Elsevier API Query Tool – the Data Fetcher

Getting Started Guide

Version 7.4.3

ELSEVIER

Table of Contents

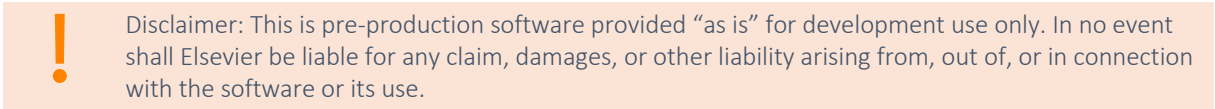
Overview

The API Query tool, affectionately known as the Data Fetcher for its ability to fetch data from Scopus, PlumX, SciVal, and other APIs, was built in-house by an inspired Elsevier employee who wanted to help his clients more easily download and analyze publication and bibliometric data. In addition, the tool supported and continues to support analytical and intelligence use cases with needs beyond what's available in the .com interfaces of Elsevier's Research Products, but that depend on the data available in those products. The menu-driven interface of the tool makes it easy to use and serves both casual users without the time or background to code as well as advanced coders and scripters.

On the technical side of things, the Data Fetcher requires no installation. It is a self-extracting .exe file run on your device, and you simply need to download the tool archive via dev.elsevier.com/datafetcher, enter the password (contact datafetchersupport@elsevier.com), unzip the file into a local folder (*e.g.* on your Desktop, *etc.*), ensure your personal API key is placed into the tool's configuration file, and then run the executable in place.

Why use the Data Fetcher?

- You are a data analyst but not a programmer
- You do not have the time to code
- Your data needs exceed what is possible or practical using the built-in export facilities of Scopus.com and other Elsevier products the tool accesses (*e.g.* PlumX Metrics, the Elsevier Fingerprint Engine, *etc.*
- You need to mix data from several sources into one cohesive output (i.e. Scopus, SciVal, PlumX, etc.)
- The Scopus.com interface isn't optimized for your use case, but Scopus data is



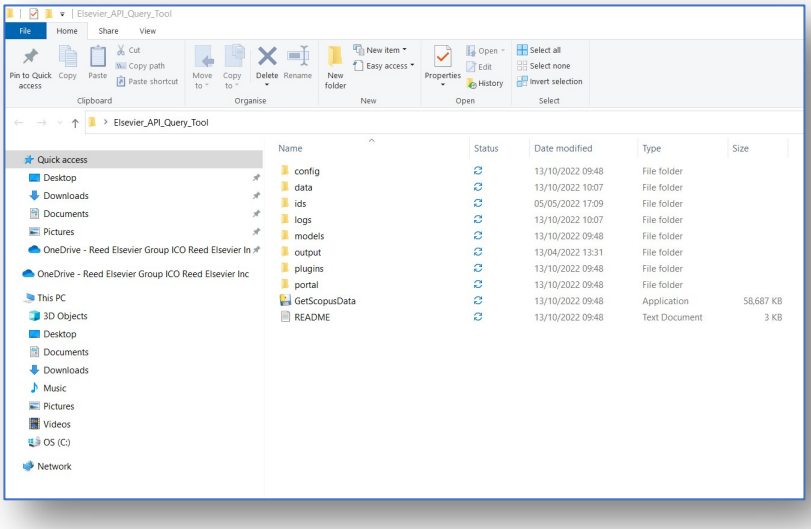
Prerequisites

- Machine that runs Windows. There is an optional web-based interface that can be accessed from other platforms (*e.g.* MacOS, Linux). We are also currently working on a MacOS-compatible version.
- Data Fetcher downloaded via Dev Portal with permission of Data Fetcher team

- Microsoft Excel, R, or other tool for analyzing table-based results.

Installation Process

To begin, please download the Data Fetcher archive via dev.elsevier.com/datafetcher. The zip file contains an executable (.exe) file which allows users to run this tool on their hard drive. Currently, the file/tool only runs on Windows machines (MacOS is currently being worked on).



Once you have downloaded the tool via the Developer Portal, please create a new folder in a convenient location titled “Elsevier Data Fetcher” or something similar. It’s recommended that you create additional folders under that one for each version and/or project for which you use the Data Fetcher. Then copy and paste every file and folder from the zip file into the new folder. Once this step is complete, you are ready to launch the Data Fetcher.

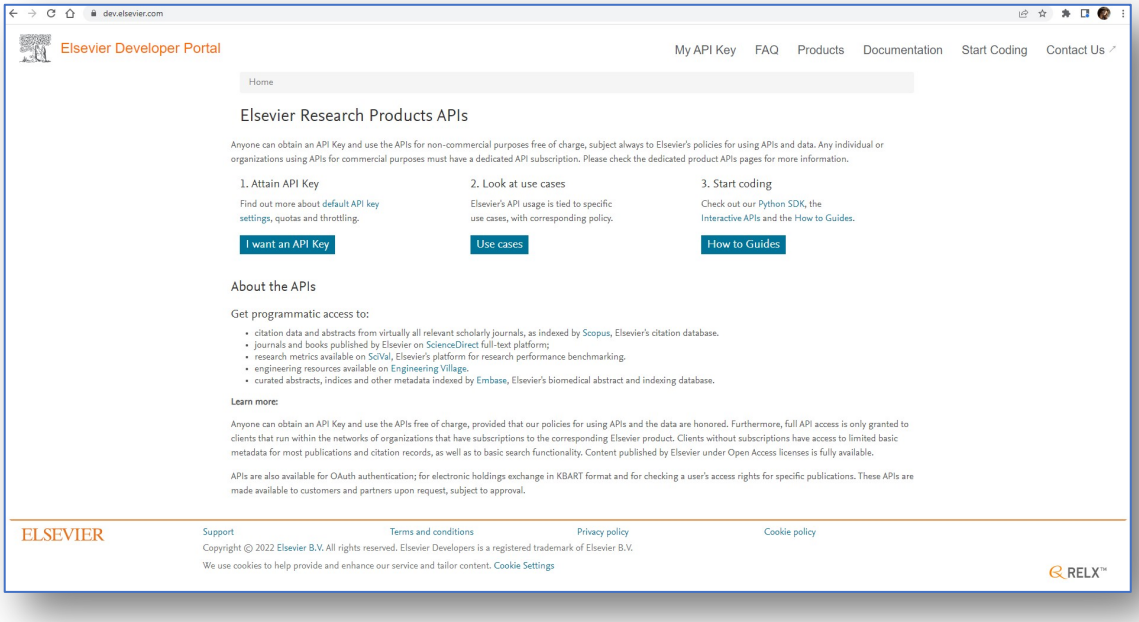
Note that since there is no installation process beyond copying the files from the Zip download archive, you can create multiple copies of the tool, each running independently within its own folder. This allows you to preserve settings and outputs for individual projects or use cases over time. This is why we recommend a two-level hierarchy for the tool, so you can create multiple project folders in which to work.

Launching the Tool

To know if your install was successful, run the .exe file by clicking GetScopusData within the Data Fetcher folder into which you copied the files from the downloaded archive. The program takes a little while to load as the file is self-extracting and sets up temporary directories from which to launch. You will see a pop-up splash screen during this process.

Configuring the tool to use your unique API Key

After executing the tool successfully, and noting the introductory pop-up, it is time to update the tool with your own unique API Key, obtained via the Elsevier Developer Portal. To obtain a key, please navigate to dev.elsevier.com and click 'I want an API Key.'



Under Label, add a descriptor referencing your project. Your website should be a valid entry, perhaps a LinkedIn or university profile if you don't have a department or project website to enter.

There are two ways to enter your API key for use in the tool.

1. You can navigate to the Credentials area of the Settings tab in the Data Fetcher tool and enter your API key there, as well as your Institutional Token if you have been provided with one.

2. You can edit the .env file in the program folder and enter the credentials there. The .env file has the advantage of separating your API key from the rest of your settings so you can more easily share settings files and/or archive them without including your API key. Also, setting up new projects becomes easier and faster, as you can

simply copy the .env file from your old project to a new one without carrying along all your project settings as well.

Please note that once your API Key has an institutional token generated and paired with it, the two credentials will always need to be together. Your individual API Key will not work alone if it has been paired with an institutional token.

Elements of the Tool

The Data Fetcher interface is divided into two main areas, highlighted below:

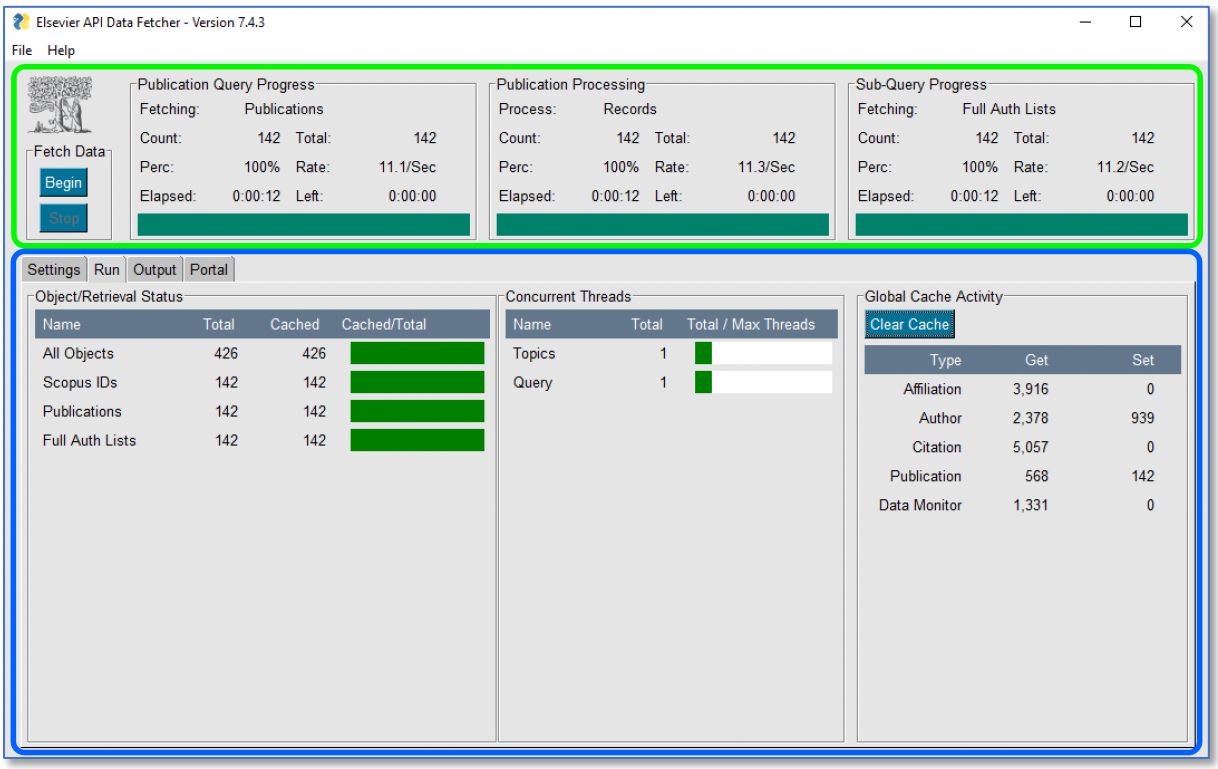
1. The main progress area, highlighted in green, shows the overall progress of the job currently being executed. There are four main sub-areas in this region:

    a. **Fetch Data** – this area holds the **Begin** and **Stop** buttons, where you can initiate the main search and stop it mid-process if required. Note that stopping the process takes a few seconds as all currently executing queries and threads must quiesce first.

    b. **Publication Query Progress** – this box holds information regarding the querying and loading of the publications being gathered for output. You will notice that generally this part of the process finishes first, then final processing completes thereafter.

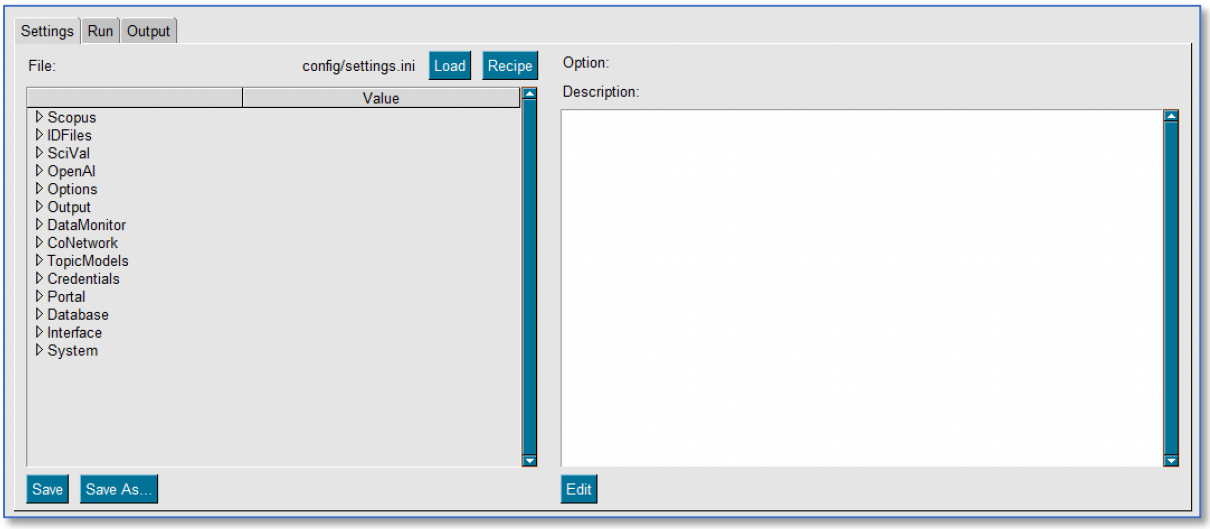
    c. **Publication Processing** – this box shows the overall progress as each publication is fully processed, including all sub-queries, plugin activities, and other work. The overall job is completed when this box has finished.

    d. **Sub-Query Progress** – this box shows the status of various sub-queries that might be required to complete the job. In the example below, you'll note that "Full Auth Lists" is being tracked, which indicates the tool is fetching full author lists for all publications, rather than simply using the default, incomplete lists returned from a basic search. The contents of this box will change greatly based on the output fields selected.



2. The lower area in the figure above, highlighted in blue, displays various sub-components. Each will be explained below in a separate section.

Explanation of the Settings tab



The Settings tab shows all the settings of the application. In addition, there are several buttons and an editing box to the right.

1. The **Load** button allows you to choose between various config files. This is useful if running different projects. Once you load a new config, future Save operations will save to that file instead of the default.

2. The **Recipe** button allows you to overwrite individual settings temporarily for a particular project. A recipe file is essentially just a full settings.ini file with all non-relevant parts stripped out. Only those settings that remain will be applied over the full settings that have been loaded. For instance, this would allow you to save a particular Scopus Query for re-loading later, while leaving all other settings as-is. Note that if you Save after loading a recipe, it will overwrite the main settings file with the new state.

