PIPECANDY WINTER INTERN

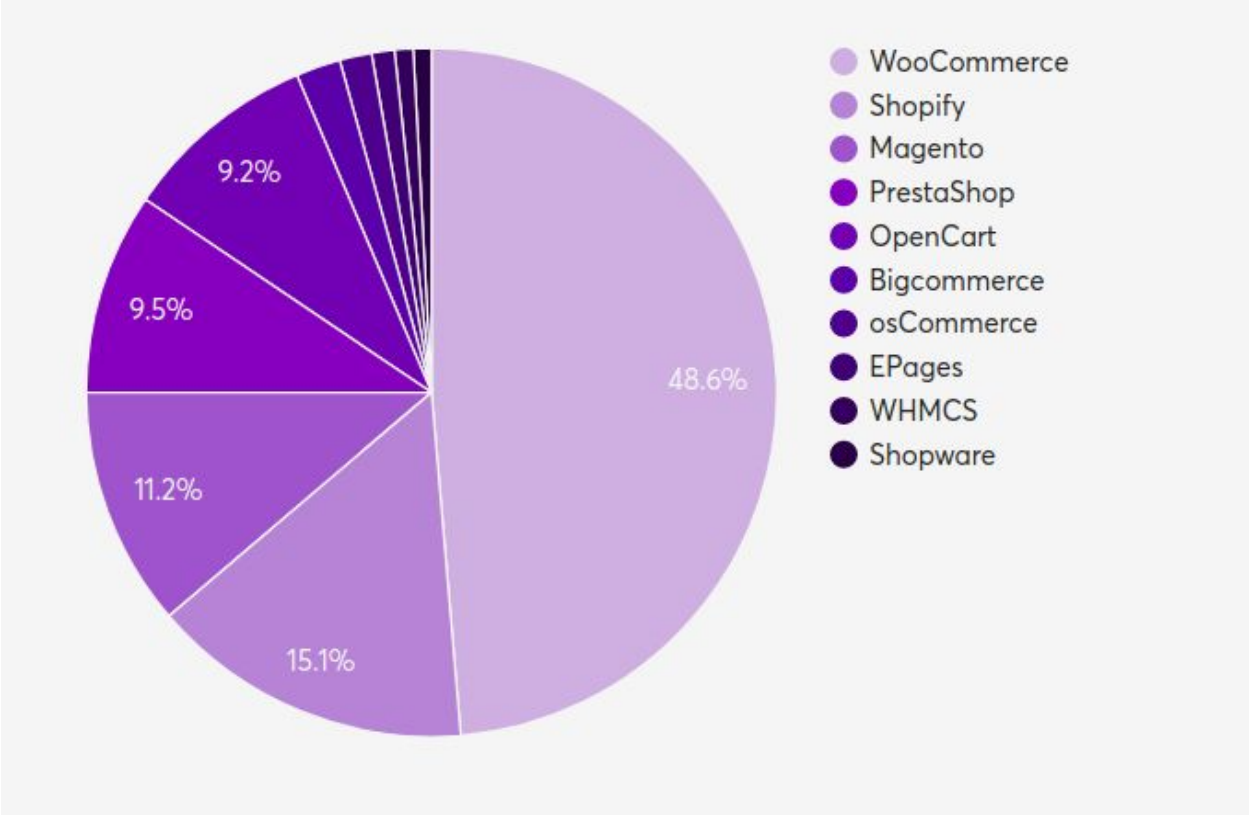
- **By Bhargav D**

AIM:

Given any E-Commerce website domain, get all the product urls of that domain, without actually crawling the entire website.

IDEA:

The first basic idea was to optimize the existing web crawler by having a bag of keywords which would tell us not to explore the URL further but it was later realized that it would be computationally expensive. So then I decided to see the the number of players in the E-Commerce field. By ‘players’ I mean the number of platform providers.



We soon realize that there are only 6 major players namely, WooCommerce, Shopify, Magento, PrestaShop, OpenCart and Bigcommerce. So if we find some kind of pattern or similarity for each of the platform to get all the product URLs we

would be able to cover about 95% of the E-Commerce Landscape which is a pretty good number.
So I came up with a set of Heuristic Methods for each platform which will let me easily get all the Product Urls of that domain at one place. I also have a set of last resort methods for platforms such as Magento where I couldn't find any common pattern.

1. **WooCommerce:-**

domain-name/shop or domain-name/products takes you to the product listing which contains all the products listed in pages.

2. **Shopify:-**

domain-name/shop or domain-name/products or domain-name/collections takes you to the product listing which contains all the products listed in pages.

3. **Bigcommerce:-**

domain-name/categories takes you to the product listing which contains all the products listed in pages.

4. **OpenCart:-**

For websites made with OpenCart they always have a search bar in their home page, so just by entering an empty space and searching will list all the products they have in a product listing page which contains all the products listed in pages. Sometimes this process can be made easier by trying out the following URL: ***domain-name/index.php?route=product/search&search=%20.*** If this URL exists it will directly take you to the required page of product listing if not then we have to enter the space in the search bar and search.

