**Drug Database Scraping for Facilitating Pharmaceutical Data Management**

**Abstract**

Python request and python-HTML requests are both very useful tools for scraping data off websites. These provide the backbone for a code developed to provide easy access to a wide variety of drugs with the simple input of what a researcher may want. The code has some set conditions specific to enhancing drug delivery through the buccal & sublingual systems, however it also can take input from the user making it a very malleable code.

**Introduction**

Effervescent reactions are seeing the light for more scientific investigation as they have been shown to provide an easier way for provide drug doses to patients as well as providing a faster uptake of drug absorption. They are usually made by combining an acid and base to produce a reaction releasing CO2 to help facilitate drug absorption ( Ipci et al. 2016). My thesis project aims to add a carrier molecule that is already binded to CO2 and help facilitate drugs absorption of drugs by having the CO2 directly push the drug through the buccal and sublingual epithelium directly into the bloodstream. Drugs that would benefit from this approach would be ones that would require faster uptake such pain killers or antihistamines.

**Methodology**

This code relies on several modules and imports those consist of

from pprint import pprint, from requests\_html import HTML, HTMLSession and import csv.

The functions that were made are called **drug\_list\_function(r): , get\_max\_pages(r): , get\_all\_pages(r):, get\_all\_drugs(list\_of\_urls): , safe\_drug(r): , drug\_name(r): , drug\_description(r): , is\_fever(r):, is\_painkiller(r): , is\_input(r,category): , is\_antihistamine(r): .** The output for the testing these functions can be found in Appendix C, Function Tests.

The overall way this program works is that you give an input of a web search query on DrugBank.ca and it will give you all the drugs that match the preset conditions from every single webpage relating to the initial search query. This is very useful as it can speed up looking through drugs that might help with your research, as actually using Drugbank.ca search query isn’t that great, as you can search up something like painkiller and there will be many things not related to painkiller. An example of the program can be found in Appendix B, this shows a snippet of what Python is doing to the search query. An example of an output for the program is in Appendix A.

**Getting All Possible Links**

To start of the program we must use this logic , convert a search query to its HTML code for example : session = HTMLSession() , r=session.get('https://www.drugbank.ca/unearth/q?utf8=%E2%9C%93&query=hydrophobic&searcher=drugs') , where r would be the input for many of these functions. .

1. Then we use the first function **drug\_list\_function(r): .** This function provides all the drugs that appear on the first page of the search query.
2. The second function we use is the **, get\_max\_pages(r):** This function takes in that search query and gives the final page number, aka the number of pages that we will have to search through this is actually based off the >> indicator in the HTML code , we use [-1] to denote the last page in the code though.
3. Next the **get\_all\_pages(r):**  function is used to get a list of all the urls in order with each of their associated pages. This requires the **get\_max\_pages(r):** it looks for **{}** in a an initial search query such as [https://www.drugbank.ca/unearth/q?c=\_score&d=down&page={}&query=vitamin&searcher=drugs](https://www.drugbank.ca/unearth/q?c=_score&d=down&page=%7b%7d&query=vitamin&searcher=drugs) , and it gives an output of a list of the all the urls at the end .
4. Next the **get\_all\_drugs(list\_of\_urls)** function is used and this takes the input of the **get\_all\_pages(r):**  function in the big search query, or just a list of urls you are interested in looking at. It uses a for loop to loop through every single drug on every page in all of the pages in the search query. Example looking up vitamins gives , 25 drugs per page for 6 pages with 14 drugs on the last page, so that is 139 links this will open up and check.

**Deciding What to Keep**

This is the section of the code where we start to actually analyze the list of urls give to us, these function work at accepting individuals however in the last in the last instance of the program we use a loop to compile a tuple which we apply all our analysis functions to. This section of code is very malleable as it provides the location in the HTML where the conditions you are looking for can bet set up in a Boolean and if found at that particular HTML section example pain = r.html.find('.list-unstyled' +'.table-list') , where the '.list-unstyled' +'.table-list'  **class** is indicating where that condition can be found and something like in the is\_painkiller function we see pain[0].text.find("Painkiller") >=0 , indicating that we want to find Painkiller at its associated place [0] in the list of HTMLS that have that associated class. If the drug meets some of the desired requirements we will printing that information to a CSV file.

