Kelli Trei

ktrei2@illinois.edu

IS 452, A FA 19:-Elizabeth Wickes

**Final Project**

I am the Biosciences Librarian and University representative to the [Biodiversity Heritage Library](http://www.biodiversitylibrary.org) (BHL). The BHL is an open access digital library of biodiversity literature; particularly rare materials published pre-1923. It allows international unfettered access to scans of this hard to find literature. Recently the BHL provided and OCR on all of its allowing for full text searching. This also allows for other possibilities. A lot of this literature, primarily book or journal bound “items”; contain what are known as “Identification Keys”. These keys are similar to what are found in field guides, they are basically rubrics for identifying plants and animal species based on physical characteristics. They tend to be presented in text as indented numbered tables detailing these characteristics. I aimed to create code to parse out these keys from the surrounding text. I have tested a program I created using Python that can be applied to the BHL’s OCR.

I completed the workflow and it helped me frame my plan. First, I created a GitHub repository. It is located here: <https://github.com/ktrei2/IS452AFA19>. I downloaded the OCR of five monographs from the BHL library. These were monographs I know contain various kinds of scientific identifications keys. I tried to get a mix of key types, mushrooms, fish, marine species, etc. I created a text file compiling all five monograph OCR in one, I named this file bhlfivevolumes.txt, and it is in the GitHub repository. Then I acquired an external hard drive and the entirety of the BHL OCR. This file was given to me as a compressed tar file. I learned that my work computer does not have the power to decompress a file of this size. I enlisted the help of a coworker, and now, seven days in, we are still waiting for the decompression to finish, and it was last at 57 million files. Thus, I decided to focus on my sample file. However, I was able to secure a sample of the output of this decompression. This file is called bhl1.zip in the GitHub folder. I attempted to write a code to “find” the beginning of keys using terminology or other identifying features. I plan to use some code from the midterm project to assist me in pulling out sections or chapters of the works and also splitting on specific terms if possible. I hoped to figure out how to use proximity searching as part of the code as well since these keys are not consistently titled.

I actually went some different routes. Having read a bit on regular expressions, I went down a wormhole that week of class on the internet. I found some expressions that seemed like they would be useful. I knew that the word “key” would be all over the place in the text. It is part of compound words (keyhole) it lower case can be used to mean different things than I want (“the key point”). And I felt it would be too many false hits. So I decided to try the re.match(). I tried search, findall, and finditer as well, but match seemed most efficient since I wanted titles of these keys, not every time someone mentioned a key. Another thing I discovered was retaining capitalization of Key helped a great deal as these were titles, and titles are generally capitalized. Sometimes the keys were also titled entirely in caps, so I had to adjust the code to find both Key and KEY (as well as pluralized). Finding the keys is where most of my problems occur. These monographs have extensive citations, many times the title of keys is in a citation in its entirety. Additionally, in these monograph examples, some are collections of keys, and these monographs contain a table of contents which also has the title of keys in the text. From the perspective of a finding aid, I think this could still be useful because it could help the user identify texts with a key, but it might not take you directly to that key. Regardless, I realized I needed a location, I needed information on where in the text these results were coming from, not just the output. Back to regular expressions I used re.group(). What I found is that while I could find the character location in a given line of the word key while using .match, I could not find the line itself. However, I could with finditer. In all honesty, I am still not sure I understand entirely why this is so. I believe it is because finditer is searching through the entire string as opposed to the beginning, but I am a bit beyond my understanding. So I ultimately wrote code both ways, .match and .finditer, but chose to comment out .match, because even with erroneous key results, I was able to find the specific line location in the text file.

Now, this was a file I created, a long text file but a teeny tiny sample of the BHL OCR as a whole. I need code to instruct my program to read through all the files within a file. The output of the decompression is hundreds of text files for each item in the BHL. My small example of bhl1.zip demonstrates this. First I wanted to code the program to open the zip file within pycharm. Tis way I could bring the file in and then have my folder structure ready and integrated. I used this with the module Zipfile. However, then I hit a wall. I cannot figure out how to loop over the text files within a file. I could certainly code each file separately but that is insane with millions of files. I believe os or glob might be the modules I need. However, the code that I could find examples of seemed very confusing to me and I was not even sure where to begin.

This issue proved to be just the begging of many questions. It occurred to me that the BHL contains materials in many different languages. While English is the major language, there are many journal series in French, German, Russian, etc. I will need to ascertain the words these works use to “name” their keys. Many keys use a series of ellipsis to denote the beginning of a list. Unfortunately, the BHL OCR has pulled out the majority of punctuation. I wonder if the API would include punctuation? I admit I do not know if that makes sense, but might be worth a try if it does. This project has taught me that I can write simple code, but that I get very overwhelmed once things become complex. I have certainly learned how to speak more clearly as to what I want a program to do. I feel I understand more what a program can actually do. However, there are still many black boxes in the process that I do not understand. I hope to pursue investigating this BHL project, most likely by speaking to someone with more prowess than myself. I feel that I can go seek the information in a cohesive manner.