

# Kestrel Taxonomy Finder

Version 0.5

5/01/2018

Copyright 2017 by Shawn Rupp

Shawn.M.Rupp@asu.edu

## Contents

|                               |   |
|-------------------------------|---|
| Introduction.....             | 2 |
| Installation.....             | 2 |
| Getting Started.....          | 3 |
| Running Kestrel.....          | 4 |
| Extracting Search Terms.....  | 4 |
| Searching for Taxonomies..... | 5 |
| Scripts.....                  | 6 |

## New in v0.5

Gets genus from Wikipedia if it includes a citation (as of 0.4)

Scores pairs of results to find best match before counting number of NAs

## Introduction

Kestrel is a program for resolving species' common names and synonyms with “official” scientific names and extracting taxonomies from internet databases (i.e. [EOL](#), [NCBI](#), [IUCN](#), [ITIS](#), [Wikipedia](#)). It is meant to automate this process as much as possible, although, depending on your input data, some manual curation may still be required.

### *Dependencies:*

Python3  
Cython  
NLTK  
BeautifulSoup4  
Selenium  
Chrome or Firefox  
Active internet connection

Note: Kestrel is written and tested on a Linux platform, so there is no guarantee that it will work on Mac or Windows without troubleshooting. Additionally, the following instructions are written for Linux and may require some modification for different operating systems. (I would love to provide further support for this program, but unfortunately time and funding constraints dictate otherwise.)

## Installation

### *Cython*

Kestrel utilizes Cython to compile python code into C and improve performance. Cython can be installed from the [pypi](#) repository or via [Miniconda](#) (it is installed by default with the full Anaconda package).

To install with Miniconda:

```
conda install cython
```

### *NLTK*

Kestrel uses python's [Natural Language Processing Toolkit](#) to differentiate between common and scientific names in its input. To install on any Debian-based Linux platform, enter either of the following into a terminal:

```
conda install nltk
```

or

```
sudo pip install -U nltk
```

Kestrel comes with it's own training dataset, so there is no need to download any additional data from NLTK.

### *BeautifulSoup4*

Kestrel also uses the [BeautifulSoup](#) module, and the lxml parser, to parse html pages.

```
apt-get install python3-bs4
apt-get install python-lxml
```

### *Selenium*

Since few grant-funded institutions can afford to pay for a custom Google search, Kestrel uses [Selenium](#) to run automated Google searches. To install the Python repository, enter the following into a terminal:

```
pip install -U selenium
```

Selenium requires browser-specific drivers. Kestrel supports [Chrome](#) and [Firefox](#) since they are both platform independent. To install, click the link for the browser you want to use (the browser must also be installed on your computer), download the correct version for your system, unzip, and move it to /usr/bin (or another location in your PATH).

### *Kestrel*

Download the git repository, change into the directory, and build the Cython scripts.

```
git clone https://github.com/icwells/Kestrel.git
cd Kestrel/
./install.sh
```

## **Getting Started**

### *EOL API Key*

Kestrel queries the [Encyclopedia of Life](#) heavily, so you will need to generate an api key. To do so, create an EOL profile if you do not already have one. Sign in, go to your profile, and click “edit my profile.” Next, generate an API key in the box on the right side. Once you generate the API key, copy it and paste it into example-API.txt on the same line as “EOL=”. Finally, change the file name to API.txt (otherwise it will be erased if you run “git pull”).

### *NCBI API Key*

As of 2018, NCBI requires an API key for any extensive searching. Create an [NCBI account](#), or sign in, and click on your user name in the upper right hand corner of the screen. This will take you to your settings page. The fourth box down will allow you to generate an API key. Copy this key and paste it after “NCBI=” in API.txt.

### *IUCN API Token*

Starting with version 0.3, Kestrel also optionally queries the [IUCN Red List](#) database. If you wish to request a token, fill out [this form](#) and submit it. Once you receive your API token, paste it after “IUCN=” in API.txt. It can take several days for approval, so Kestrel can run without this API key. Fortunately, IUCN results can still be found via Kestrel’s Google search.

## Running Kestrel

Assuming you have a functional internet connection, you are ready to run the program. As of v0.2, Kestrel must be run in two separate steps. The first step involves extracting target terms from an input file, filtering out terms which are unlikely to produce a result, and formatting the terms. These terms are then used as input in the second stage to search for taxonomies from NCBI, EOL, [GBIF](#), and/or Wikipedia and selects the best match.

For both steps, Kestrel starts by reading names from any existing output files before reading names from the input file. If the given output file already exists, it will append output to this file which allows you to resume if the program is interrupted. (Since the extraction step is much faster, this feature is more relevant for searching.)

To run Kestrel, you must open a terminal and change into `~/Kestrel/bin`. This directory contains the API keys and reference common and scientific names for NLTK.