5. **PrestaShop:**

For websites made with PerstaShop for certain layouts which they provide entering an empty space in the search box will give you the desired result, even though a Generalized URL as in OpenCart can't be generated but this method gives you the desired result. And for some layouts in PrestaShop

domain-name/catalog or domain-name/catalog/product or domain-name/catalog/product-list will also lead us to the product listing page which contains all the product URLs. If these URL hits don't work then we can use the Last Resort Search queries to take us to the Product URLs.

6. Magento:

This platform proved to be the most tricky as I couldn't find a method in general at all, because they lacked a page which contains all their products but instead they had classified all their major product categories and had links for each major category in their home page. And entering each of those categories broad link will lead you to the product listing page which contains all the products in that broad category.

For example this domain (oliversweeney.com) contains the major categories, namely Men's, Women's and Accessories in the top of the page and choosing them would lead to the product listing page containing all the products in that major category. So getting all the product URLs from each of those product listing pages would lead to us getting all the product URLs from that website. I realize that this is a very difficult approach to automate but this is the only similarity between the web pages made with magento. For Magento specifically, I have given a special approach at the end of this document.

Last Resort Methods:

These methods are used in case any of the above methods don't work and we have no other option left. One method is to us the search box present in the E-Commerce website. Entering certain set of characters in the search box will let us get the search results which contains all the products in the website.

Some of these are:

- **?** (a question mark). This method works even in flipkart.com
- **Any random set of characters** which don't make any sense. This method works even in amazon.com, since it is not able to fund a product related to the entered text it will show all the products it has to offer.
- **-(random-text)**. A negative search also gives us the desired results in some websites.
- ***1==1*** (a true query) This method works for websites searching based on Query.
- ***** (asterisk symbol). This method works in certain Magento Layouts.

When we resort to using the Search Box to find out the product URLs, my suggestion is to try all of the above Search Queries and use the one which lists out the most Products and to ignore the ones which don't give any results. This is to avoid any Possible Overlaps.

For websites which haven't used any platform to build their E-Commerce websites are normally small-scale websites in which it would be good to enough to crawl the entire website since they don't have many products.

Another approach would be to use the site-map of the website to get the category urls and start crawling for product URLs from there instead of the home-page.

Implementation:

I have written a Python Script which will try out the different URL patterns with the given domain name and which based on the HTTP response we get for the URL combination will identify the platform that was used to build the E-Commerce website and return both the product listing page URL as well as the platform that was used to build the E-Commerce website. This script can be integrated into the crawler in such a way that the crawler knows which platform and from where to start crawling so that it can get all the product URLs.

For Magento: There were a few methods I tried out. The first one was to train a Machine Learning Model where I train the Model on the List of URLs where both Product URLs as well as Non-Product URLs and then it will be able to predict if a new URL is a Product one or non-Product one. I used various algorithms such as Logistic Regression, SVM, Naive Bayes, etc. Logistic Regression gave me the highest accuracy. But the problem with the model was you needed a Rich Data Set which contained varied type of Product URLs and Non-Product URLs for it to handle new URLs which it hadn't seen before. The other idea was to use the Metadata of a URL without actually scraping the entire Web-Page. And then use the Metadata content to predict if it's a Product URL or not. So basically it seemed like the initial idea I had come up with involving 'a Bag of Words' to predict if it was a Product URL or not seemed to be the best idea to go forward for Magento URLs, in fact it is the best way for any E-Commerce website which failed all the Last Resort Heuristics. This method is Computationally Expensive but it will be much faster than Scraping the entire website for Product URLs. I implemented the ML model in Python and also implemented my initial approach of using 'a bag of words' in a Python Script.

The approach of using 'a bag of words':- The basic idea of the web crawler is to go the home page of the e-commerce website and then start exploring the outgoing URLs. The idea is to intelligently weed out the useless outgoing paths.It was in general observed that the product landing pages always had the product name as a n-gram in their URL along with the Product ID or Item ID. The format of the item or the product ID is unique to a given E-Commerce Website but since we might be scanning hundreds of thousands of them, it is not feasible to optimize them based on the ID itself, although certain keywords such as ID or Product ID or Item are always present in them. It was also observed that the presence of a n-gram is very good indicator of a product page or leads to a product page eventually, like a product listing page.It was also observed that there are a list of keywords we don't want to be in the URL such as Contact Us, Sell, Shipping, Services, Stories and many more, they lead us to a page from which we are certain that it won't lead to a Product page. Also Social Media links are also not useful for us. And links whose domain name isn't same as the domain name which lead us to it can also be ignored, these maybe in cases where it leads us to the shipping company's website and crawling them won't be of any use to us. This even goes for Social Media Links. So filtering out the internal and external links at any stage of exploration should be our first step. It was also decided to check if the URL which is to be visited has already been visited or already in Queue to be visited, because it is obvious we never want to scrap the same web page twice and also if it's already in queue to be visited it doesn't make sense to add it again. I have attached the Flow Chart for the algorithm to be used in the absolute final case if none of the above heuristics work in the next page.

Flow Chart for the Algorithm to be used as a Last resort if none of the above mentioned Heuristics work: