	Mumbai	2018316	http://www
Best Farms Banana Robus	1 Kg	33	Mumbai	2018316	http://www
Best Farms Banana Yellal	1 Kg	110	Mumbai	2018316	http://www

Figure 4: Uncleaned data



name	quantity	price	city	date	source	normalized price
Fresho Onion - Medium	1000g	17	Banglore	4/6/2014	https://ww	0.017
Fresho Potato	1000g	25	Banglore	4/6/2014	https://ww	0.025
Fresho Apple - Fuji	1000g	177	Banglore	4/6/2014	https://ww	0.177
Fresho Tomato Hybrid - Grade A	1000g	11	Banglore	4/6/2014	https://ww	0.011
Fresho Ladies Finger - Grade A	1000g	35	Banglore	4/6/2014	https://ww	0.035
Fresho Cauliflower (Medium) - Grade A	600 g	19	Banglore	4/6/2014	https://ww	0.031666667
Fresho Spinach	500 g	45	Banglore	4/6/2014	https://ww	0.09
Fresho Pomegranate - Kesar	1000g	178	Banglore	4/6/2014	https://ww	0.178
Fresho Banana - Yelakki Semi Ripe (Grade A	1000g	46	Banglore	4/6/2014	https://ww	0.046
Fresho Capsicum Hybrid Green - Grade A	1000g	35	Banglore	4/6/2014	https://ww	0.035
Fresho Cucumber Hybrid - Grade A	1000g	29	Banglore	4/6/2014	https://ww	0.029
Fresho Lemon - Grade A	250 g	26	Banglore	4/6/2014	https://ww	0.104
Fresho Fresh Ginger - Grade A	250 g	43	Banglore	4/6/2014	https://ww	0.172

Figure 5: Cleaned data

The data cleaner module consists of six main subsystems which are explained below:

- Data segregation: The input text given to this module contains all the relevant information which the website contained which includes product name, price, quantity source and type of commodity. This data is analyzed by this section and the separate data types of the respective categories are segregated and stored according to their data category in CSV files.
- Cleaning missing values: The segregated data then might contain missing values due to incomplete representation or due to inclusion of irrelevant data while scrapping. These empty or incomplete rows are processed and recovered if possible or removed from the data file.
- Irrelevant data removal: After the removal of missing data this system then removes irrelevant data by identifying the type of data and the predefined rule set. If the category of data present does not match the defined data type then the row is analysed for relevance or misalignment. If the row is not misaligned then this row is discarded.

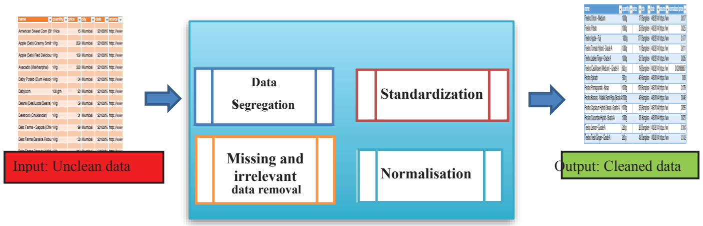
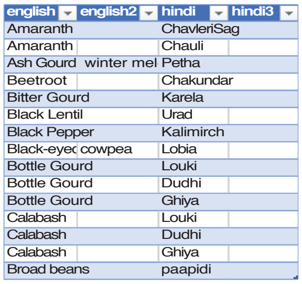


Figure 6: Data cleaning module

- Units standardisation: The data scrapped consists of several different sources and hence has different unit representation for the same product. This component converts the several different units present in the data set to a standard unit via use of conversion ratio. This makes the data comparable.
- Normalisation: This component normalises the data to a standard format to comparable quantities and makes the data set into a homogenous time series data representation.
- Format standardisation: This component standardises the representation format after acquiring the data from various data sources. This module analyses the data rows and applies a check to see if the data row complies with the preset data format if not converts the data row into requisite format.