1. The first function that limits what we print to the CSV file is **safe\_drug(r): ,**  this checks if the drug is safe or not by looking at the ( '.col-md-10' +'.col-sm-8') class in its fourth position **[3].** This is done by checking the word ‘Withdrawn’ or ‘withdrawn’ appears there as this indicates that the drug is actually dangerous for people to use and shouldn’t be given. However this search is still relative as things like Cocaine will be present, the withdrawn factor means that doctors will not describe it. In my case this is important as I would want to find pharmaceutical drugs/ nutraceuticals that would be prescribed to people so that way I can identify if they would work with my system and provide potential benefit to these patients.
2. **The drug\_name(r)**  function is useful for providing the name of the drug from the given url. In the long list it will only print out if the drug meets all the requirements that input into the program, however you can input any url and it will give the drugs name. The **drug\_description(r):**  also operates on this logic, as we would not want to have vast amounts of drug description to sieve through we also only print it if the drug meets the required criteria.
3. **,** The functions **is\_fever(r):, is\_painkiller(r): , is\_input(r,category): , is\_antihistamine(r): ,**  all operate on the same logic. They look at the position where the html code for the conditions or associated conditions is and tries to find the strins associated to them if they do then the Boolean criteria is met and it comes back as true and is added into the list. These are the final things to check if they add intot the CSV output. An example of a of this working would be for the painkiller function . pain[0].text.find("Painkiller") >=0 or pain[0].text.find("Pain") >=0 or pain[0].text.find('Analgesic') . This here is looking at the conditions at ('.list-unstyled' +'.table-list') [0] to see if the drug is associated with ‘Painkiller’ , ‘Pain’, ‘Analgesic’ .

**Saving All the Possible Drugs to a CSV**

The overall purpose of the program is to take all the drugs that match the given criteria and put them into a CSV file that has the headers of drug name, url and description. This is done by opening up a csv file and feeding it the output of the functions such as  **my\_data = (drug\_name(r), url, drug\_description(r)) .** It also takes into account any checks that were provided to it such as **keep\_condition = is\_fever(r) or is\_painkiller(r) or is\_antihistamine(r) or is\_input(r,'Corticosteroids') .** The rest of the code can be found in in Appendix D.

**Results**

The results for the main programs function were based off using a search query from the main website of Drug Bank.ca , which would look like <https://www.drugbank.ca/unearth/q?utf8=%E2%9C%93&searcher=drugs&query=pain+killer>, as pain killer was the key term searched in the drug base home page. The search query for Painkillers had **also is\_fever**, **is\_ Antihistamine**, **is\_input(r,'Corticosteroids')**

Search Query in Python

False

False

True

('Acetaminophen', 'https://www.drugbank.ca/drugs/DB00316', 'Acetaminophen (paracetamol), also commonly known as Tylenol, is the most commonly taken analgesic worldwide and is recommended as first-line therapy in pain conditions by the World Health Organization (WHO).10 It is also used for its antipyretic effects, helping to reduce fever.23 …

True

('Betamethasone', 'https://www.drugbank.ca/drugs/DB00443', 'Betamethasone is a long-acting corticosteroid with immunosuppressive and antiinflammatory properties.11 It can be used topically to manage inflammatory skin conditions mineralocorticoid activity.11' ….)

False

False

False

The other functions were tested with a drug that met the given requirement and another drug that did not meet the given requirement. In Appendix C Function Tests, Figure 1 . the function **is\_painkiller**  is tested with an input of the link to Advil and another input to the link for Vitamin C. This was done for every requirement as well as the withdrawn condition. Looking through the rest of Appendix C we can see that the all of the functions work by taking in the search query and giving the desired output.