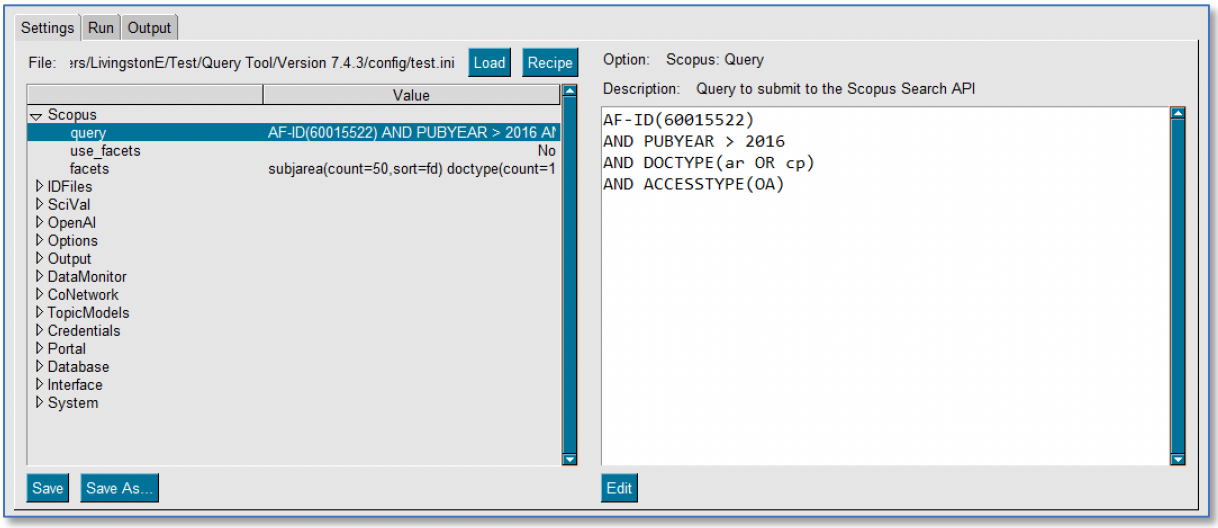
3. The **Save** button saves the active settings file, listed in the "File:" Area to the left of the Load button. This includes any updates and recipe overwrites that have been applied to it.

4. **Save As…** saves the existing settings to a new filename. Note that once you do that, the new file becomes the active file, and future Save operations will save to the new file and not the previous one.

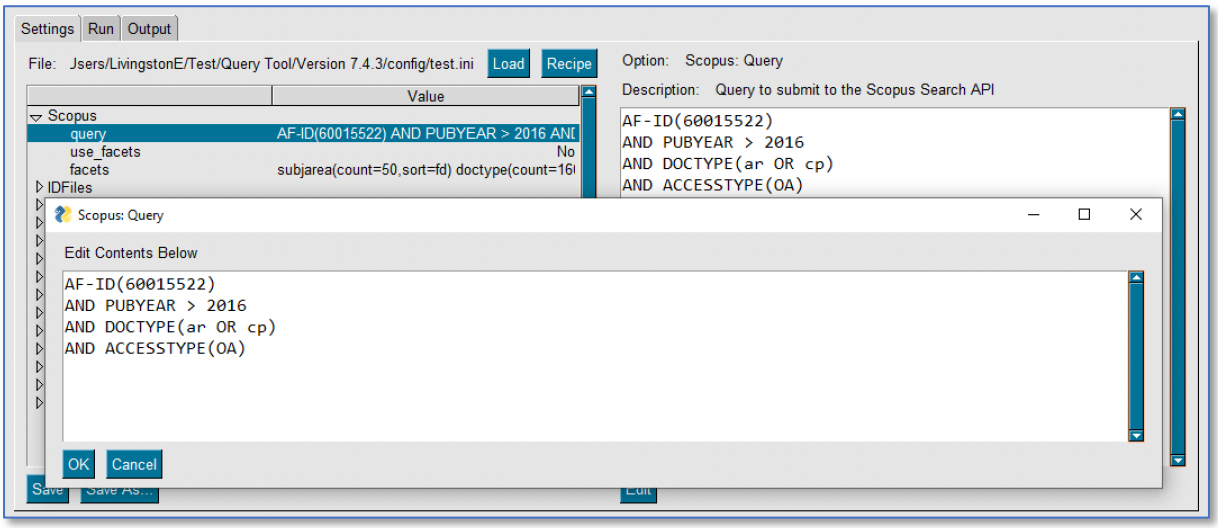
5. The box to the right is where you can edit/adjust any setting by clicking on it to the left in the tree. Below that box is an **Edit** button which will pop up a new window for potentially easier editing.

Editing Settings

To edit a setting, first expand the displayed tree of settings in the left pane and click on the setting you would like to change. The value of the setting will be shown in the box to the right, wherein it may be edited. In the figure below, we've selected the Scopus Query setting, and that query text shows up on the right. This can be edited in place. Once you have completed editing, click the Save button in the lower left of the tool's window.

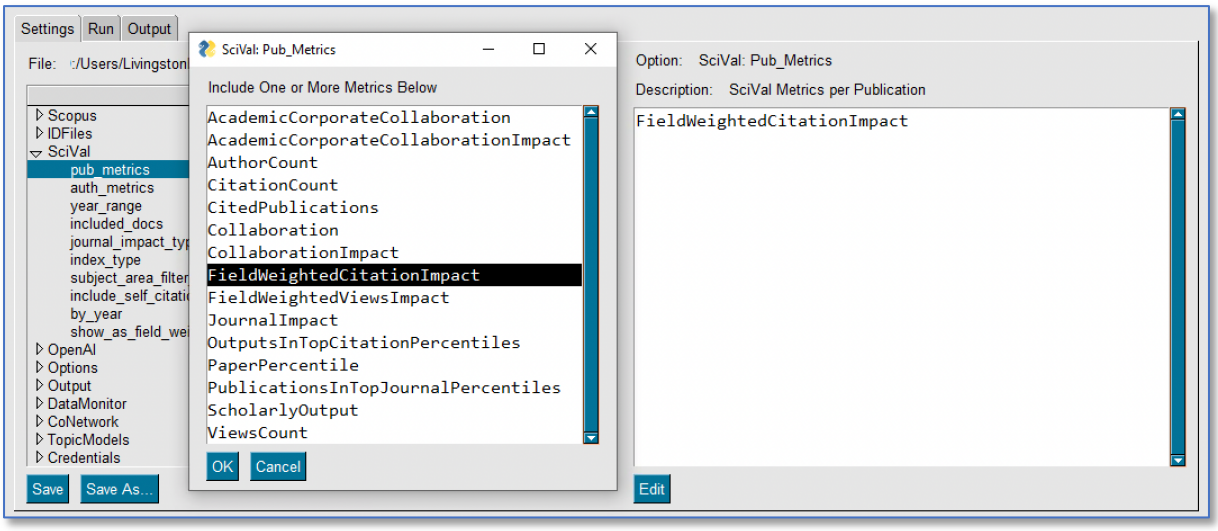


In addition to directly editing the value in the box, you can also click the **Edit** button located below the editing pane. This will pop up a dedicated window which can be resized to allow for a much larger area in which to type; this is handy for large queries, for instance.



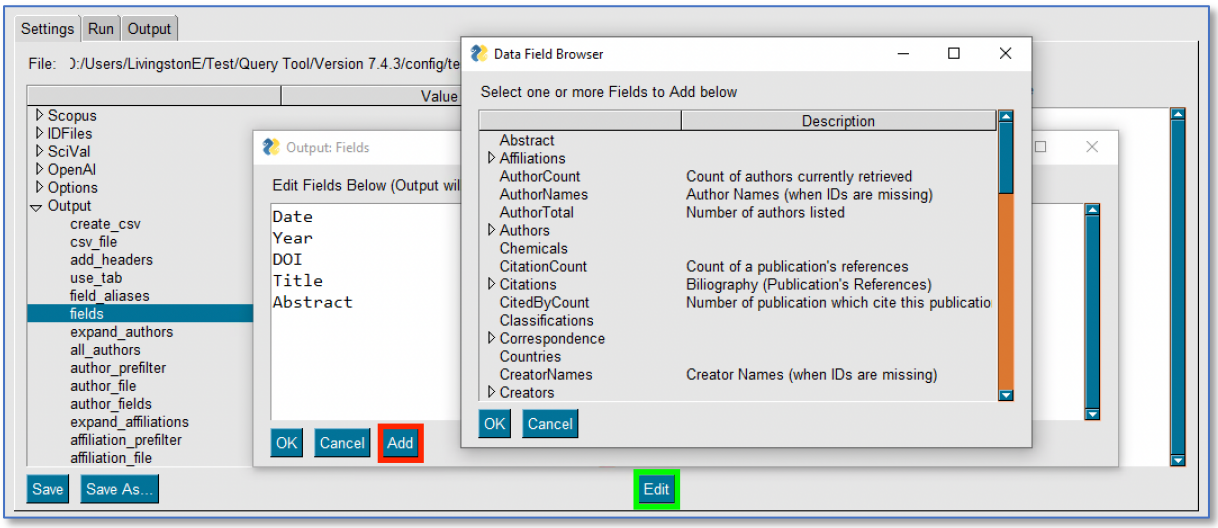
Note that for many types of settings, clicking on the **Edit** button will bring up a much more helpful interface for editing. For instance, for any files or folders, clicking on **Edit** will bring up a one-line window where you can change the name directly, or you can click the **Browse** button to open a familiar file-selection panel.

Similarly, if editing a list of values that must all fall within a defined list, clicking on **Edit** will bring up a dedicated panel that allows you to choose from the available options, such as in this example that presents valid metrics available from SciVal.



Finally, in some cases, clicking on the **Edit** button will bring up a window in which you can type and adjust values, but will also present an Add button. This button can be clicked to bring up a dedicated browsing window wherein values can be selected from an expandable

tree of options. In the example below, the Output->fields option has been selected on the left. Clicking the green-highlighted **Edit** button on the right brings up an editing box, and then clicking on the red-highlighted **Add** button will open the browsing panel, as shown.

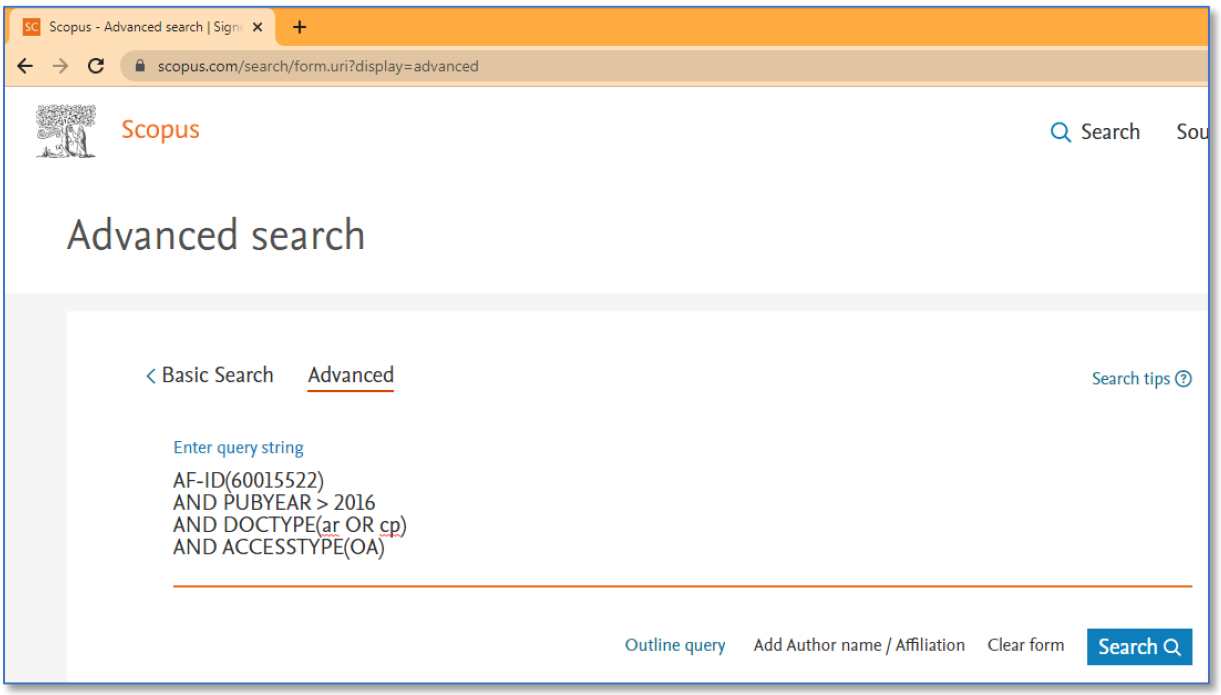


> **!** Settings available in the browser version of the tool are bolded in the table below.

Scopus Settings

Scopus Query Syntax

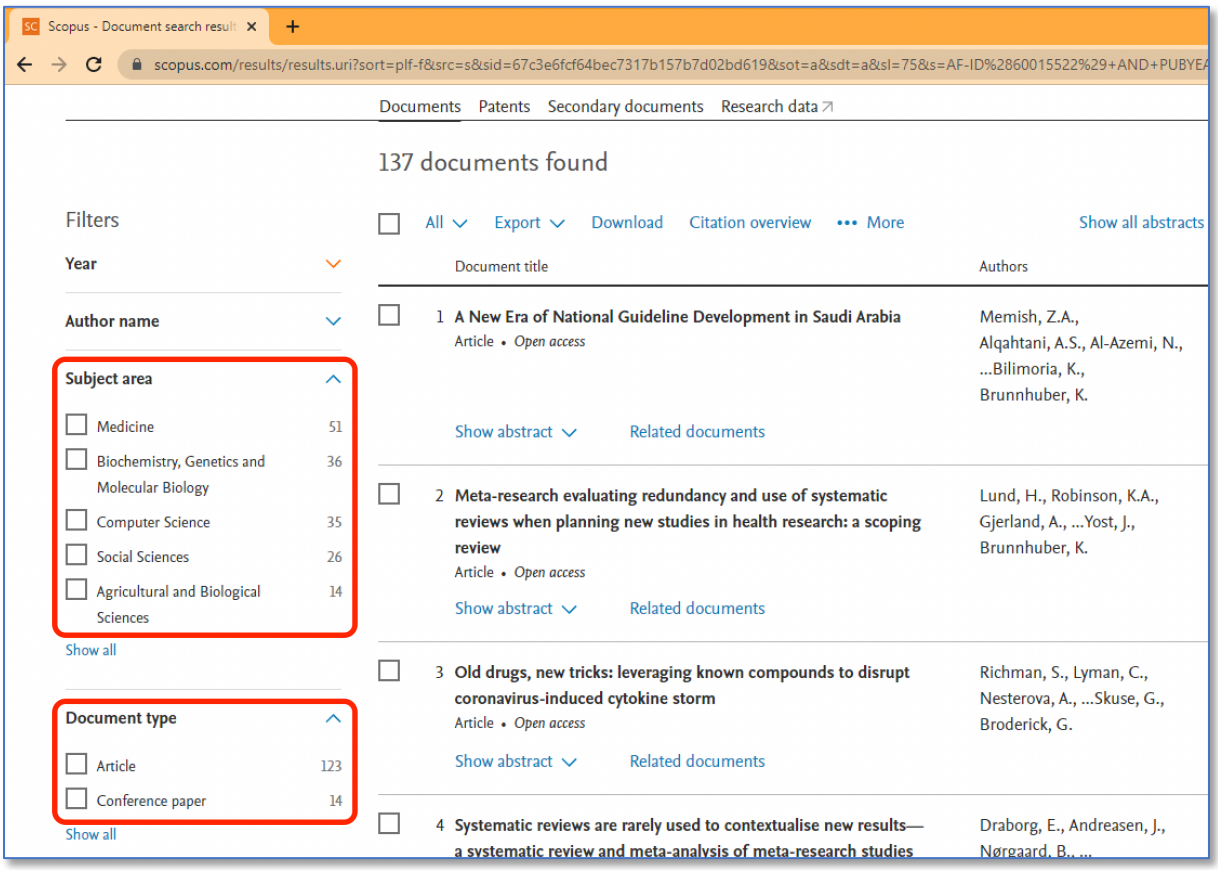
The two primary ways of using the Data Fetcher are to issue queries to Scopus and return a set of publications, or to direct the tool to a file containing IDs to look up. This first section pertains to the Scopus query mode of operation. The query syntax is essentially similar to that used in the Advanced Search of Scopus. In fact, it's recommended that you first use Scopus Advanced Search to form your query and ensure the results are what you are expecting, prior to copying your query into the Data Fetcher to perform a full download. This is a faster way to iterate, test various query options, and debug query issues and errors.