D. Integration module

This module has a main function of analysing the cleaned data set and integrating different product data into separate data sets or files. Use of different data sources have resulted same products listed under different names as websites have the products listed in Hindi and English. Hence same products have been listed as different names and won't be recognised as the same.



english	english2	hindi	hindi3
Amaranth		ChavleriSag	
Amaranth		Chauli	
Ash Gourd	winter mel	Petha	
Beetroot		Chakundar	
Bitter Gourd		Karela	
Black Lentil		Urad	
Black Pepper		Kalimirch	
Black-eyec	cowpea	Lobia	
Bottle Gourd		Louki	
Bottle Gourd		Dudhi	
Bottle Gourd		Ghiya	
Calabash		Louki	
Calabash		Dudhi	
Calabash		Ghiya	
Broad beans		paapidi	

Figure 7: Dictionary

This problem has been addressed by use of a dictionary a snapshot has been illustrated in figure 7. Each of the possible names of Fruits and vegetables have been listed and the subsequent rows contain all possible variants of the product in both English and Hindi. This dictionary is used as an interface to fetch and integrate all same products into one category.

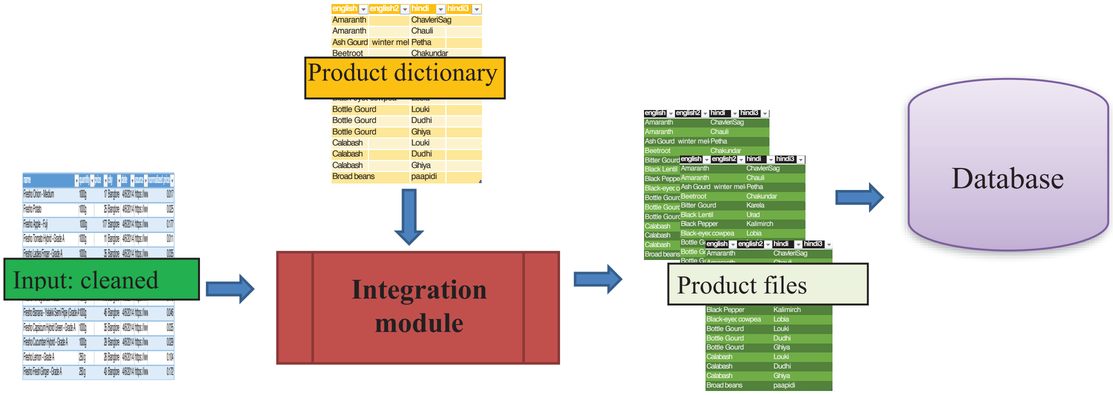


Figure 8: Data integration

Integrator module comprised certain difficulties where the product names where the product names were listed with different spelling while translating Hindi words into English. These were not recognised by the integrator module. Certain product names were not listed in the dictionary were also not recognised by the module. Constant updating of the dictionary is necessary to encounter these products. Some of the products were miss classified due to their name containing another product's name for example banana and banana leaves were classified as one product. This issue has to address by maximum string matching of product names where the algorithm looks for the length of product name matched and incase of multiple matching it selects the longest string as product category.

E. Data Analyser module

The data analyser module asses the quality of data after cleaning and is used as a visualiser before forecasting and prediction purpose. This module serves an exploratory system before actual utilisation of the data to assess the quality relevance and usability of the data.

The obtained product specific data is visualised to check its authenticity and usefulness. The major errors of product misclassification, missing values are all recognised here. These errors are later rectified by increase of dictionary content or change in the algorithm capacity.

IV. Web Crawling Strategies

Data extraction types include both live and archive data. The live nature of websites prevents direct access to any of the past data. The changes are immediately in effect as soon as a web page is modified and the previous price data are not accessible. Hence this system extracts data of two different kinds. The live data extraction is done via use of scheduler and is extracted every day at a fixed time-slot. The past data extraction is done from internet archived files

A. Live data extraction

A The live web page crawling system operates on a daily basis with a predefined time of execution. This system operates to extract the daily price values via dynamic web crawling and stores this data after process in a time series data set. As the system interacts directly with the web browser and mimics human behaviour via python selenium module this system is classified as dynamic crawling. Figure 9 illustrates the data flow and the output obtained by this system. The extraction of three different websites happens simultaneously in a parallel manner.