### *Extracting Search Terms*

The first step is to extract, filter, and format the query terms. To extract terms, change into `Kestrel/bin/` and enter:

```
python kestrel.py --extract {--common/scientific}  
-c <column number> -i <input file> -o <output file>
```

The input file should be either a csv, tsv, or txt (either single column or tab delimited) file containing the scientific or common names of species of interest. The `-c` flag signifies the column number where the names are located. This is a 0-based integer, so if the names are in the first column specify “`-c 0`”. The program will extract unique names from this column. It will only search for a specific name once to save time and avoid burdening servers. There is no default value since it is very easy to forget this parameter and it is better to run into an error up-front.

If the “`--common`” or “`--scientific`” flags are given, Kestrel will assume every input name is of the given type. If the name type is not specified, Kestrel will generate a feature classifier with NLTK to determine whether an input name is a common name or a scientific name. Kestrel then filters and formats the terms based on their type.

For both types, commas are replaced with spaces and any name which contains too many numbers (more than  $\frac{1}{4}$  of the total number of characters) will be rejected. Scientific names will be rejected if they contain any punctuation other than a period (or commas since they have already been removed). Common names will be rejected if there is an indication of uncertainty (i.e. a question mark, the words “not” or “unknown,” ...). If any other punctuation is detected, Kestrel will attempt to format the name so it has the best chance of returning a match (e.g. “&” are replaced with “and,” items in parentheses or quotation marks are removed, ...). Lastly, spaces and apostrophes will be percent encoded immediately prior to searching (in the next step) to keep the results of this step readable.

Formatted search terms will be written to the given output file along with the unformatted name and its type. Any rejected names will be written to “`KestrelRejected.csv`” with the reason it was rejected. Since there is no easy way to predict exactly what might be in the input file, terms in the output file may be

examined and edited if needed. Rejected terms may also be manually formatted and pasted into the output file. Depending on the quality of the input, this could greatly improve the quality of the search output and will be less time consuming than editing the original input file.

## ***Searching for Taxonomies***

### ***API-based Search***

To search for matches to the formatted search terms (whether or not they were manually checked), Kestrel will read the extracted names and formatted search terms from the input file and will search for matches with one thread per search term. If multiple input queries have the same formatted search term (i.e. “Dog,” “dog,” and “DOG” will all be formatted to “Dog”), the term is only searched once but output will be written for each input query.

```
python kestrel.py [--firefox] -t <threads>
-i <input file> -o <output file>
```

For common names, it will search EOL, NCBI, and Wikipedia. If two or more databases return a matching taxonomy, it will be written to the output file. If there is not a majority match and the term has more than one word, Kestrel will remove the first word (the last word of a common name is usually the noun) and repeat the above process until a match is found or there is only one word left.

For scientific names, Kestrel will query GBIF, EOL, NCBI, and IUCN (if an API key is provided). If a match is not found among these four, Kestrel will search Wikipedia to find a consensus match. If no match is found, it will not attempt to remove words from the search term since scientific names *should* be in binomial format.

Any queries which do not return a match will be written to KestrelMisses.csv in the same directory as the output file. This is a temporary file so, if the program is not interrupted, you might not notice it.

### ***Google Search***

Once the API-based search is complete, Kestrel will automatically submit the terms in KestrelMisses.csv for a Google search using Selenium. Since this process cannot easily be run in parallel and using Selenium as an intermediary adds to the time taken to find a match, this step is reserved for terms which could not be resolved using simpler means. If the “--firefox” flag is given, Kestrel will open a Firefox browser; otherwise, it will open Google Chrome.

For each term, Kestrel will search for the given term and the word “taxonomy.” It creates a list of the urls found in the search results and iterates through them. If a url is hosted on Wikipedia, IUCN, or [ITIS](#) (which are the most common results with complete taxonomies), the program will search that page for taxonomy data. Once the first complete match is found, it is appended to the output file and Kestrel proceeds with searching for the next term. Once all terms in KestrelMisses.csv are searched, the file is removed.

## Output

For names with an identified taxonomy, output will be written to the given output file in csv format. The query name will be in the first column, followed by a columns for each taxonomy field. There is one column for each database that Kestrel queries and urls from accepted hits will be recorded in the source database's column. This identifies the source of the data and allows it to be accessed again. Any names for which a taxonomy is not identified will be written to the KestrelNoMatch.csv file, which will be located in the same directory as the output file.

## Scripts

### *kestrel.py*

This is the only executable script in the package and will take input from a given column of a given file and search GBIF, EOL, CBI, and Wikipedia for taxonomy information.

### Usage:

```
python kestrel.py {options} -i <input file> -o <output file>
```

- h, --help show this help message and exit
- v Prints version info and exits.
- extract Extracts and filters input names.
- common Indicates that input contains only common names (use with --extract).
- scientific Indicates that input contains only scientific names (use with --extract).
- c C Column containing species names (integer starting from 0; use with --extract)).
- i I Path to input file.
- o O Path to output csv file.
- t T Number of threads for identifying taxa (default = 1).
- firefox Use Firefox browser (uses Chrome by default).