Facets

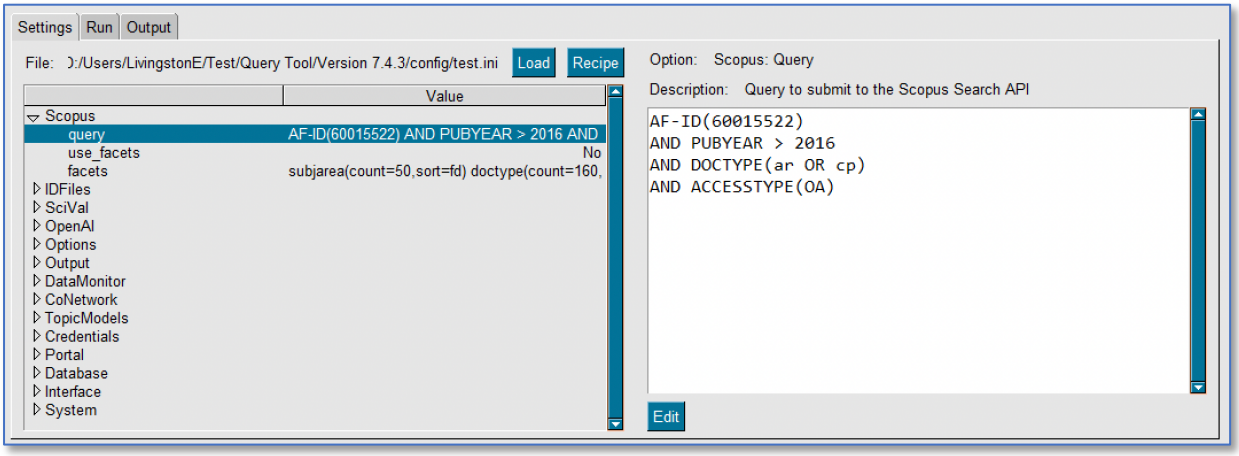
In the settings below, you will see options for “facets”: these are equivalent to the Search Result filters in Scopus, as depicted below, outlined in red. If enabled, the tool will create a separate file with totals for the faceted categories. For example, if Document Type is selected as a facet, and the search below is performed in the tool, then two rows would be created in the facet file: one for type “Article” showing a total of 123, and another for type “Conference paper” showing a total of 14.

Note that facets will only show a maximum of 160 entries. That is, if you were to facet on Author, only the top 160 most-published authors would appear in the facet file. For a complete list of more than the top 160 entries, the dataset would have to be downloaded and counted explicitly.



Scopus Settings Table

The Scopus settings within the Data Fetcher are shown below:



Value	Description	Value Options	Additional Notes
query	Query to submit to the Scopus Search API endpoint	Scopus Search categories, full schema of options viewable here	The tool follows the fundamental syntax of Scopus Advanced search. Please see the Scopus

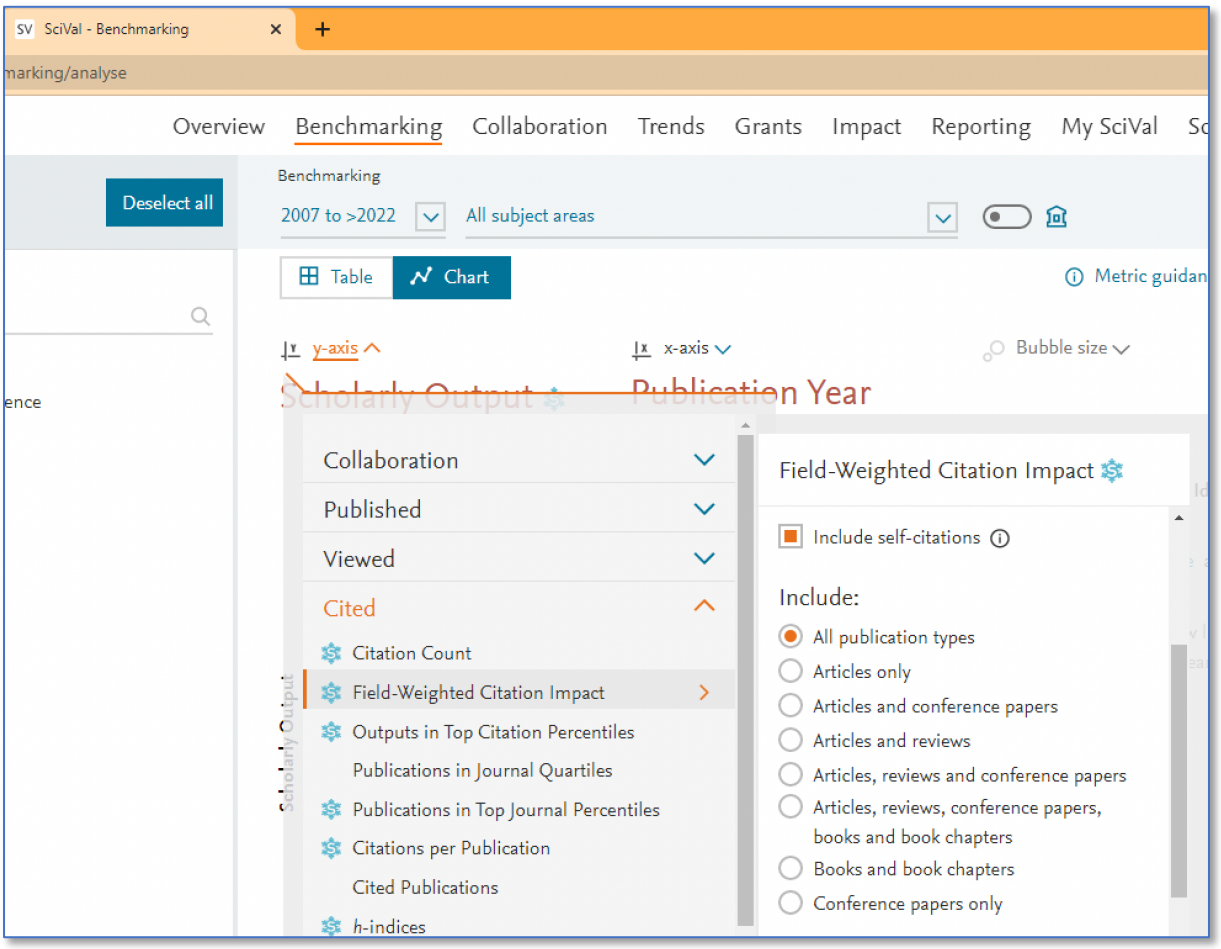
Value	Description	Value Options	Additional Notes
			search tips page, and these six simple search tips from Elsevier.
use_facets	The facets are the Refine results categories on the left-hand sidebar of the Scopus.com search results page.	Available values are: Yes; No Default: No	Leave as No if you don't need facets, for quicker processing Otherwise, facets will return subtotals for each requested facet.
facets	Enter any facets to retrieve	Example value is subjarea(count=50, sort=fd) count : the number of "buckets" to include (i.e. how many navigator entries) sort : how the navigators should be sorted. Options include **na** (Modifier name, ascending), **fd** (Modifier frequency, descending), and **fdna** (Modifier frequency descending, secondary sort through unity by name, ascending).	Available facets include: AF-ID - affiliation identifier AUCITE - author citation AU-ID - author identifier AUTHNAME - author identifier and author name COUNTRY - affiliation country EXACTSRCTITLE - source title FUND-SPONSOR - funding sponsor LANGUAGE - language OPENACCESS - open access status PUBYEAR - publication year RESTYPE - internal collection SUBJAREA - subject area SRCTYPE - content category

SciVal Settings

The Data Fetcher is able to fetch SciVal data for publications and authors found in the returned data following a query or ID lookup from Scopus. These additional SciVal data will be merged with the Scopus and other data fetched for the publications and authors and made available in the output file(s).

SciVal Options

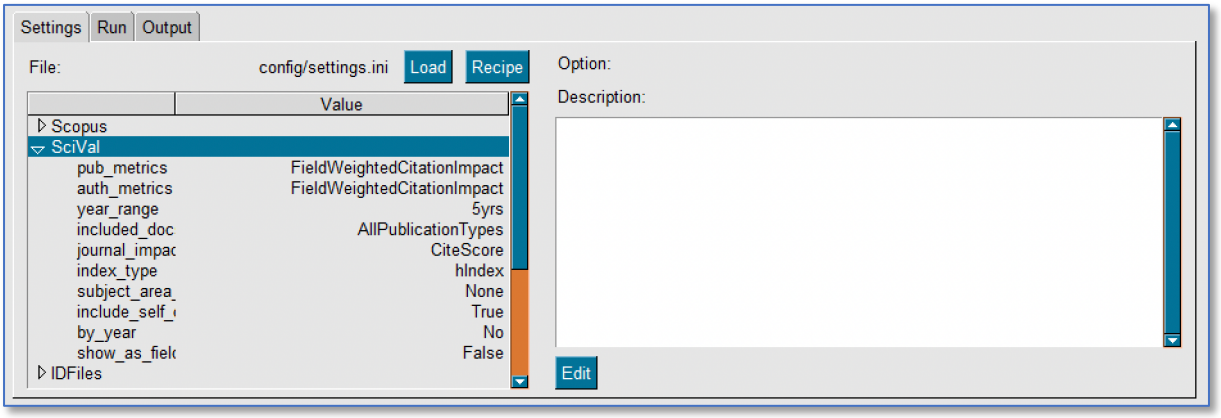
To properly fetch SciVal data, the tool must be configured with the appropriate SciVal options for the requested metrics. The options presented are equivalent to options you would find and set in the SciVal interface. For example, consider the SciVal Metric selection menu below:



In the above screenshot, the Field-Weighted Citation Impact (FWCI) option has been selected and a sub-menu has popped out to the right. Within that sub-menu, one can select whether to include self-citations and the scope of the research ouput to include in the calculation: All publication types, Articles only, and so forth.

Similarly, below in the Data Fetcher tool screenshot, options exist that parallel these: pub_metrics allows you to select FWCI, included_doc selects the types of output, and include_self selects whether to include self-citations.

SciVal Options Table



Value	Description	Value Options	Additional Notes
pub_metrics	SciVal metrics per publication	Available values are: AcademicCorporateCollaboration; AcademicCorporateCollaborationImpact; AuthorCount; CitationCount; CitedPublications; Collaboration; CollaborationImpact; FieldWeightedCitationImpact; FieldWeightedViewsImpact; JournalImpact; OutputsInTopCitationPercentiles; PaperPercentile; PublicationsInTopJournalPercentiles; ScholarlyOutput; ViewsCount;	Click the Edit button below the right-hand textbox to easily add and remove options from this list. Enabling options in this list will cause the corresponding data to be included in the information returned by the API. It's recommended to only include what you need to reduce bandwidth and time required to retrieve your publication data.
auth_metrics	SciVal Metrics per author	Available values are: AcademicCorporateCollaboration; AcademicCorporateCollaborationImpact; CitationCount; CitationsPerPublication; CitedPublications; Collaboration; CollaborationImpact; FieldWeightedCitationImpact; HIndices; OutputsinTopCitationPercentiles; PublicationsInTopJournalPercentiles; ScholarlyOutput	These options are similar to the options above, but correspond to authors, rather than publications. For instance, here you would choose Field Weighted Citation Impact (FWCI) if you wanted to retrieve the overall FWCI for an author, taking int consideration all of their research output appropriate for that metric.

Value	Description	Value Options	Additional Notes
year_range	Year range for SciVal metrics	Available values are: Default; 3yrs; 3yrsAndCurrent; 3yrsAndCurrentAndFuture; 5yrs; 5yrsAndCurrent; 5yrsAndCurrentAndFuture; 10yrs	These timeframes correspond to the equivalent drop-down option in the SciVal.com web interface.
included_docs	Included documents for SciVal metrics	Available values are: AllPublicationTypes; ArticlesOnly; ArticlesReviews; ArticlesReviewsConferencePapers; ArticlesReviewsConferencePapersAndBookChapters; ConferencePapersOnly; ArticlesConferencePapers; BooksAndBookChapters	These options correspond to the equivalent drop-down options in the SciVal.com web interface.
journal_impact_type	Journal Impact Type for SciVal metrics	Available values are: Cite Score; SNIP; SJR	These options correspond to the equivalent options in the SciVal.com web interface.
index_type	Index type for SciVal authors	Available values are: hIndex; h5index; gIndex; mIndex	These options correspond to the equivalent options in the SciVal.com web interface.
subject_area_filter_uri	Subject area filter URI	You can add a subject area filter URI	https://scival.com/overview/topics?uri=World/3&newTopicsOnly=true
include_self_citations	Include self-citations in SciVal metrics?	Available values are: Yes; No	Default value is True (Yes)
by_year	Include By-Year subtotals for SciVal metrics?	Available values are: Yes; No	Default value is False (No)
show_as_field_weighted	Show SciVal metrics as field-weighted?	Available values are: Yes; No	Default value is False (No)

ID Files Settings

The Data Fetcher's second primary mode of operation is the lookup of pre-existing lists of IDs. For example, given a list of DOIs aggregated from another source, the tool can look up Scopus (and other) details related to each of those DOIs. However, the tool can do more than one-to-one publication ID lookups. Given a list of Scopus Author IDs, the tool can query and return the full or a partial list of publications by those authors. Similarly, it can search for publications by Affiliation ID, Journal ISSN, and more.

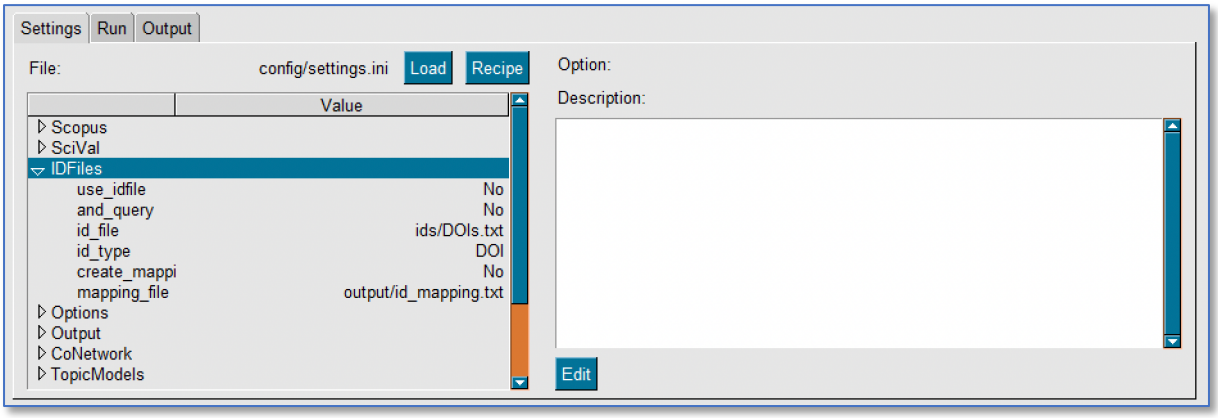
Filtering ID-based Retrievals Using a Query

In addition, the tool can filter the found set of publications by whatever appears in the Scopus Query entry of the Settings. For instance, if the query contained “Pubyear > 2020” and a list of Author IDs was provided to the tool, it could search for all publications by those authors published after 2020.