The scheduler code first initiates the automated code to execute three browsers instances. The web scrapper is scheduled to execute at specific time for extraction of data from various websites. Each of these instances scrap different websites in a parallel manner simultaneously. The webpages are scrapped by the data extraction module and then the data rich regions are recognised by the content analyser module with the pre-built rule set.

Using data from different retailers and sources is extracted in the raw format and then cleaned by the data cleaner module for consistent representation. This cleaned data is maintained in a separate CSV file for each retailer. The further integration of data is done after homogenous representation of the data with normalisation, standardisation of units and representation. The data once acquired has source, city and timestamp data inserted along with it. This CSV file comprising of data from all retailers and all cities is further analysed via use of weights according to official statistics and individual city weight as described in the previous section.

The data extraction system relies on the predefined set of rules for data location and extraction. Due to this limitation the system works only if the web page's basic internal architecture remains the same. The changes to the fundamental structure of the website require that the system's rule set be changed as well. This reduces the robustness of the web scrapper. An intelligent algorithm to automatically generate the rule set can be used in the future to increase the robustness of the system.

The complexity of the websites also prevent extraction of multiple city data in the case of bigbasket website. The complexity and required human behaviour emulation is higher in this website's case. This complexity is applied to prevent web scrappers from acquiring website data and using for commercial purposes.

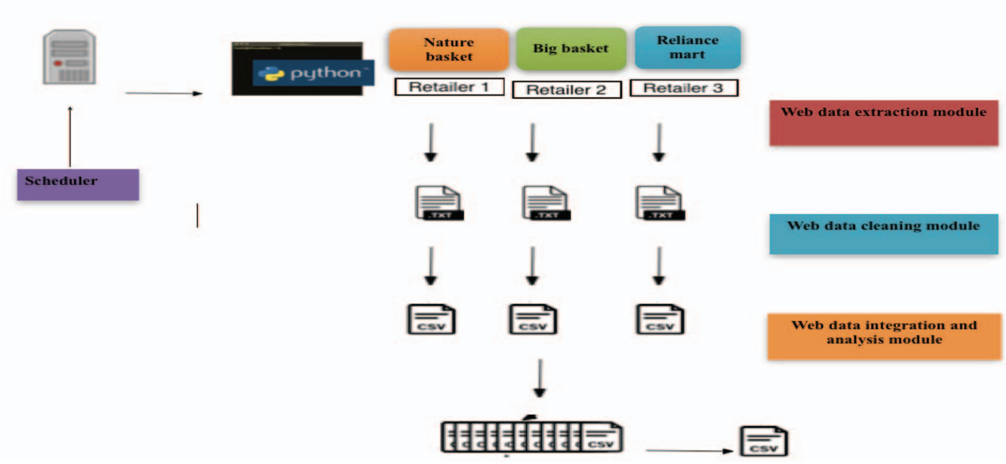


Figure 9 : Live data Extraction

B. *Archive data extraction*

The live nature of websites prevents the access of past data which is essential for this research work as the past version of the website contain the historical price data. This issue was addressed by obtaining the archived data of the internet webpages which are stored in Webarchive.org. This website stores snapshots of all available pages hosted in the internet as a library of internet pages. These online archive contains the snapshots of the online retailer and in them contains the historical product and prices data. This website is accessed through a python API called waybackmachine to access and download specific website snaps as per specified date interval.

The accessed HTML static pages are used for data extraction via static web crawling as there is no direct interaction or dynamic event simulation like scrolling or form input in this system. The analysis of the web content is done after the downloading the web content in a remote machine. The static web page content are analyzed and the data rich regions are identified due to repeated pattern representation inherent in web page designing.

The system initiates a request through python API waybackmachine to download set static web pages of a specified time interval. The snapshots of the web page are then sent from the Internet archive database to the local machine. These static html pages are then directed to the web data extraction module where a separate rule set is maintained for archive data. These rule sets are pre build by analysing the web pages to find similarity and the web pages are extracted till the rule set fails. A new rule is then added to the rule set in a recursive manner. This back propagation of this module enables this system to reiterate and increase the system capacity for extraction. After multiple reiterations the web page data are extracted into text bits which are then processed by the data cleaner module.