**Discussion**

The overall purpose of this function is take in the URL from search query from a DrugBank.ca and output all the drugs that meet the requirements that you have fed it. The program provided comes with hard coded restrictions but they do not have to be included in the keep conditions. This provides an easy way to get all the drugs the urls and the description of those drugs that you are interested in looking up. This can be useful because looking at multiple drug websites it seems that their rigidity of their search queries leave something to be desired as they have trouble searching multiple restrictions, DrugBank.ca being one of them.

From looking at the results their seems to be a huge issue with returning a lot of false negatives, as it doesn’t make sense that when looking up painkillers on DrugBank.ca that my search program only yields ~10 ish drugs. I think one of the big issues with the website is that sometimes the drug Categories or Associated Conditions html code is not their or in different spots. I had tried to use an Xpath approach in **is\_fever** instead of looking at the HTML specific case but it didn’t really solve the issue. If possible, I would like some feedback or any ideas of why my code may be returning so many false negatives.

My work in the future would include many more restrictions, such as using pka, drug size and drug structure, if it’s a prodrug, bioavailability … however this was not really possible at the time due to time restrictions and the fact that the code returns so many false negatives.

My other main challenge with this code was having to learn so many new techniques to actually get something to work. In the references I show all the YouTube tutorials where I had to learn how to use the python modules. I also had to learn HTML, XPATH, Selenium, SQLIte (didn’t use in the end). All of the code was inspired by the logic used in the python modules from requests and the way HTML and XPaths behave, as well as the knowledge that was taught in the class such as using read, write, for loops, strings, functions...

**Special Considerations with the Functions**

The fever function works a bit differently as I did encounter some issues with my code returning a lot of false negatives, this was attempted be overcame with the brute force approach of using the xpath command **fever=r.html.xpath('//ul[@class="list-unstyled table-list"]/../../\*[contains(text(), "Associated Conditions")]/following::dd[1]', first=True) .** This approach was not ideal though as it is much more complicated and did not solve the problem of many false negatives.

The **is\_input(r,category):** function is probably the most useful for anyone who wants to use this code as this one allows for user input to check the long list of URLs. The reason why the other conditions are hard coded in is because those are more specific to the search for drugs/ nutraceuticals that would benfit the enhanced delivery through the buccal/ sublingual system.

The final comment on the full program is that sometimes it returns false positives that don’t really belong / shouldn’t be there this could be partially from DrugBank.ca and partially from my code not functioning completely. One example of this is when looking through my search query of Pain killers , Adenosine hits True and sent to my CSV. The problem with this is that Adenosine is nucleotide so obviously it would be in many search queries , the problem is that my search query is through the drug tab, and yet Adenosine makes up the drugs but isn’t itself a drug.

Overall it would seem that even though this code is has some logical errors in it still can be useful for providing the framework for other pharmaceutical data scraping programs. As well it did provide me with a nutraceutical that I will be doing more research into for my project which is Evening Primose Oil, a nutraceutical that may provide relief to people suffering from systemic inflammatory disorder it meets my hydrophobic requirement and the pain killer requirement among others. Finally, flavour compounds were not investigated as all the HTML that I had examined were horrible to work with for those websites and would require a lot time.

Disclaimer web scraping can be illegal on some websites and John Alexandre Weilenmann is no way responsible of endorsing the use of web scraping for committing such acts, the purpose this webs scraper is to facilitate the act of acquiring pertinent pharmaceutical information to aid in the progress of Science