Mapping IDs to Publications

Finally, the tool can create a mapping between each of the provided IDs (*e.g.,* Author IDs) and the subsequent list of retrieved publications. This allows each author ID to be easily matched to the set of publications on which the author was listed. However, note that by enabling the mapping file, the initial gathering of publication IDs by the tool will be slowed down. This is because each author ID must be submitted separately to Scopus, one at a time, to establish the exact set of publication IDs returned for that author. If the mapping file is disabled, the tool can aggregate many author IDs together in a single query, more efficiently gathering publication IDs that correspond to any of the provided author IDs at once. It is still possible to map the authors to their individual publications, of course, during post-processing of the data, as each publication-author combination can be exported as part of the set of output files produced by the tool.

ID Files Options Table



Note that by enabling the id file option, the ID file is used instead of the Scopus query, or in combination with it, depending on the options below. The Scopus Query will not be executed on its own if the option to use an id file is instead selected.

Value	Description	Value Options	Additional Notes
use_idfile	Load publication IDs directly from id_file?	Available values are: Yes; No Default: No	This setting will **override** using the Scopus Search. Instead of performing the search listed under Scopus->Query, the tool will instead read IDs directly from the indicated file.

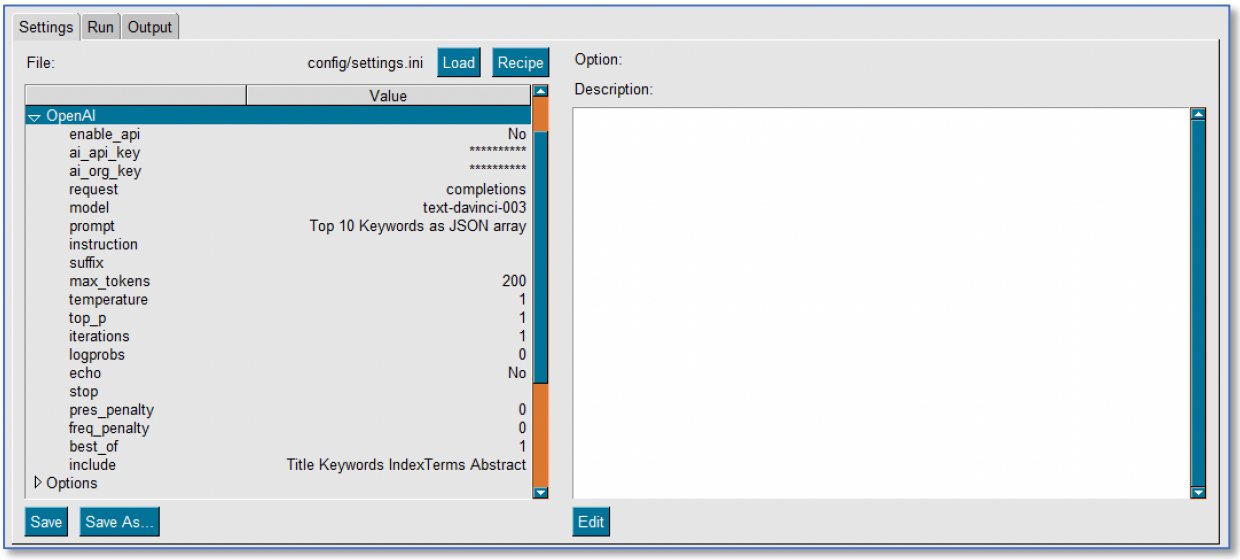
Value	Description	Value Options	Additional Notes
and_query	Filter ID searches by ANDing with Query? If Yes, this option indicates that only publications that satisfy the Query listed in the Scopus->Query option will be returned from the ID search.	Available values are: Yes; No Default: No	Example: You'd like to retrieve publications from a list of Author IDs, but only those published after 2020. You would supply the ID file below, and set the Scopus->Query option to Pubyear > 2020 and the tool would combine these to form the result set.
id_file	ID File containing EIDs, DOIs, or other IDs to download	File Name (e.g. EIDs.txt)	ID files are simple text files that contain one ID per line. They are typically stored in the "ids" subfolder of the tool.
id_type	ID type in ID_FILE	Available values are: EID; DOI; Scopus-ID; PubMed_id; Dynamic; RefEID; AU-ID	This informs the tool what sort of IDs the ID file contains. Note the special permission is required for the RefEID option. The Dynamic option is special, for mixed-ID collections. Inquire with Support if this might apply to your use case.
create_mapping	Create mapping to AU-ID, RefEID, etc? (NOTE: Scopus Only)	Available values are: Yes; No Default: No Note: Generally this should only be used for RefEID: any other mapping can be done faster and use far less quota with the Plugin **MapIDFields** option. See the Plugin section for more.	If Yes, this option causes a file (indicated in the option below) to be created, containing a mapping of ID-File IDs to publication IDs. Note that enabling this option will slow down the ID retrieval process and consume quota a bit faster an individual API query must be performed for each ID in the ID file to create the map.
mapping_file	File to receive InID->MappedID mappings?	Default is output/id_mapping.txt	The file can be specified by the user. This file will create a one-to-many mapping from ID to publications. Example: if id_file contains Author IDs, this file will contain a row for each author-id/pub-id pair in the result set.

OpenAI Settings

The Data Fetcher comes with the ability to submit publication titles, keywords, abstracts, and other elements to OpenAI's GPT and other API interfaces. This can be handy for requesting short summaries of article abstracts, generating keywords, and other enhancements. A full

treatment of the OpenAI API framework is beyond the scope of this manual. For details, refer to the following URL: https://beta.openai.com/docs/api-reference/introduction

OpenAI Settings Table



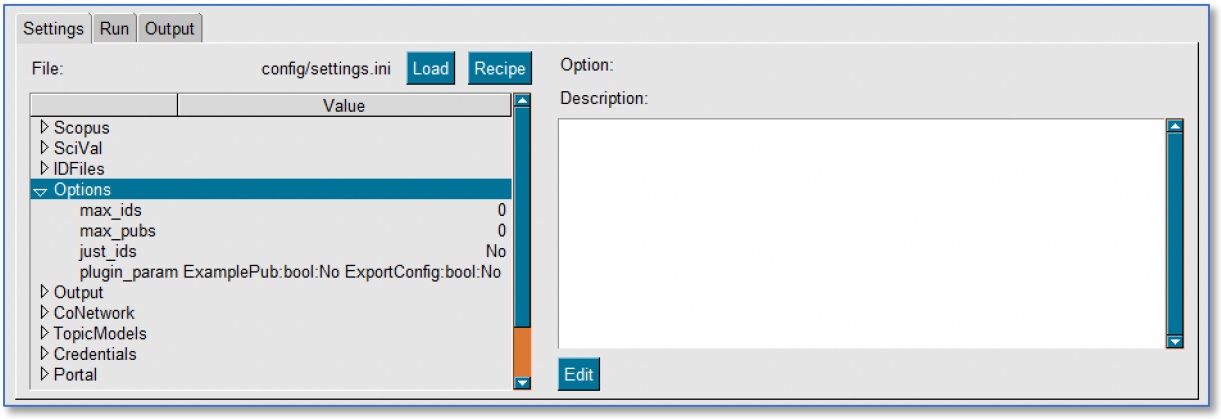
Value	Description	Value Options	Additional Notes
enable_api	Enables or Disables access to the OpenAI API Interface	Available values are: Yes; No Default: No	OpenAI's API interface is a paid-for service. This option allows you to enable it only when required for cost management
ai_api_key	OpenAI API Key	Default is empty This is where you would place your API key once you have signed up for one at OpenAI	It's best to place this in your .env file to ensure it's controlled and not inadvertently shared, as this API incurs usage charges
ai_org_key	OpenAI Org Key	Default is empty This is where you would place your Org key once you have signed up for one at OpenAI	It's best to place this in your .env file to ensure it's controlled and not inadvertently shared, as this API incurs usage charges
request	The specific request type (API endpoint) you would like to access	Available Options are: Completions Edits Embeddings	These are the three most-useful endpoints when submitting article abstracts and information to OpenAI's models
model	The OpenAI Model to use for the request	See OpenAI's documentation for details	See OpenAI's documentation for details

Value	Description	Value Options	Additional Notes
prompt	The prompt(s) to generate completions for	See OpenAI's documentation for details	See OpenAI's documentation for details
instruction	The instruction that tells the model how to edit the prompt	See OpenAI's documentation for details	See OpenAI's documentation for details
suffix	The suffix that comes after a completion of inserted text	See OpenAI's documentation for details	See OpenAI's documentation for details
max_tokens	The maximum number of tokens to generate in the completion	See OpenAI's documentation for details	Note that charges are assessed per-token so this is an important parameter to keep track of
temperature	What sampling temperature to use	See OpenAI's documentation for details	See OpenAI's documentation for details
top_p	An alternative to sampling with temperature	See OpenAI's documentation for details	See OpenAI's documentation for details
iterations	How many completions to generate for each prompt	See OpenAI's documentation for details	This is simply called 'n' in the API documentation, and >1 can incur multiplicative charges
logprobs	Include the log probabilities	See OpenAI's documentation for details	See OpenAI's documentation for details
echo	Echo back the prompt in addition to the completion	See OpenAI's documentation for details	See OpenAI's documentation for details
stop	Up to 4 sequences where the API will stop generating further tokens	See OpenAI's documentation for details	See OpenAI's documentation for details
pres_penalty	presence_penalty	Number between -2.0 and 2.0	See OpenAI's documentation for details
freq_penalty	frequency_penalty	Number between -2.0 and 2.0	See OpenAI's documentation for details
best_of	Generates best_of completions server-side and returns the "best"	Integer, defaults to 1	Another option that can multiply charges. See OpenAI's documentation for details
include	Specifies which fields to include in the information passed to the OpenAI API	Options include: Title Keywords IndexTerms Abstract Body Fingerprint	Include more for better results, but balance with charges as each token submitted for analysis counts against the max_tokens and also incur charges.

Options Settings

The Data Fetcher has a few options that impact the overall job execution. The settings in this section will limit the amount of information obtained via the APIs and allows for configuration of the plugin architecture, which is described in a separate section.

Options Settings Table



Value	Description	Value Options	Additional Notes
max_ids	Maximum IDs to import from ID file, if used. Note the subset will be a random sample from the IDs	Numerical	Enter 0 for all. This refers to IDs listed in the IDFiles->id_file option. If IDFiles->use_idfile is false, this setting has no effect.
max_pubs	Maximum publications to download. Note this will be a random sample taken from the total IDs available for download.	Numerical	Enter 0 for No Limit It is helpful to first run your query with a small number to ensure that the outputs meet your expectation before running the full data pull
just_ids	Only download publication IDs	Available values are: Yes; No Default: No	Enter Yes for import elsewhere (*e.g.,* SciVal) If Yes, will only download list of publication IDs that satisfy the search.
plugin_params	Custom parameters for your plugin file, if present	Example: ExamplePub:bool:No ExportConfig:bool:No ExportASJCFile:bool:No ProcessAuthors:bool:Yes CheckAuthAffs:set:	This is a dynamic list of options available to the plugin system of the Data Fetcher. See the appendix on Plugins for further details and instructions on the use of this section of Settings.

Output Settings

The primary output mode of the Data Fetcher is to create several comma or tab-delimited files, each with one row corresponding to each entry of the type of data it contains. The most common output of the tool (and the preconfigured mode by default) creates three primary output files:

1. **Publications.txt** – This contains one row for each publication

2. **Authors.txt** – This contains one row for each publication-author combination. That is, if a publication has 10 authors, this file will have 10 rows for that publication, one each for each author. Note that if an author appears on multiple publications, that author will have multiple rows in the author.txt file: one row each for each time that author was associated with a different publication.

3. **Affiliations.txt** – This file contains one row for each affiliation listed on a paper. That is, if a publication's authors listed a total of 5 affiliations between them, there would be five rows in this file, one each for each of the five affiliations. Note that if an affiliation is listed on more than one publication in the returned set, there would be one row in this file for each occurrence of that affiliation on a different paper.

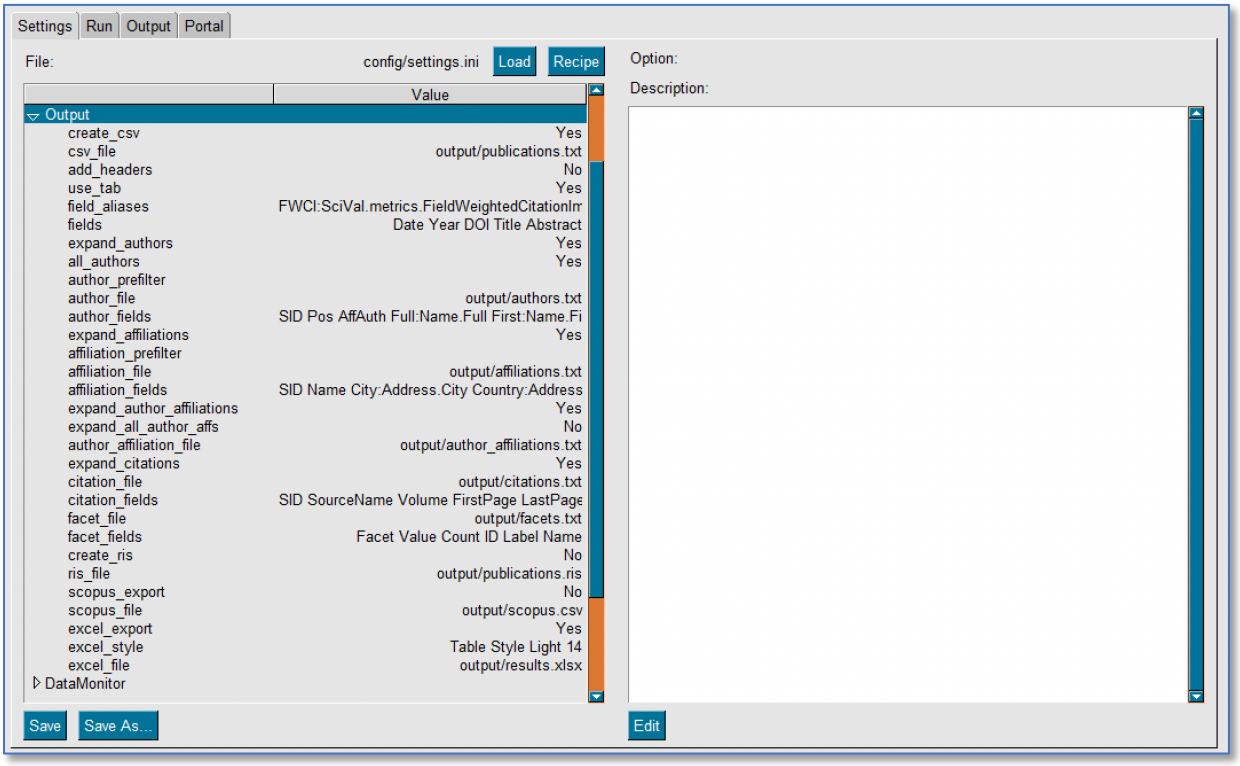
Other files which can be produced include **author_affiliations.txt** which includes a row for each affiliation listed by each author for each paper (this can get quite large), **citations.txt** which outputs the bibliography for each paper, one row per reference, and other export files including Scopus.com format, RIS format, and facets.