The data cleaner module performs the missing data removal, irrelevant data removal, data formatting, data normalisation, data alignment and then represents the input text into a cleaned CSV file for further processing. This cleaned CSV file is then processed by the data integration and analysis module. This system uses product matching algorithm to extract individual products based a dictionary approach to detect all possible versions of the product name and then integrates this data into a single file. This module integrates separate product into respective CSV files and later these files are analyzed to find data discrepancies or product misclassification.

The issues faced were that only snapshots of the web page were accessible from the internet archive data. The complete web page was not available due to data storage limitations of the web firm. Hence only some products were having complete historical data while others had scanty historical data. The webpages contained only some cities snapshots and not all of them hence city wise data was unavailable. These issues were addressed by assuming the available data as a representative for the missing data and was considered the average of all cities.

V. RESULTS

The extracted data was recoverable to 87% and the rest contained data schema which could not be used productively. Excess data of combination of products and some data which contained irrelevant information apart from product data was also extracted which consisted the majority of the unusable data.

I. RESULTS TABLE

Ecommerce name	Dataset description	
	Data collected	*Recoverable percentage*
Bigbasket	25 MB	75%
Natures basket	50 MB	97%
Reliencesmart	10 MB	95%
Webarchive	5 MB	85%

The challenges faced include dynamic structuring of web pages with separate representation of web elements due to constant updating of web page and structural changes. Use of dynamic web page content representation by online retailers where the displayed content is incomplete and only emulation of human behavioural pattern of slow scrolling and clicking on web elements results in the access of complete website data. The use of forms and interfaces also increased the complexity of accessing the website online prices data. Use of automated data filling and imitating web behaviour resulted in acquiring the entire web data.

The lack of availability of past data which is essential for forecasting inflation was dealt with by using web archive data. Web-archive website retains past versions of the web pages hosted online. These data were acquired by web scrapping and were used as past data representation. The frequency of retention by webarchive.org varied and hence extrapolation is used for prediction purposes. This data can be used for inflation forecasting purposes due to its time series representation and normalisation.

Conclusion

The web scrapping process involved dynamic website data extraction as well as static web data scrapping for archived data. The web spider for live websites needed to extract very specific data from the website via focused crawling as opposed to general downloading of the whole text content in simple crawling. This increased the complexity of the web scrapper module. Due to websites containing only live price the past data was collected via internet archive websites from static web pages. This had to handle the complexity of different layouts of the retailer websites across the years. The system contained web scrapper, data content analyser and extractor, data cleaner, data integration modules. The collected data was recoverable to 87% extent of the captured. This reduction included capturing of irrelevant data, partial representation and other discarded data as well.

The work so far uses a web crawler with certain manual inputs but in future use of Artificial Intelligence can enable either use of image processing or reinforcement learning to extract specific portion of the data from a dynamic website with the module adapting itself over any changes in the website. Other challenges faced include the incomplete website snapshots in archived data where the frequency of storing the past web pages were limited and the web site past versions were not containing all linked internal pages. This limited the amount of data extractable. The complexity of the websites also limited the amount data that could be extracted. Use of artificial intelligence to adapt to the websites changes is the proposed future work here.

Acknowledgment

The Author is indebted to Dr S. S. Maiti Director, and Mr Indranil Gayen, Research Officer, Big Data Analytics Division DSIM, RBI or their guidance and mentorship provided and to the Reserve Bank of India for DSIM internship scheme of mentoring student researchers.