**References**

Database Reference

<https://www.drugbank.ca/>

Knowledge References

# Python Tutorial: Web Scraping with Requests-HTML

<https://www.youtube.com/watch?v=a6fIbtFB46g>

# Python SQLite Tutorial: Complete Overview - Creating a Database, Table, and Running Queries

<https://www.youtube.com/watch?v=pd-0G0MigUA>

# XPath Basics Tutorial HD <https://www.youtube.com/watch?v=r_AP1I3T9yM>

# 10 Python Tips and Tricks For Writing Better Code

<https://www.youtube.com/watch?v=C-gEQdGVXbk>

# HTML Full Course - Build a Website Tutorial

<https://www.youtube.com/watch?v=pQN-pnXPaVg>

# Python Requests Tutorial: Request Web Pages, Download Images, POST Data, Read JSON, and Mor

<https://www.youtube.com/watch?v=tb8gHvYlCFs>

Python Documentation and Knowledge

<https://requests.readthedocs.io/projects/requests-html/en/latest/>

<https://requests.readthedocs.io/en/master/>

<https://docs.python.org/3/library/csv.html>

<https://www.journaldev.com/23666/python-string-find#:~:text=This%20function%20returns%20the%20lowest,found%20then%20%2D1%20is%20returned.>

<https://www.selenium.dev/selenium-ide/>

Scientific References

­­­­­­Ipci, Kagan & Öktemer, Tuğba & Birdane, Leman & Altıntoprak, Niyazi & Bayar Muluk, Nuray & Passali, Desiderio & Lopatin, Andrey & Bellussi, Luisa & Mladina, Ranko & Pawankar, Ruby & Cingi, Cemal. (2016). Effervescent tablets: a safe and practical delivery system for drug administration. ENT Updates. 10.2399/jmu.2016001009.

**Appendix**

**Appendix A**

CSV Output for search Query of Painkiller , with Corticosteroid as Input

my\_search= session.get('https://www.drugbank.ca/unearth/q?utf8=%E2%9C%93&searcher=drugs&query=pain+killer')