References

1. Ahmed Toubar, “Image Recognition with Deep Learning for Web Scrapped Images,” Bachelor’s thesis.
2. Alberto Cavallo and Roberto Rigobon, “The Billion Prices Project: Using Online Prices for Measurement and Research,” In Journal of Economic Perspectives Vol. 30, No. 2 Springer pg. 151–178, 2016.
3. Alberto Cavallo and Manuel Bertolotto., “Filling the Gap in Argentina's Inflation Data,” In SSRN: https://ssrn.com/abstract=2782104 or http://dx.doi.org/10.2139/ssrn.2782104, 2016.
4. Arnaud Sahuguet and Fabien Azavant, “Building Intelligent Web Applications Using Lightweight Wrappers,” In Data and Knowledge Engineering, Volume 36, Issue 3, pg. 283-316, 2001.
5. Bing Liu and Lei Zhang, “A survey of opinion mining and sentiment analysis,” In Springer science DOI 10.1007/978-1-4614-3223-4_13, 2012.
6. Carlos Castillo, “Effective Web Crawling,” Doctoral Thesis, 2011.
7. Chen Hsinchuna, Chung Yi-Ming, Marshall Ramsey and Christopher C. Yang, “An intelligent personal spider (agent) for dynamic Internet/Intranet searching,” In Elsevier Science, 1998
8. Christopher H. Brooks, “Web Crawling as an AI Project, ” In Proc. of the 13th World Wide Web Conference, 2004.
9. D.F. Barrero, D. Camacho and M.D. R-Mormeno, “Automatic Web Data Extraction based on Genetic Algorithms and Regular Expressions,” In Springer DOI https://doi.org/10.1007/978-1-4419-0522-2_9
10. Dayne Freitag, “Machine Learning for Information Extraction in Informal Domains, ” In Machine Learning Journal, Vol. 39, pg.169–202, 2000.
11. Eloisa Vargiu and Mirko Urru, “Exploiting web scraping in a collaborative filtering based approach to web advertising,” In Artificial Intelligence Research, Vol. 2, No. 1, 2013.
12. Emilio Ferrara, Pasquale De Meo, Giacomo Fiumara and Robert Baumgartner, “Web Data Extraction, Applications and Techniques: A Survey,” International Journal of Engineering and Computer Science ISSN: 2319-7242, Vol. 2 Issue 4, 2014.
13. Felipe Jordao Almeida Prado Mattosinho, “Mining Product Opinions and Reviews on the Web,” Master’s thesis, 2011.
14. Frances Krsinich, “Price indexes from online data using the fixed-effects window-splice (FEWS) index,” In paper presented at the Ottawa Group, 2015.
15. G.Lazyan, T.Baghdasaryan and G.Aghajanyan, “The use of Big Data in Central Bank of Armenia,” In Proc. IFC-Bank Indonesia Satellite Seminar on “Big Data” at the ISI Regional Statistics Conference, 2017.
16. Gerd Stem, Andreas Hotho and Bettina Berendt, “Semantic Web Mining State of the Art and Future Direction,” In Journal of Web Semantics Vol. 4, No 2, 2006.
17. Gerson Francis DaCosta, Vijay Gaskadvi and Rahul Bhinde, “System for providing database functions for multiple internet sources,” In United States patent no. 6826553 B1, Nov 2004.
18. Harmandeep Kaur and Kamaljit Kaur Dhillon, “An Analysis of Naïve Bayes Hypothesis for Web Scraping and Data Mining,” In International Journal of Emerging Research in Management &Technology ISSN: 2278-9359 Vol. 6, Issue-7, 2017.
19. Harry T Yani Achsana and Wahyu Catur Wibowob, “A Fast Distributed Focused-Web Crawling,” In 24th DAAAM International Symposium on Intelligent Manufacturing and Automation, 2013.
20. Hsinchun Chen and Michael Chau, “Web Mining: Machine Learning for Web Applications,” In Annual Review of Information Science and Technology, 2004.
21. I Hull, M Löf, M Tibblin and S Riksbank, “Price information collected online and short-term inflation forecasts,” In Proc. IFC-Bank Indonesia Satellite Seminar on “Big Data” at the ISI Regional Statistics Conference2017
22. Ioan Dzitac and Ioana Moisil, “Advanced AI Techniques for Web Mining,” In Proc. MAMECTIS'08 10th WSEAS International conference on Mathematical methods, computational techniques and intelligent systems pg.343-346, 2008.
23. Jaideep Srivastava, Robert Cooley, Mukund Deshpande and Pang-Ning Tan, “Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data,” In ACM SIGKDD Explorations Newsletter Homepage archive Vol. 1 Issue 2, pg. 12-23, Jan 2000.
24. Jialun Qin, Yilu Zhou and Michael Chau, “Building Domain-Specific Web Collections for Scientific Digital Libraries: A Meta-Search Enhanced Focused Crawling Method,” In Proc. ACM / IEEE Joint Conference on Digital Libraries, pg. 135-141, 2004.

25. Jose´ Ignacio Fernandez-Villamor, Jacobo Blasco-Garcıa, Carlos Alglesias and Mercedes Garijo, “A semantic scraping model for web resources: Applying Linked Data to Web Page Screen Scraping,” In Proc. ICAART 2011 3rd International Conference on Agents and Artificial Intelligence pg. 451-456, Jan 2011.
26. Julian Seitner et.al., “A Large Database of Hypernomy Relations Extracted from the Web,” In Proc. LREC conference, 2016.
27. Junghoo Cho, Hector Garcia-Molina and Lawrence Page, “Efficient Crawling Through URL Ordering,” In Journal of Computer Networks and ISDN Systems Vol. 30 Issue 1-7 pg.161-172, 1998.
28. Jussi Myllymaki, “Effective Web Data Extraction with Standard XML Technologies,” In ACM 1-58113- 348, Jan 2005.
29. Liran Einav and Jonathan Levin, “The Data Revolution and Economic Analysis,” In Innovation Policy and the Economy, NBER, Vol. 14. University of Chicago Press, pg. 1-24, June 2014.
30. K. F. Bharati, P. Premchand and A. Govardhan, “Web Crawlers for Searching Hidden Pages: A Survey,” In International Journal of Computer Applications (0975 – 8887) Volume 64– No.14, February 2013.
31. Lu Jiang, Zhaohui Wu, Qian Feng, Jun Liu and Qinghua Zheng, “Efficient Deep Web Crawling Using Reinforcement Learning,” In: Zaki M.J., Yu J.X., Ravindran B., Pudi V. Advances in Knowledge Discovery and Data Mining. PAKDD, 2010. Lecture Notes in Computer Science, Vol. 6118. Springer, Berlin, Heidelberg.
32. Maciej Beresewicz, “On the Representativeness of Internet Data Sources for the Real Estate Market in Poland,” In Austrian Journal of Statistics, Vol. 44 and 45, April 2015.
33. Manvi, Ashutosh Dixit, Komal Kumar Bhatia and Jyoti Yadav, “Design and Implementation of Domain based Semantic Hidden Web Crawler,” International Journal of Innovations & Advancement in Computer Science Volume 4, Special Issue, May 2015.
34. Michael Chau, Jialun Qin, Yilu Zhou, Chunju Tseng and Hsinchun Chen, “SpidersRUs: Creating specialized search engines in multiple languages,” In Elsevier Publications Decision Support Systems Vol.45, pg. 621–640, 2008.
35. Michael Chau, Daniel Zeng, Hsinchun Chen, Michael Huang, David Hendriawan, “Design and evaluation of a multi-agent collaborative Web mining system,” In Elsevier Decision Support Systems Vol. 35, 2003.
36. Michael Chau and Hsinchun Chen, “A machine learning approach to web page filtering using content and structure analysis,” In Elsevier Science publications, 2007.
37. Mohammed Kayed and Chia-Hui Chang, “FiVaTech: Page-Level Web Data Extraction from Template Pages,” In IEEE transactions on knowledge and data engineering, Vol. 22, no. 2, February 2010.
38. Monica Peshave, “How search engines work and a web crawler application,” In Research-gate publication, 2018.
39. Nadzeya Kiyavitskaya, Nicola Zeni, Luisa Mich, James R. Cordy and John Mylopoulos, “Text mining through semi-automatic semantic annotation,” In Springer, Practical Aspects of Knowledge Management. PAKM, Lecture Notes in Computer Science, Vol 4333, 2006.
40. Natalie Glance, et.al. “Deriving Marketing Intelligence from Online Discussion,” In Proc. of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining pg. 419-428, 2005.
41. Nirali N. Madhak, Shahida G. Chauhan and Chintan R.Varnagar, “Understanding the Scope of Web Mining - Comprehensive Study,” In Proc. National conference on emerging trends in computer & electrical engineering (ETCEE), 2014.
42. P.H.Cording, “Algorithms for Web Scraping,” Master’s Thesis 2011.
43. Rashmi K. B, Vijaya Kumar T, H. S. Guruprasad, “Deep Web Crawler: Exploring and Re-ranking of Web Forms,” In International Journal of Computer Applications Vol. 150, 2016
44. Rayid Ghani, Rosie Jones, Dunja Mladeni, Kamal Nigam and Sean Slattery, “Data Mining on Symbolic Knowledge Extracted from the Web,” In Proc. Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2000
45. Ricardo Baeza Yates, Carlos Castillo, Mauricio Marin, Andrea Rodriguez, “Crawling a Country: Better Strategies than Breadth First for Web Page Ordering,” In Proc. 14th International World Wide Web Conference, WWW, 2005
46. S. P. Victor and M. Xavier Rex, “Analytical Implementation of Web Structure Mining Using Data Analysis in Educational Domain,” In International Journal of Applied Engineering Research ISSN 0973-4562 Vol. 11, No. 4, 2016
47. Salim Khalil and Mohamed Fakir, “RCrawler: An R package for parallel web crawling and scraping,” In Elsevier Science, SoftwareX Vol. 6, pg. 98-106, 2017.
48. Sergey Brin and Lawrence Page, “The Anatomy of a Large-Scale Hypertextual Web Search Engine,” In Elsevier Science, Computer Networks and ISDN Systems archive Vol. 30 Issue 1-7, pg. 107-117, April 1998.
49. Shaikh Phiroj Chhaware and Mohhamad Atique, “Web Data Extraction from Multi Data Region Deep Web Pages based on Visual Features,” In Proc. of the International Conference on Applied Mathematics and Theoretical Computer Science, 2013.
50. Sheilini Jindal and Gaurav Kumar, “A proportional analysis on the illustrious practices for the extraction and discovery of hidden patterns - data and web mining,” In International Journal of Enterprise Computing and Business Systems Vol. 1 Issue, 2011
51. S Raghavan, H G Molina, “Crawling the Hidden Web,” In Proc. of the 27th VLDB Conference, 2001
52. Soumen Chakrabarti, Martin van den Berg and Byron Dom, “Focused crawling: a new approach to topic-specific Web resource discovery,” In The International Journal of Computer and Telecommunications Networking archive Vol. 31 Issue 11-16, pg. 1623-1640, May 17, 1999.
53. Soumick Chatterjee and Asoke Nath, “Auto-Explore the Web – Web Crawler,” In International Journal of Innovative Research in Computer and Communication Engineering Vol. 5, Issue 4, 2017.
54. Suraj Gaikwad, Kunal Pokale and Anirban Dutta, “Implementation Paper on Visual Education using Data Mining and Innovative Visualization on Cloud,” In International Journal of Computer Applications (0975 – 8887) Vol. 116 – No. 5, April 2015.
55. T. V. Udapure, R. D. Kale, R. C. Dharmik, “Study of Web Crawler and its Different Types,” In Journal of Computer Engineering (IOSR-JCE) ISSN: 2278-8727, Vol. 16, Issue 1, 2014.
56. Trupti V. Udapure, Ravindra D. Kale and Rajesh C. Dharmik, “Study of Web Crawler and its Different Types,” In Journal of Computer Engineering (IOSR-JCE), e-ISSN: 2278-0661, p- ISSN: 2278-8727 Vol. 16, Issue 1, Ver.4 pg. 01-05, Feb. 2014.
57. Vasani Krunal A, “Content evocation using web scrapping and semantic illustration,” In Journal of Computer Engineering (IOSR-JCE) Vol. 16, Issue 3, 2014
58. Vinayak B. Kadam, Ganesh K. Pakle, “A Survey on HTML Structure Aware and Tree Based Web Data Scraping Technique,” In International Journal of Computer Science and Information Technologies, Vol. 5 Ver. 2, 2014.
59. Yolande Neil, “Web Scraping the Easy Way,” Master’s Thesis, 2016.