|  |  |  |
| --- | --- | --- |
| drug name | url | drug description |
|  |  |  |
| Choline | https://www.drugbank.ca/drugs/DB00122 | A basic constituent of lecithin that is found in many plants and animal organs. It is important as a precursor of acetylcholine, as a methyl donor in various metabolic processes, and in lipid metabolism. |
|  |  |  |
| Menthol | https://www.drugbank.ca/drugs/DB00825 | Menthol is a covalent organic compound made synthetically or obtained from peppermint or other mint oils. Forming clear or white waxy, crystalline substance, menthol is typically solid at room temperature. (-)-Menthol is the naturally-occurring and main form of menthol, and is assigned the (1R,2S,5R) configuration. Menthol mediates anesthetic properties and anti-irritating properties locally, thus it is widely used to relieve minor throat irritations. |
|  |  |  |
| Salicylic acid | https://www.drugbank.ca/drugs/DB00936 | A compound obtained from the bark of the white willow and wintergreen leaves, and also prepared synthetically. It has bacteriostatic, fungicidal, and keratolytic actions. Its salts, the salicylates, are used as analgesics. |
|  |  |  |
| Adenosine | https://www.drugbank.ca/drugs/DB00640 | A nucleoside that is composed of adenine and d-ribose. Adenosine or adenosine derivatives play many important biological roles in addition to being components of DNA and RNA. Adenosine itself is a neurotransmitter. |
|  |  |  |
| Chlorpheniramine | https://www.drugbank.ca/drugs/DB01114 | A histamine H1 antagonist used in allergic reactions, hay fever, rhinitis, urticaria, and asthma. It has also been used in veterinary applications. One of the most widely used of the classical antihistaminics, it generally causes less drowsiness and sedation than promethazine. |
|  |  |  |
| Doxylamine | https://www.drugbank.ca/drugs/DB00366 | Histamine H1 antagonist with pronounced sedative properties. It is used in allergies and as an antitussive, antiemetic, and hypnotic. Doxylamine has also been administered in veterinary applications and was formerly used in parkinsonism. |
|  |  |  |
| Alitretinoin | https://www.drugbank.ca/drugs/DB00523 | An important regulator of gene expression during growth and development, and in neoplasms. Tretinoin, also known as retinoic acid and derived from maternal vitamin A, is essential for normal growth; and embryonic development. An excess of tretinoin can be teratogenic. It is used in the treatment of psoriasis; acne vulgaris; and several other skin diseases. It has also been approved for use in promyelocytic leukemia (leukemia, promyelocytic, acute). |
|  |  |  |
| Acetaminophen | https://www.drugbank.ca/drugs/DB00316 | Acetaminophen (paracetamol), also commonly known as Tylenol, is the most commonly taken analgesic worldwide and is recommended as first-line therapy in pain conditions by the World Health Organization (WHO).10 It is also used for its antipyretic effects, helping to reduce fever.23 This drug was initially approved by the U.S. FDA in 1951 and is available in a variety of forms including syrup form, regular tablets, effervescent tablets, injection, suppository, and other forms.15,16,23,Label  Acetaminophen is often found combined with other drugs in more than 600 over the counter (OTC) allergy medications, cold medications, sleep medications, pain relievers, and other products.19 Confusion about dosing of this drug may be caused by the availability of different formulas, strengths, and dosage instructions for children of different ages.19 Due to the possibility of fatal overdose and liver failure associated with the incorrect use of acetaminophen, it is important to follow current and available national and manufacturer dosing guidelines while this drug is taken or prescribed.20,21,Label |
|  |  |  |
| Betamethasone | https://www.drugbank.ca/drugs/DB00443 | Betamethasone is a long-acting corticosteroid with immunosuppressive and antiinflammatory properties.11 It can be used topically to manage inflammatory skin conditions such as eczema, and parenterally to manage several disease states including autoimmune disorders.11 Betamethasone has potent glucocorticoid activity and negligible mineralocorticoid activity.11 |
|  |  |  |
| Evening primrose oil | https://www.drugbank.ca/drugs/DB11358 | Evening primrose oil comes from the extraction from Oenothera biennis seeds and it is commonly used as an alternative source for omega-6 essential fatty acids. In its composition it presents some fatty acids such as Linolenic acid and Gamolenic acid.1 Evening primrose oil has been filled for the FDA by Humanetics Corporation on April 2000 to be a new dietary ingredient but its current status is "Inadequate basis for expectation of safety".5 By Health Canada, evening primrose oil is approved in over-the-counter combination dietary supplements.6 By the EMA, evening primrose oil is approved in herbal preparations.7 |

**Appendix B**

Python Print of Search Query Snippet

my\_search= session.get('https://www.drugbank.ca/unearth/q?utf8=%E2%9C%93&searcher=drugs&query=pain+killer')

False

False

False

False

True

('Acetaminophen', 'https://www.drugbank.ca/drugs/DB00316', 'Acetaminophen (paracetamol), also commonly known as Tylenol, is the most commonly taken analgesic worldwide and is recommended as first-line therapy in pain conditions by the World Health Organization (WHO).10 It is also used for its antipyretic effects, helping to reduce fever.23 This drug was initially approved by the U.S. FDA in 1951 and is available in a variety of forms including syrup form, regular tablets, effervescent tablets, injection, suppository, and other forms.15,16,23,Label\nAcetaminophen is often found combined with other drugs in more than 600 over the counter (OTC) allergy medications, cold medications, sleep medications, pain relievers, and other products.19 Confusion about dosing of this drug may be caused by the availability of different formulas, strengths, and dosage instructions for children of different ages.19 Due to the possibility of fatal overdose and liver failure associated with the incorrect use of acetaminophen, it is important to follow current and available national and manufacturer dosing guidelines while this drug is taken or prescribed.20,21,Label')

True

('Betamethasone', 'https://www.drugbank.ca/drugs/DB00443', 'Betamethasone is a long-acting corticosteroid with immunosuppressive and antiinflammatory properties.11 It can be used topically to manage inflammatory skin conditions such as eczema, and parenterally to manage several disease states including autoimmune disorders.11 Betamethasone has potent glucocorticoid activity and negligible mineralocorticoid activity.11')

False

False

False

False

**Appendix C**

Function Tests

Figure 1: Testing Painkiller Function

# test painkiller

advil = session.get('https://www.drugbank.ca/drugs/DB00316')

if is\_painkiller(advil)==True:

print('helps with pain')

else:

print('doesnt help with pain')

ascorbic\_acid = session.get('https://www.drugbank.ca/drugs/DB00126')

if is\_painkiller(ascorbic\_acid)==True:

print('helps with pain')

else:

print('doesnt help with pain')

**Prints out**

helps with pain

doesnt help with pain

Figure 2: Drug Safety Function

# testing if drug is safe or not

Grepafloxacin= session.get('https://www.drugbank.ca/drugs/DB00365') #i know it is not safe

if safe\_drug(Grepafloxacin)==True:

print('Safe')

else:

print('Not Safe')

**Prints out**

Not Safe

Figure 3: My input / My medical condition Function

#test for my input / my medical condition

hist = session.get('https://www.drugbank.ca/drugs/DB00920')

my\_problem= 'Antihistamine'

if is\_input(hist,my\_problem)==True:

print('helps with that problem')

else:

print('doesnt help with that problem')

**Prints out**

helps with that problem

(The input drug was an antihistamine)

Figure 4: Using the get\_all\_drugs function

'https://www.drugbank.ca/drugs/DB01125',

'https://www.drugbank.ca/drugs/DB13257',

'https://www.drugbank.ca/drugs/DB01592',

'https://www.drugbank.ca/drugs/DB13152',

'https://www.drugbank.ca/drugs/DB00630',

'https://www.drugbank.ca/drugs/DB11126',

'https://www.drugbank.ca/drugs/DB11112',

'https://www.drugbank.ca/drugs/DB00100',

'https://www.drugbank.ca/drugs/DB14496',

'https://www.drugbank.ca/drugs/DB11136',\..............

Figure 5: Code Snippets

#looking up painkiller

my\_search= session.get('https://www.drugbank.ca/unearth/q?utf8=%E2%9C%93&searcher=drugs&query=pain+killer')

max\_pages = get\_max\_pages(my\_search)

print(max\_pages) #gives 2 which is correct

all\_the\_pages = get\_all\_pages(my\_search)

print(all\_the\_pages)

#prints out so works

#['https://www.drugbank.ca/unearth/q?c=\_score&d=down&page=1&query=vitamin&searcher=drugs', 'https://www.drugbank.ca/unearth/q?c=\_score&d=down&page=2&query=vitamin&searcher=drugs']

drugs\_urls = get\_all\_drugs(all\_the\_pages) #this is what I use to fill my csv below

pprint(drugs\_urls)

#prints out all of the nice drugs that are in sech query

# ['https://www.drugbank.ca/drugs/DB00173',

# 'https://www.drugbank.ca/drugs/DB14484',

**Appendix D**

Saving All Drugs That Meet Criteria to CSV

Figure 1: CSV File Saver

csv\_file = open('drug\_scrape2.csv', 'w')

csv\_writer = csv.writer(csv\_file) #writing to the csv file

csv\_writer.writerow(['drug name','url', 'drug description'])

my\_output\_list = []

for url in drugs\_urls: #iterating through all the urls

r = session.get(url)

if not safe\_drug(r): #if the drug is not safe then just contineu on and dont add it to csv

continue

keep\_condition = is\_fever(r) or is\_painkiller(r) or is\_antihistamine(r) or is\_input(r,'Corticosteroids') #these are the keep conditions can be anything though

print(keep\_condition)

if keep\_condition:

my\_data = (drug\_name(r), url, drug\_description(r)) #if one of the keep conditions is met run all the functions on that url

print(my\_data)

csv\_writer.writerow(my\_data)

csv\_file.close()