Each of these output files can be customized to include only the columns of data requested. They can also be filtered to include only the rows necessary, they can combine related data together, and have other options as well, explained below.

In addition, the tool can optionally create an Excel .XLSX workbook, either instead of the tab-delimited files, or in addition to them. In this case, the workbook will have tabs within it corresponding to each of the named .TXT files with the same information, arranged as tables. Note that there are limits on the size of both individual cells in Excel and the total rows allowed in a spreadsheet, so for very large datasets only the tab-delimited text files will support them.

The tool's plugin architecture allows for an infinite variety of additional outputs to be created as well, including the default ASJC mapping file which ships as part of the initial configuration. See the appendix on Plugins for further details.

Output Settings Table



Value	Description	Value Options	Additional Notes
create_csv	Create comma-delimited output files? (Typically, tab-delimited, however)	Available values are: Yes; No Default: Yes	This is the main output format for the data returned by the tool.
csv_file	Main publication output file	Default: output/publications.txt	–
add_headers	Add header information to publications and facet output files?	Available values are: Yes; No Default: No	Header information includes date, query, and other useful information for archiving.
use_tab	Use Tabs instead of commas to delimit output files	Available values are: Yes; No Default: Yes	Can still be opened in programs like Excel but does not have the same issues that large .csv exports sometimes face.
field_aliases	User-defined field shortcuts	Create a short name for a data value	E.g. Policy Citations: PlumMetrics.citation.policy_citby_count.total

Value	Description	Value Options	Additional Notes
fields	Fields to include in the publication output file	Default values are: Date; DOI; Title	A full list of fields is available as a dropdown menu by clicking Edit then Add
expand_authors	Create an author file in addition to the publication file?	Available values are: Yes; No Default: Yes	Author files list each publication author on a separate line. If an author is on multiple publications, there will be multiple entries, one for each publication-author combination.
all_authors	Fetch All authors vs. Top 100 (Separate API Call for each publication)	Available values are: Yes; No Default: Yes	By default, only the first 100 authors are available per publication. If this is Yes, each publication will be checked and those with more than 100 authors will be separately downloaded to ensure the entire author list is present and output
author_prefilter	Filter authors prior to individual author fetches	Valid options are the same for author_fields. Fields included in this area must have a 'true' or non-blank value for additional details to be fetched for the author. Default: Blank	Fetching individual author details is time-consuming and consumes quota rapidly. This filter can be used to limit which authors are fetched. Example: You're only interested in details of your own institution's authors; the rest can remain un-fetched. See the appendix on filtering and plugins for details.
author_file	Author output file	Default: output/authors.txt	–
author_fields	Fields to include in the Author output file	Default: SID; Name.Full; Name.First; Name.Initials; Name.Last	A full list of fields is available as a dropdown menu by clicking Edit then Add. (SID corresponds to Scopus ID)
expand_affiliations	Create a publication affiliations file?	Available values are: Yes; No Default: Yes	As with 'expand_authors', this option will cause a separate file to be created, with one row for each affiliation associated with each publication.
affiliation_prefilter	Filter all Affiliations prior to Individual Affiliation Fetches	Affiliation fields. Must have a "true" or non-blank value for details to be fetched for the affiliation. Default: Blank	This allows for limiting which affiliations are submitted for additional details. See the appendix on plugins and filtering for more details.

Value	Description	Value Options	Additional Notes
affiliation_file	Publication affiliations file	Default: output/affiliations.txt	The file in which to place publication affiliation information
affiliation_fields	Fields to include in the Publication Affiliations and Author Affiliations (if selected for expansion)	Default values are: SID; Name; City:Address.City.Country: Address, Country	A full list of fields is available as a dropdown menu by clicking Edit then Add
expand_author_affiliations	Create an author affiliation file?	Available values are: Yes; No Default: No	**Warning**: file can get very large. This file will contain one row for each pub-author-affiliation combination.
expand_all_author_affs	Expand full author affiliation history?	Available values are: Yes; No Default: No	If Yes, the tool will retrieve detailed information on each publication author, including their entire list of affiliations stored in Scopus across all of their works (not just those satisfying the query or ID list submitted).
author_affiliation_file	Author affiliation output file	Default: output/author_affiliations.txt	
expand_citations	Create a publication citations file?	Available values are: Yes; No Default: No	Will produce an export with a separate row for each reference included in the bibliography of each of the publications specified.
citation_file	Publication citations file	Default: output/citations.txt	
citation_fields	Fields to include in the Publication Citations file	Default values include: SID; SourceName; Volume; FirstPage; LastPage; FirstYear; LastYear; Title; Text; SourceText; Authors[*].FullName; WebSite; WebType	A full list of fields is available as a dropdown menu by clicking Edit then Add
facet_file	Publication facets file	Default: output/facets.txt	Created if Scopus->use_facets is True.
facet_fields	Fields to include in the Publication facets file	Default values include: Facet; Value; Count; ID; Label; Name	

Value	Description	Value Options	Additional Notes
create_ris	Create RIS-format output file?	Available values are: Yes; No Default: No	If Yes, will create a RIS-compatible output file with publication data. Note that only fields requested in the "fields" options for pubs and authors will be output into this file.
ris_file	RIS-format output file	Default: output/publications.ris	–
scopus_export	Create Scopus-format export file?	Available values are: Yes; No Default: No	If Yes, will create a Scopus Export file compatible with 3rd-party tools such as VosViewer for visualization. Note that only fields requested in the "fields" options for pubs and authors will be output into this file.
scopus_file	Scopus-format export file	Default: output/scopus.csv	–
excel_export	Create an Excel-format export workbook?	Available values are: Yes; No Default: No	This option will enable the creation of an excel workbook, with tabs for each of the corresponding output files for pubs, authors, affs, auth affs, citations, and Data Monitor related files (see below)
excel_style	Style to apply to generated Excel tables	Default: Table Style Light 14	This is one of the named Excel styles for tables. You can set this to whatever default table style you'd like to use for exported data
excel_file	Excel-format export file	Default: output/results.xlsx	

DataMonitor Settings

From Elsevier's Data Monitor's web site: https://www.elsevier.com/solutions/data-monitor

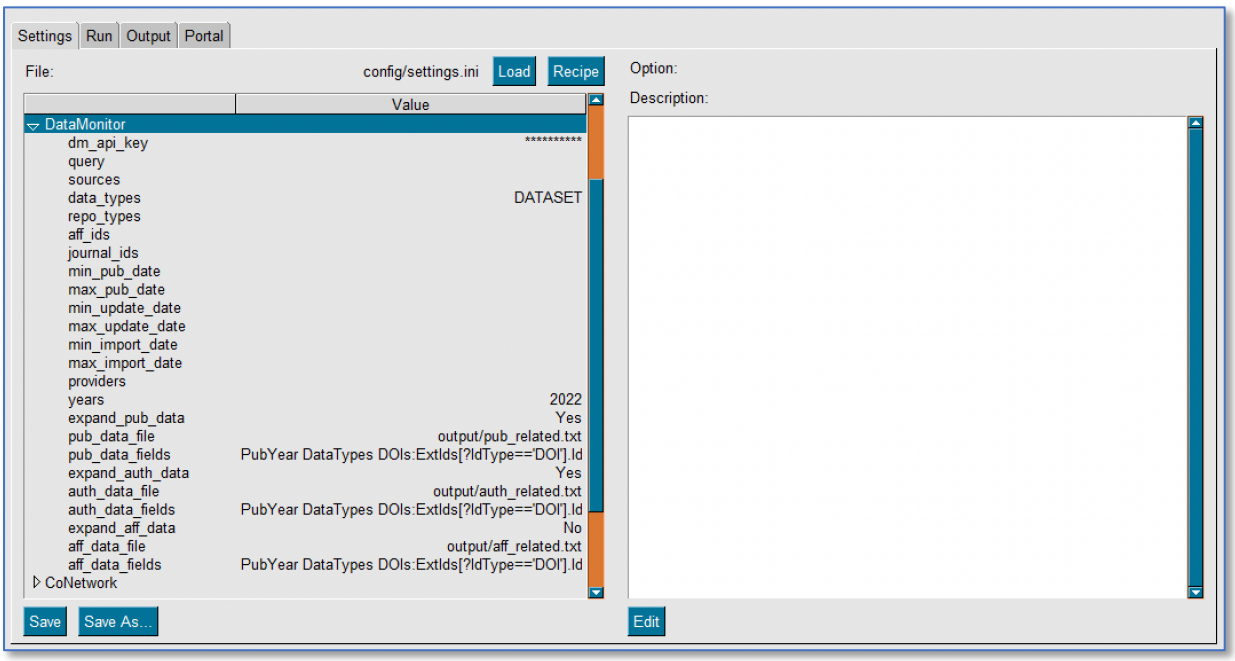
Data Monitor (DM) harvests research data from 2000+ generalist and domain-specific repositories and then cleans up and enriches the metadata by adding publications, authors, and institutions links. This makes it easy for librarians and research offices to track their institution's research data and monitor compliance.

The Data Fetcher makes use of the Data Monitor API by matching publication, author, and affiliation IDs to Data Monitor's database, allowing you to merge in the rich metadata stored

there regarding datasets, software, and many other data types related to those entities. While the Data Monitor product and interface allows for general searching of the repository, the Data Fetcher is limited to fetching data directly related to the entities it's already found and downloaded based on supplied Scopus searches, ID list files, etc. The Data Monitor Query parameter is still included, but it is used to logically "And" the results with the provided IDs, acting as a filter on the data that would otherwise be returned solely based on the publications, authors, and/or affiliations in the found set. For example:

1. You supply a Scopus Query that results in three publications being found and returned
2. You edit the query parameter below to read AUTHOR("John Doe")
3. Data Fetcher will return all records in Data Monitor that relate to the three publications the original query found, but only those that also satisfy the author query in #2.

Data Monitor Settings Table



Value	Description	Value Options	Additional Notes
dm_api_key	Data Monitor API key	Available values are: Blank; Filled Default: Blank	If blank, Data Monitor will fall back to a public, but limited interface that will return some information related to just publications. For mapping to Scopus authors and affiliations, an API key will be necessary.

Value	Description	Value Options	Additional Notes
query	Data Monitor filtering Query	Default: Blank (See DM Web Site for details)	If supplied, this query will filter the found set of DM records to just those that satisfy the supplied query, as opposed to all records related to the supplied publication, author, and/or affiliation IDs found.
sources	A list of sources with which to filter the found records	Default: Blank (See DM Web Site for details)	If supplied, this will be used to filter the DM records found to just those originating from the given sources.
data_types	A list of data types with which to filter found records	Options include: AUDIO, CHEMICAL_STRUCTURES, COLLECTION, DATASET, DOCUMENT, EVENT, FILE_SET, GEO_DATA, IMAGE, INTERACTIVE_RESOURCE, MODEL, OTHER, PHYSICAL_OBJECT, SEQUENCING_DATA, SLIDES, SOFTWARE_CODE, TABULA_DATA, TEXT, VIDEO, WORKFLOW Default: None	If supplied, this list of one or more of the values to the left will limit the types of data that will be returned by DM for each of the entities for which it finds data, including (if desired) publications, authors, and affiliations.
repo_types	One or both repository types from which to gather DM results	Options include: ARTICLE_BASED_REPOSITORY, NON_ARTICLE_BASED_REPOSITORY Default: None (*i.e.,* both)	Leaving this field blank will select from both types by default. Otherwise, you can select one or the other to limit the found records to just that type
aff_ids	A list of Scopus Affiliation IDs or SciVal Institution IDs with which to filter the DM found results	Valid options are blank, or a comma-delimited list of Scopus affiliation IDs and/or SciVal Institution IDs	If supplied, this list will filter the DM results to include only those which are related to one or more of the IDs listed.
journal_ids	A list of Scopus Journal IDs with which to filter the DM found results	Valid options are blank, or a comma-delimited list of Scopus Journal IDs	If supplied, this list will filter the DM results to include only those which are related to one or more of the IDs listed.
min_pub_date	Minimum Publication Date	Blank or YYYY-MM-DD	If supplied, will filter DM records
max_pub_date	Maximum Publication Date	Blank or YYYY-MM-DD	If supplied, will filter DM records
min_update_date	Minimum Update Date	Blank or YYYY-MM-DD	If supplied, will filter DM records
max_update_date	Maximum Update Date	Blank or YYYY-MM-DD	If supplied, will filter DM records

Value	Description	Value Options	Additional Notes
min_import_date	Minimum Import Date	Blank or YYYY-MM-DD	If supplied, will filter DM records
max_import_date	Maximum Import Date	Blank or YYYY-MM-DD	If supplied, will filter DM records
providers	List of Providers with which to filter DM found results	Default: Blank (See DM Web Site for details)	If supplied, will limit found DM records to just those that come from the listed providers
years	Years in which to search	Blank, or a comma-delimited list of years Default: Blank	If supplied, will limit found DM records to just those years specified
expand_pub_data	Expand Publication Related Records?	Available values are: Yes; No Default: No	If yes, will create a separate output file (and/or excel tab) that lists all DM related records for each publication found
pub_data_file	Publication Related Data File	Default: output/pub_related.txt	File in which to place DM data related to each publication
pub_data_fields	Fields to extract and output from each found DM related data record, per publication	Default: PubYear DataTypes DOIs:ExtIds[?IdType=='DOI'].Id Keywords	A full list of fields is available as a dropdown menu by clicking Edit then Add
expand_auth_data	Expand Author Related Records?	Available values are: Yes; No Default: No	If yes, will create a separate output file (and/or excel tab) that lists all DM related records for each author found
auth _data_file	Author Related Data File	Default: output/auth_related.txt	File in which to place DM data related to each author
auth _data_fields	Fields to extract and output from each found DM related data record, per author	Default: PubYear DataTypes DOIs:ExtIds[?IdType=='DOI'].Id Keywords	A full list of fields is available as a dropdown menu by clicking Edit then Add
expand_aff_data	Expand Affiliation Related Records?	Available values are: Yes; No Default: No	If yes, will create a separate output file (and/or excel tab) that lists all DM related records for each affiliation found

Value	Description	Value Options	Additional Notes
aff_data_file	Affiliation Related Data File	Default: output/aff_related.txt	File in which to place DM data related to each affiliation
aff_data_fields	Fields to extract and output from each found DM related data record, per affiliation	Default: PubYear DataTypes DOIs:ExtIds[?IdType=='DOI'].Id Keywords	A full list of fields is available as a dropdown menu by clicking Edit then Add

CoNetwork Settings

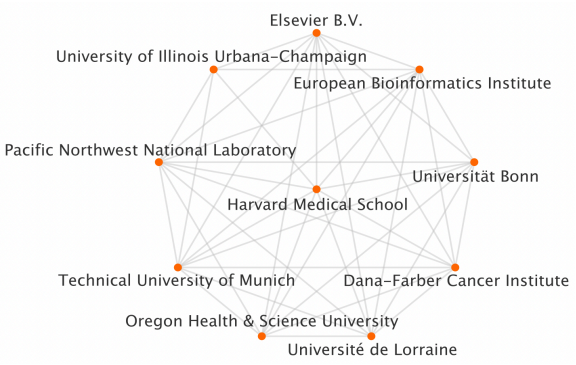
The co-networking module of the Data Fetcher is a flexible way to create co-occurrence networks with any appropriate data within the returned set of publications. If there are two or more of any sort of field in the records (*e.g.* Authors, Affiliations, Countries, Subject Areas, Keywords, IndexTerms, *etc.*) you can create a co-occurrence network. This will create an output file showing every unique pairing of those data within the publication set, along with (if requested) the list of publication IDs in which each pairing occurs.

CoNetwork Examples

The example to the right shows a co-author network. Each of the "Node" columns on the left of the table contains a Scopus Author ID, followed by the count of publication in which that pair of co-authors appear, and (optionally) the list of publication IDs themselves. Note that this function of the tool is completely handled by the Plugin system, and the **plugin_params** option under the Options area of Settings is where you specify the **CoNetworkField** on which to create the network. See the appendix on Plugins for further detail on this and other plugins, and how to create or modify the existing plugins that come with the tool.

	Node1	Node2	Count	Pubs
	9276240700	55533306700	1	85137240028
	9276240700	53164491500	1	85137240028
	9276240700	55365495700	1	85137240028
	9276240700	23101831000	1	85137240028
	9276240700	55299847600	1	85137240028
	9276240700	7403229142	1	85137240028
	9276240700	7103197296	1	85137240028
	55533306700	53164491500	1	85137240028
	55533306700	55365495700	1	85137240028
	55533306700	23101831000	1	85137240028
	55533306700	55299847600	1	85137240028
	55533306700	7403229142	1	85137240028
	55533306700	7103197296	1	85137240028
	53164491500	55365495700	1	85137240028
	53164491500	23101831000	1	85137240028
	53164491500	55299847600	1	85137240028
	53164491500	7403229142	1	85137240028
	53164491500	7103197296	1	85137240028
	55365495700	23101831000	2	85137240028, 85141926401
	55365495700	55299847600	2	85137240028, 85141926401
	55365495700	7403229142	2	85137240028, 85141926401
	55365495700	7103197296	2	85137240028, 85141926401

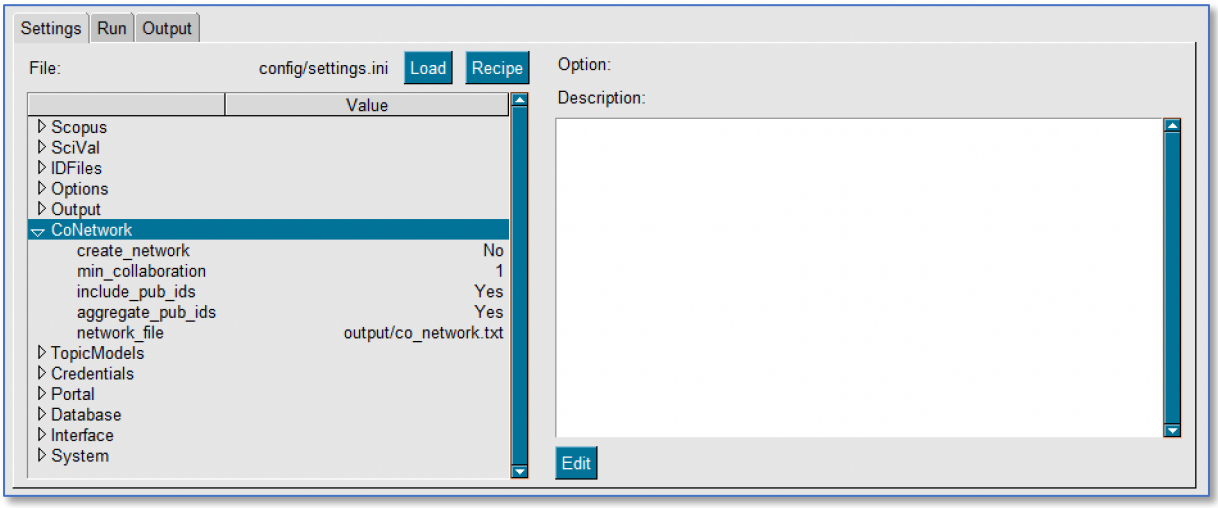
To the right is a second example of the co-networking function, this time showing co-occurences of countries appearing in the publication set. Below is a final example demonstrating an affiliation co-occurrence network visualization using the same technique.



	Node1	Node2	Count	Pubs
	Saudi Arabia	Saudi Arabia	1	85142907633
	Saudi Arabia	Netherlands	1	85142907633
	Saudi Arabia	United States	1	85142907633
	Saudi Arabia	India	1	85142907633
	Netherlands	United States	8	85137240028, 85136340733, 85139520510, 85131753911,
	Netherlands	India	1	85142907633
	Netherlands	Norway	2	85137240028, 85141926401
	Netherlands	Denmark	3	85137240028, 85141926401, 85131753911
	Netherlands	Netherlands	3	85131753911, 85134229574, 85139571715
	Netherlands	United Kingdom	2	85136340733, 85141926401
	Netherlands	Switzerland	2	85131753911, 85136340733
	Netherlands	Germany	3	85138804468, 85136340733, 85133873822
	Netherlands	Spain	1	85136340733
	Netherlands	China	1	85136340733
	Netherlands	Hong Kong	1	85136340733
	Netherlands	France	1	85136340733
	Netherlands	Australia	1	85136340733
	United States	India	1	85142907633
	United States	Norway	2	85137240028, 85141926401
	United States	Denmark	3	85137240028, 85141926401, 85131753911
	United States	United States	5	85137240028, 85136340733, 85139520510, 85141926401,
	United States	United Kingdom	2	85136340733, 85141926401

The options to create co-occurrence networks like this are below. Note that to select the field on which to create the network, the option under Options->plugin_params must be changed, as described in the plugins appendix.

CoNetwork Settings Table



Value	Description	Value Options	Additional Notes
create_network	Create a Co-Occurrence network file?	Available values are: Yes; No Default: No	If Yes, will create a file containing one row for each unique combination of fields requested in plugin options. Example: output a co-author network file with each author-author ID combination present in the data

Value	Description	Value Options	Additional Notes
min_collaboration	Minimum co-authored pubs for network?	Numerical	Minimum co-occurrences to be included in the output. Example: create a co-author network file but only for author pairs that have co-authored at least twice.
include_pub_ids	Include Publication IDs in Network file?	Available values are: Yes; No Default: Yes	If yes, the co-occurrence file will indicate which publication IDs correspond to each pairing of co-occurrences. (*e.g.,* which papers relate to each co-author pair)
aggregate_pub_ids	All co-auth pub IDs in one row?	Available values are: Yes; No Default: Yes	If yes, will aggregate paper IDs per-pair. *i.e.,* if two co-authors wrote 3 papers together, a Yes here will create 1 output line with all three paper IDs. A No here will create 3 lines in the output file, one each for each paper ID of the co-authors.
network_file	Author network file	Default: output/co_network.txt	The file can be specified by the user

Topic Models Settings

Note: The Topic Modelling feature of the Data Fetcher is experimental. A full treatment of this feature is beyond the scope of this guide, but a good starting point to better understand this module is this Wikipedia site: https://en.wikipedia.org/wiki/Latent_Dirichlet_allocation

Algorithm and Libraries

The Topic Modeler uses a Latent Dirichlet Allocation (LDA) model to cluster publications according to a probabilistic word phrase co-occurrence algorithm. Further, as LDA requires that the number of topics (clusters) be provided in advance, the tool is capable of using a Hierarchical Dirichlet Process (HDP) model to initially estimate the appropriate number of topics for the LDA to target, within limits provided below in the options. The output of this modelling is to provide a list of topics that represent the publications in the found set, and a mapping of each publication to the most likely topic(s) to which it belongs.

Regarding the Python implementation used by the Data Fetcher, there are two primary libraries used:

1. **Spacy**, for tokenization, lemmatization, part-of-speech detection, and other data preparation: https://spacy.io/

2. **Tomotopy**, to perform HDP and LDA topic modelling: https://bab2min.github.io/tomotopy

Pre-Trained Embedding Model

These libraries and their documentation will help to understand the parameters given below. Note that the Data Fetcher ships with the “medium” size Spacy model for compactness. This is appropriate for testing and to try out the feature but lacks detail and precision.

Downloading a Larger Model

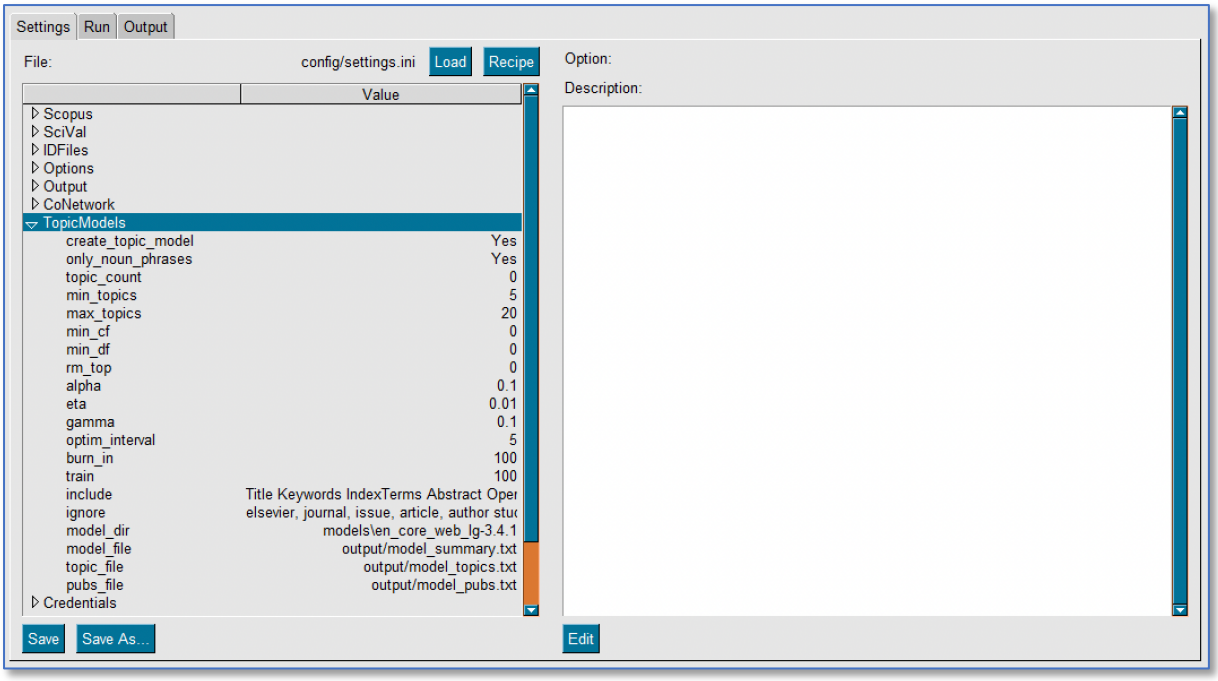
To fully explore Topic Modelling it is recommended that you download the large model from the following repository:

https://github.com/explosion/spacy-models/releases

The appropriate model for the tool would be one titled “en_core_web_lg-{version}” with the latest version. The downloadable archive is located in the section describing that model, at the bottom in the “Assets” list: look for the file ending in .tar.gz with the appropriate name.

Once you have downloaded that archive, drill into its folder hierarchy to a folder named similarly to, and containing similar files to, the default folder below. You can then save that folder anywhere you like (under the same models folder would be fine) and then use the **model_dir** setting below to direct the tool to use that model instead of the default medium model.

Topic Models Settings Table



Value	Description	Value Options	Additional Notes
create_topic_model	Create a topic modeling output file?	Available values are: Yes; No Default: No	If Yes, will create three output files (listed below) containing the process summary report, the topics generated, and the mapping of each publication to the generated topics.
only_noun_phrases	Only model Noun Phrases?	Available values are: Yes; No Default: Yes	If yes, will only consider noun phrases for the model. Otherwise, it will use all forms of speech when construction phrases.
topic_count	Topic count	Numerical Default: 0	If zero (0) the tool will use a HDP model to first approximate the number of topics to use, then proceed with LDA processing
min_topics	Minimum topics using HDP modeling	Numerical	Default value is 5
max_topics	Maximum topics using HDP modeling	Numerical	Use 0 for no limit
min_cf	Minimum corpus frequency (words)	Numerical	Default value is 0
min_df	Minimum document frequency (words)	Numerical	Default value is 0
rm_top	Remove N most frequent words	Numerical	Default value is 0
alpha	Dirichlet distribution hyperparameter for document-topic	Numerical	Default value is 0.1
eta	Dirichlet distribution hyperparameter for topic-word	Numerical	Default value is 0.01
gamma	Dirichlet Process concentration coefficient for table-topic	Numerical	Default value is 0.1
optim_interval	Optimization interval for HDP topic estimation	Numerical	Default value is 5
burn_in	Optimization burn-in iterations	Numerical	Default value is 100

Value	Description	Value Options	Additional Notes
train	Training cycles for modeling	Numerical	Default value is 100
include	Publication elements to include in model	Available values are: Title; Keywords; IndexTerms; Abstract; Body; Fingerprint	–
ignore	Topic modeling terms to ignore (comma-delimited)	User's choice	Default values are Elsevier, journal, issue, article, author study, result, datum
Ignore_file	File containing additional ignore terms	Default: config/model_ignore.txt	Multiple lines are allowed, each containing comma-delimited lists of terms to ignore. This is merged with anything listed in the **ignore** field, above.
model_dir	Topic modelling model folder	Default: models/en_core_web_md-{version}	Location of the Spacy model to use for Topic Modeling data preparation.
model_file	Topic modeling summary file	Default: output/model_summary.txt	Contains a basic summary of the model creation with cohesion metrics, *etc.*
topic_file	Topic modeling topics file	Default: output/model_topics.txt	This file contains the created topics. Each is named using the top 5 most-likely terms associated with the topic
pubs_file	Topic modeling publications file	Default: output/model_pubs.txt	This file maps each publication to the topics created, showing how likely the publication belongs to each topic listed.

Credentials Settings

In order to interact with the API server and fetch data, you are required to have an API key. This is freely available at https://dev.elsevier.com/

Institutional Tokens

Along with an API key, you may require an Institutional Token. This is another string of numbers and letters that is paired with your API key and allows access to the APIs from networks outside your institutional/campus network. The most common scenario when an Institutional Token would be necessary is if your VPN network isn't recognized by Elsevier's

servers as being part of your subscriber network. The quickest way to know if you need one is to simply try the tool with your API key and see if you get an error.

Options for Entering Credentials

The Data Fetcher has three options regarding the handling your API key:

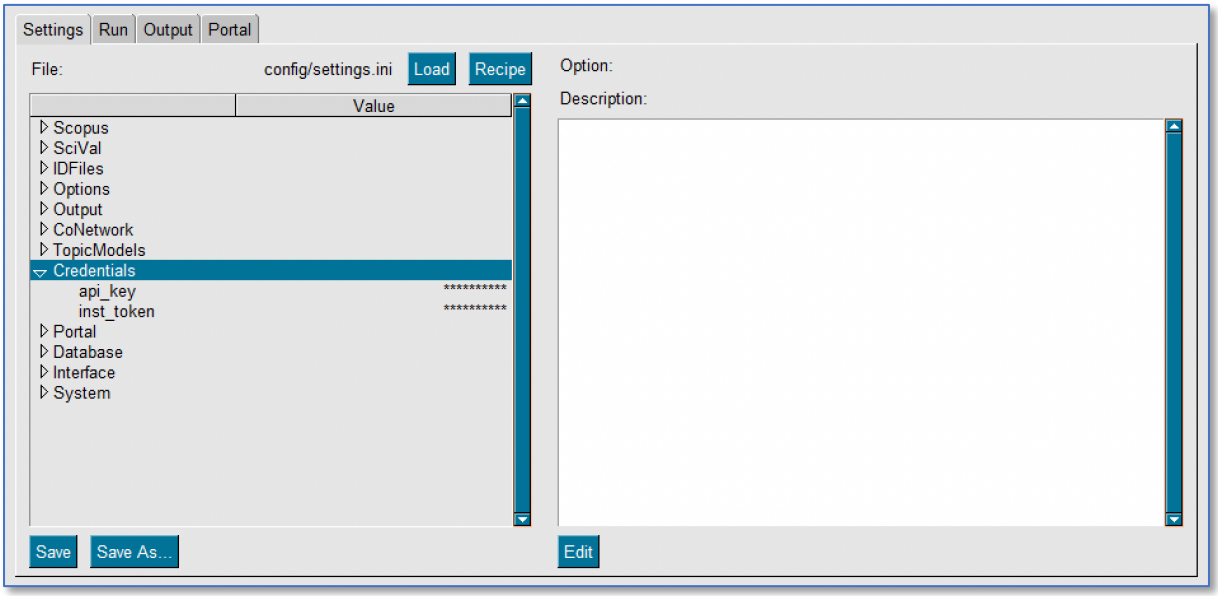
1. You can enter your credentials directly into the tool. This is the most straightforward way to do it. The downside of this approach is your API key is then part of your settings file and must be removed prior to sharing or archiving those settings in a way that it might be distributed, as this is against our Terms of Service. (*Note: This is the ONLY way of entering credentials for web portal users. More on that below*)

2. You can enter your credentials into the ".env" file within the program folder. This small file is examined for credentials along with the settings file, which means you can leave the credentials in your settings.ini file blank, and instead place them in this file. The main benefit is your settings are then separate from your credentials, and thus your settings can be freely copied and archived.

3. You can set environmental variables. This is the least-common way to use credentials, as it requires changes to your system settings and some knowledge of how to do that, but for those that wish to, you can set the variable "ELS_API_KEY" with your API key and "ELS_INST_TOKEN" (if necessary) with your institutional token, as these environmental variables will also be checked for credentials during a search.

NOTE: Only the primary process/interface of the Data Fetcher will look to these places for credentials; the portal interface will NOT do so. For users accessing the portal via the built-in web interface (described below), credentials must be entered through that web-based interface into the credentials area of the settings. It's important to NOT use the same credentials in the web portal as the main program, as if more than one search is being conducted using the same credentials, you will run into throttling errors as the same API key tries to query the server(s) too many times per second.

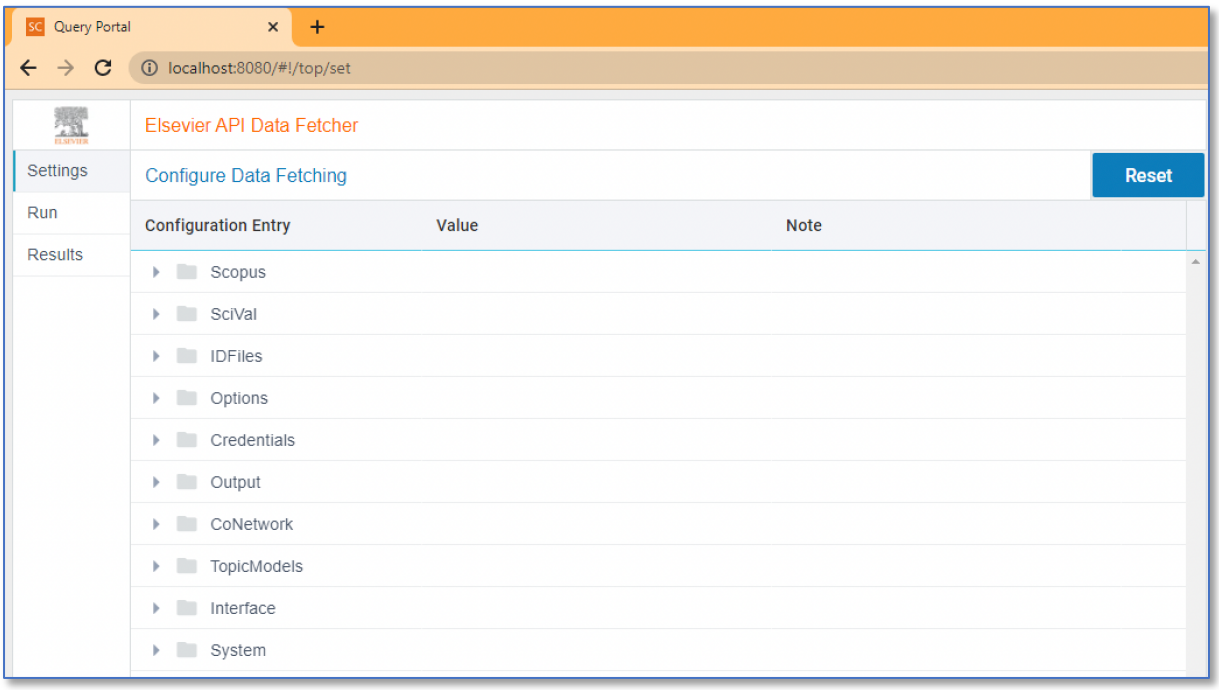
Credentials Settings Table



Value	Description	Value Options	Additional Notes
api_key	API Key	Insert your unique API Key generated via dev.elsevier.com	Options are the .env file, or setting the ELS_API_KEY environmental variable
inst_token	Institutional Token	Insert your institutional token provided by the Data Support CX team (datasupportRD@elsevier.com), if eligible	Only required in some cases. If required, can also be entered into the .env file, or set as the ELS_INST_TOKEN environmental variable/

Portal Settings

The Data Fetcher ships with a built-in web-based portal interface. This is, by default, DISABLED when you first launch the tool, and must be manually enabled for it to be accessible. This is for security, so you aren't inadvertently running a web server on your PC without your knowledge. If enabled, the portal interface will be available at http://localhost:8080 (by default) and will look like this:



Portal Use Cases

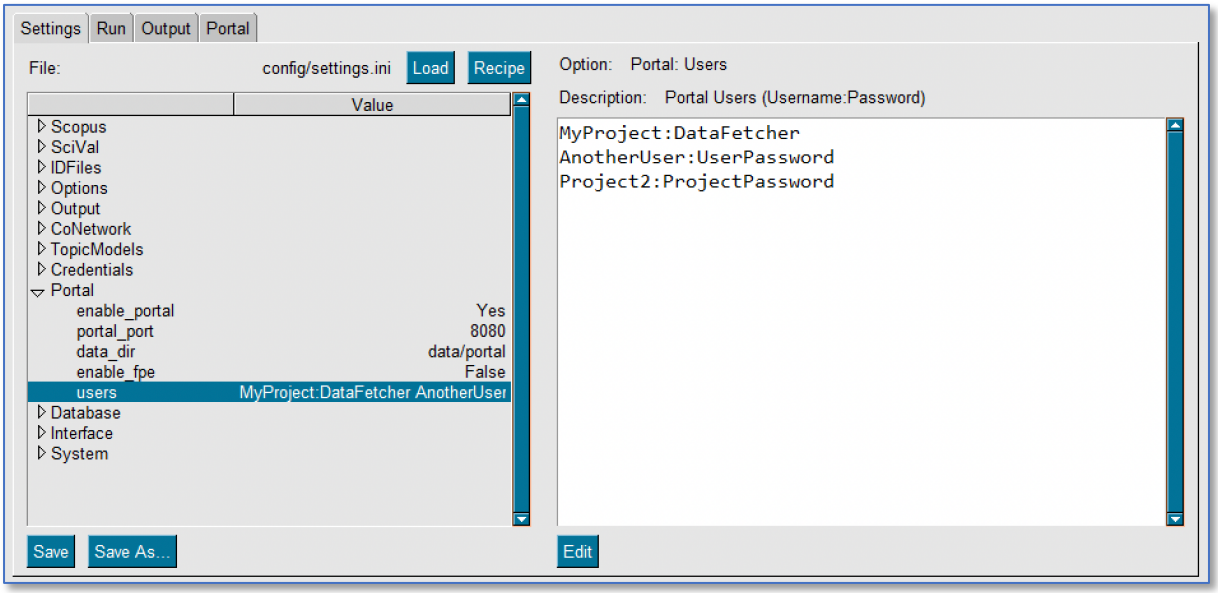
There are three main use cases for using the portal interface:

1. Access from another platform. If you do most of your work on a Mac or Linux machine but have a PC available (or run a PC virtual machine), you can run the Data Fetcher in the PC environment and then access it conveniently and remotely from your main non-Windows machine.

2. On-site workgroups. If you have a small group of librarians or researchers who all require API data, and a spare PC, you could run the Data Fetcher on that machine and have several people remotely accessing it for various projects. Note that each user would still require their own API key.

3. Multiple projects. If you have several use cases you are working on simultaneously, especially one that might be a long-running query and several smaller ones, you could initiate the longer query in the main interface, then use the web-based portal to execute smaller queries for other purposes while the main one continues to run. Note that you would need separate API keys for each project to avoid throttling issues.

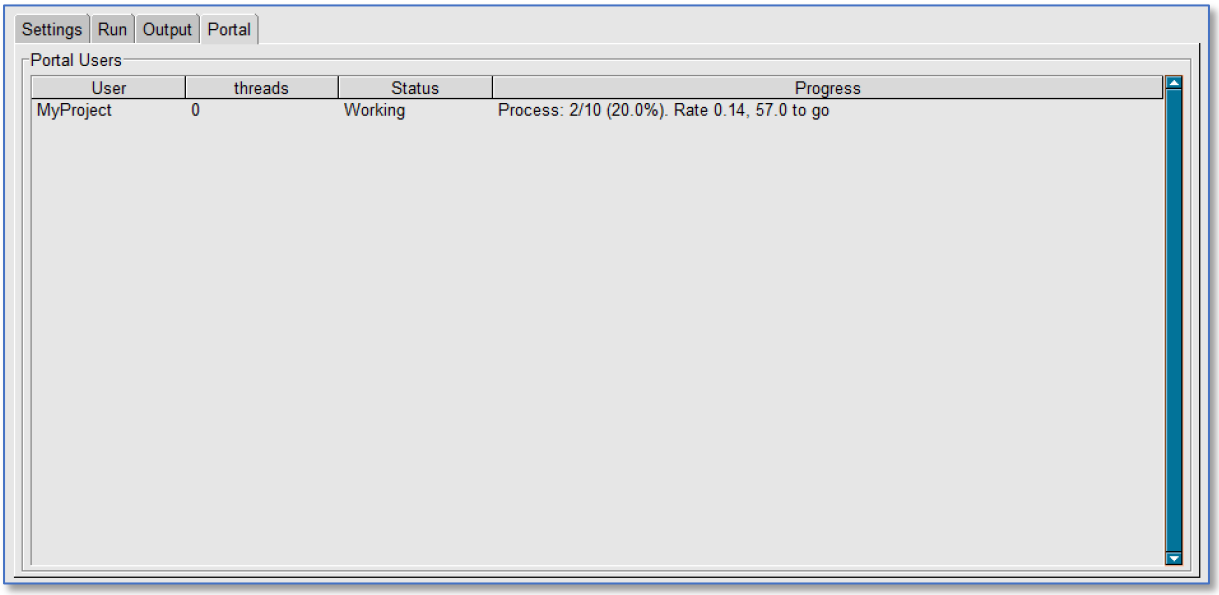
Portal Settings Table



Value	Description	Value Options	Additional Notes
enable_portal	Enable web-based portal access?	Available values are: Yes; No Default: No	This must be set to Yes for the web portal to be started. Note you must save your settings and re-start the tool to take effect.
portal_port	IP port to bind?	Numerical Default: 8080	This is the port on which to listen for web-based connections
data_directory	Web portal data directory	Default: data/portal	The root folder where the portal will store sessions, settings, uploaded id files, and output
enable_fpe	Enable Fingerprint Config on Portal? (Requires subscription)	Available values are: Yes; No Default: No	For clients with a subscription to Elsevier's Fingerprint Engine, this enables options for semantic analysis of fetched publications
users	Portal users (Username:Password)	Default: MyProject:DataFetcher	Multiple users/projects can be listed here, as in the example screenshot. Each user or project must have their own API key so as not to conflict with others

Once the portal interface is enabled, a new "Portal" tab will appear on the main program interface, next to the Output tab. This tab shows the registered users of the portal and their current status. Shows below is a view taken just after starting a portal-based search for MyProject. The screenshot below shows that MyProject has a query in progress, 20% done

with about a minute to go. This view is useful if you are preparing to shut down the tool, so as not to do so while a remote user is still processing a search.



Database Settings

By default, the Data Fetcher uses a temporary, local cache to store publication and other information locally so as not to re-query the server for the same information too often. This is initially set to about a day, as Scopus is updated daily. Any data fetched by the tool will "go stale" and expire within 24 hours, requiring that it be re-downloaded the next day, in case it has changed.

The Database facility of the Data Fetcher is still experimental and in a beta stage of development. It is designed to allow more aggressive local caching, only updating publication information when the server explicitly notes that it has been altered in Scopus. Prior to performing an Abstract Retrieval API query on a publication (the most time, resource, and quota-consuming type of query), the tool will first query the server to determine the last time that record was updated. If the date matches the locally stored last-update date in the database, the database record will be used. Otherwise, the publication will be re-downloaded, and the database will be updated. Some publications could remain unchanged for months or years, requiring only a single download during that period.

As the name implies, using the Database feature of the Data Fetcher requires that a local database be created to store the cached information. By default, a SQLite database is created in a file under the data folder of the tool. This is self-contained; it does not require any database software to be running on the PC. The downside of this approach is that database technology can introduce a significant delay when writing data to the file, which will cause initial queries to run slower. However, once saved, that technology does provide extremely fast read times. Alternatively, external databases can be used, such as Postgres, MySQL, SQL Server, Oracle, *etc.* The advantage to that approach is the performance is

typically quite fast for both reads and writes. Properly installing and configuring such a database and configuring the Data Fetcher to use it is beyond the scope of this manual. Please contact DataFetcherSupport@Elsevier.com for assistance if you are interested in trying this feature.

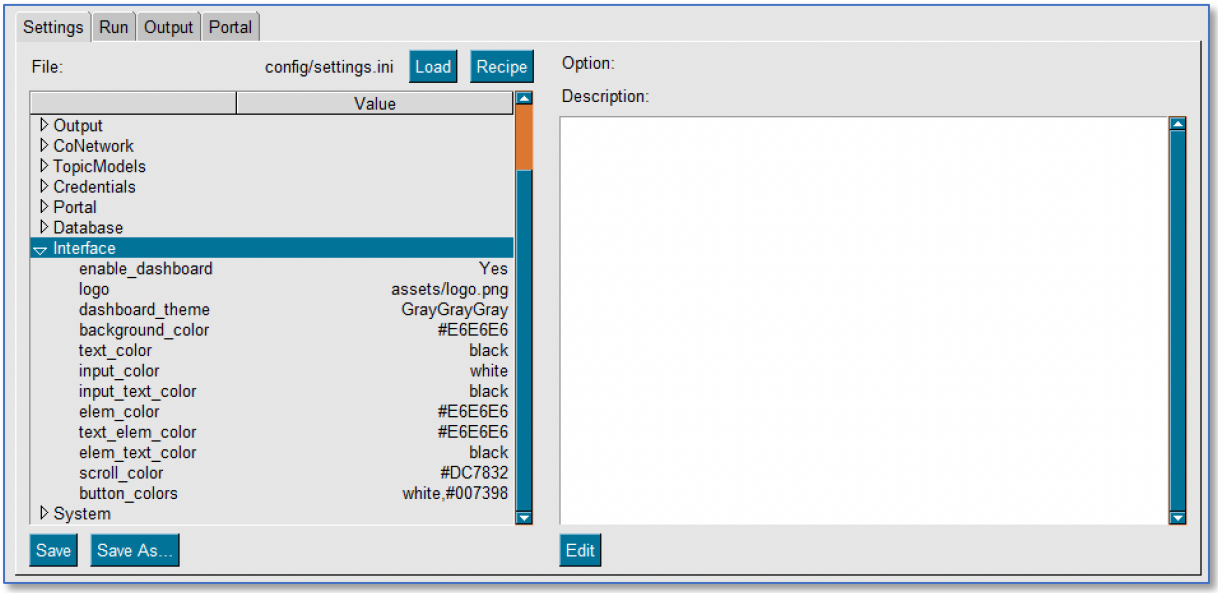
Database Settings Table

Value	Description	Value Options	Additional Notes
use_db	Enable Local Database?	Available values are: Yes; No Default: No	Experimental
pub_db	SQLAlchemy Connection URL for Pubs Database	Default: sqlite:///data/pub.db Options could include something like: *postgresql+psycopg2://fetcher:fetcher@localhost:5432/fetcher*	This is a SqlAlchemy connection string: https://www.sqlalchemy.org

Interface Settings

The Data Fetcher has a customizable interface. This allows you to alter the overall color theme of the applications, as well as select individual colors to override and adjust as you like. This would allow you to customize the application with an organization's color scheme, for instance, or simply adjust the tool to be similar to your desktop theme. You can even swap out the logo in the upper-left corner of the application, if desired, with a similarly sized .png graphics file of your choice. Note that for now, these changes only apply to the main tool interface; the web-based portal colors are unaffected.

Interface Settings Table



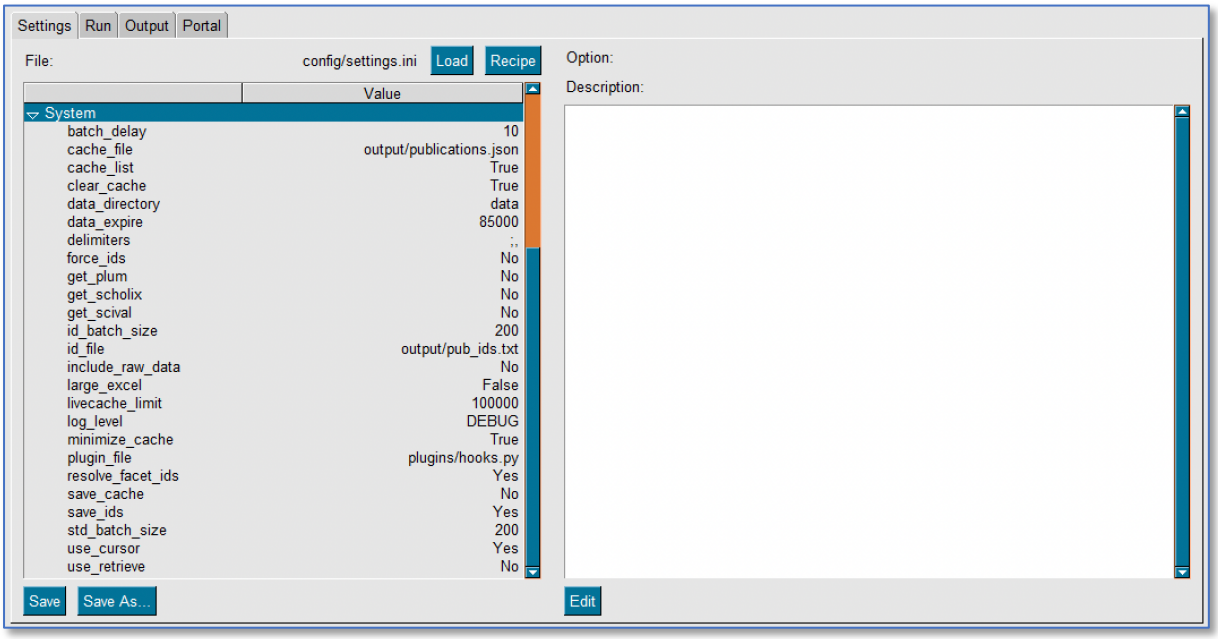
Value	Description	Value Options	Additional Notes
enable_dashboard	Enable GUI-based dashboard? Disable for command-line only	Available values are: Yes; No Default: No	This enabled the main tool window. This should only be disabled if the only use of the tool is for batch updates.
logo	Logo file (PNG)	Default: assets/logo.png	You can replace this file with another of the same size and type if you would like.
dashboard_theme	Changes the color theme of the application	Default value is GrayGrayGray	60+ options, requires a restart after changing
background_color	Background color	Default: #E6E6E6	This can be a hexadecimal value or standard web color
text_color	Non-input text color	Default: black	This can be a hexadecimal value or standard web color
input_color	Input background color	Default: white	This can be a hexadecimal value or standard web color
input_text_color	Input text color	Default: black	This can be a hexadecimal value or standard web color
elem_color	Element background color (e.g., tabs)	Default: #E6E6E6	This can be a hexadecimal value or standard web color
text_elem_color	Text element text color (e.g., tables)	Default: #E6E6E6	This can be a hexadecimal value or standard web color

elem_text_color	Element text color (e.g., tabs, tables)	Default: black	This can be a hexadecimal value or standard web color
scroll_color	Scrollbar color	Default: #DC7832	This can be a hexadecimal value or standard web color
button_colors	Button colors (text, background)	Default: white, #007398	This is a two-color combo, as text, background

System Settings

The Data Fetcher has several system settings that generally won't require changing for most users. However, they can be important on a case-by-case basis, and so are included as adjustable parameters. In particular, if JSON output is desired from the tool, this is where that can be configured. Note, however, that a JSON-format output file can only be created via the main interface and query, and not through a portal-based query.

System Settings Table



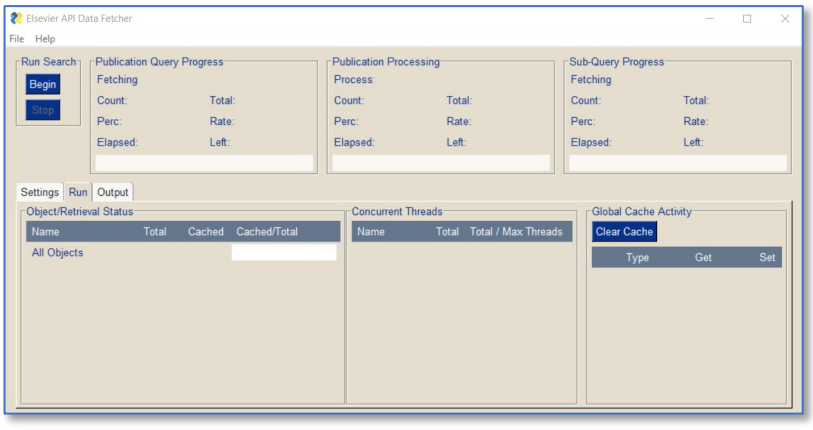
Value	Description	Value Options	Additional Notes
batch_delay	Delay in seconds in which to batch IDs for queries	Default: 10 5-10 is generally where you'd want this	While processing results from the initial query or ID list, the tool will spend at most this many seconds collecting IDs to submit to other batch APIs (such as SciVal).
cache_file	JSON Output file (if enabled)	Default: output/publications.json	If **save_cache** is enabled, this file will contain a JSON

Value	Description	Value Options	Additional Notes
			representation of each publication returned by a fetch
cache_list	Format JSON cache file as a single, large list?	Available values are: Yes; No Default: Yes	For small-to-medium datasets, this can be convenient, as it causes the JSON cache file to be one large array of values (which can then be imported easily).
clear_cache	Clear JSON cache file before each new query?	Available values are: Yes; No Default: Yes	This option resets the JSON cache file on each execution of a search. Note this only applies to the main interface; the web portal cannot generate a JSON output file.
data_directory	Data cache directory	Default: data	This is the root folder for the tool's data cache
data_expire	Data expiration (in seconds)	Default: 85000	This is the number of seconds to preserve a locally cached result from the API server(s) before marking it stale and re-requesting the corresponding data.
delimiters	Delimiters for outputting list-based data	Default: ;,	If column data in the output is a list of values, this indicates a hierarchy of characters to delimit that list.
force_ids	Require cursor-based search?	Available values are: Yes; No Default: No	If enabled, this requires the tool to first download the list of publication IDs that will satisfy a given query, then to proceed to download the publication data for that set of IDs. The tool often does this in any case; this makes it do so in all cases.
get_plum	Always retrieve Plum metrics?	Available values are: Yes; No Default: No	Primarily for debugging. If enabled, this will download and cache PlumX metrics for all publications, every time
get_scholix	Always retrieve Scholix links?	Available values are: Yes; No Default: No	Primarily for debugging. If enabled, this will download and cache Scholix metrics for all publications, every time
get_scival	Always retrieve SciVal Data?	Available values are: Yes; No	Primarily for debugging. If enabled, this will download and

Value	Description	Value Options	Additional Notes
		Default: No	cache SciVal metrics for all publications, every time
id_batch_size	Number of cursor-based IDs to request per query	Default: 200	This should always be 200
id_file	Publication ID file (if enabled)	Default: output/pub_ids.txt	This file contains a cross-reference of IDs for each publication downloaded
include_raw_data	Include Raw server response in JSON cache file?	Available values are: Yes; No Default: No	Primarily for debugging. This will include the raw JSON data returned by the server(s) for each publication record save to the JSON cache file (**save_cache** must be enabled)
large_excel	Support larger-sized Excel workbooks?	Available values are: Yes; No Default: No	Disables table creation and formatting in the tabs, but does allow a more memory-efficient writing process for large data
livecache_limit	Live (per-run) in-memory object limit	Default: 100,000	This specifies the size of the in-memory cache of all objects created during a fetch. This can be decreased if memory becomes a problem, or increased if sufficient memory exists, to increase overall performance.
log_level	Verbosity/Level of Logging Output	Default: INFO	Log Level options include CRITICAL, ERROR, WARNING, INFO, DEBUG. If contacting Support, you would want to set this to DEBUG and send the resulting Elsevier.log file along.
minimize_cache	Minimize JSON Cache file size by removing Null entries?	Available values are: Yes; No Default: Yes	If a JSON cache file is being created, this option will remove empty nodes from the structure, resulting in a much smaller file
plugin_file	File containing plugin extensions for the tool	Default: plugins/hooks.py	Determines which file to load for Python-based plugin extensions to the Data Fetcher. See the Plugins appendix for details
resolve_facets_ids	Resolve Author and Affiliation names in Facets?	Available values are: Yes; No Default: Yes	Normally, when selecting Author or Affiliation ID facets, only the ID numbers are saved. This open

Value	Description	Value Options	Additional Notes
			automatically fetches the names corresponding to those IDs.
save_cache	Create a JSON-based Cache file of downloaded Publications	Available values are: Yes; No Default: No	This option will save a JSON-formatted file containing all retrieved publication data.
save_ids	Create a separate ID file with just Publication IDs?	Available values are: Yes; No Default: Yes	This saves a file with a cross-reference of all Publication IDs in the retrieved set. This should generally always be Yes.
std_batch_size	Batch size for STANDARD view (Should be 200 for subscribers)	Default: 200	This should always be 200
use_cursor	Enable cursor-based searches (Should be Yes for subscribers)	Available values are: Yes; No Default: Yes	This enables cursor-based back-end searches on Scopus.com. This option should always be Yes
use_retrieve	Force Abstract Retrieval API call for each publication?	Available values are: Yes; No Default: No	Forces the tool to use only the Abstract retrieval API to fetch Scopus publication data. This should always be No.

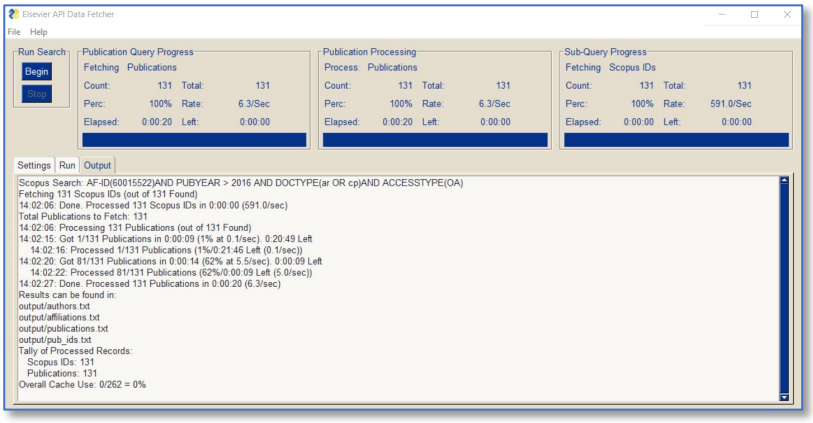
Explanation of Run Tab



The Run tab is a summary of the Output tab. You can see the Run and Output tabs update in real time as a request runs. The Object Retrieval Status box is an overall indication of entities that the tool is retrieving and providing data for. The Concurrent Threads box is a glimpse into what the tool is doing to gather data "behind the scenes". The Cache Activity box is where you can clear the cache. It is also useful to check the Cache Activity box if you suspect the tool may have stalled out- if the numbers are still climbing in the Cache Activity box, the tool is functioning as expected.

Explanation of Output Tab

During a live request, the Output tab displays a live update of the status of the request. It contains time stamps and estimates during the live update. Once the request is complete, the box contains a tally of time stamps, processed records, status of quota, cache use, and where to find the Output files (in the Output folder!)



Using Plugins

The Data Fetcher comes with a Python-based plugin architecture that allows for flexibly expanding the tool's processing and output.

Using the default Plugins

A set of default plugins ships with the Data Fetcher, that enable capabilities including:

1. **Co-Occurrence networks** – This function is handled entirely by the plugin system, in part to demonstrate how to use plugins to analyze and output custom data related to retrieved publication information.

2. **Additional Fields** – This demonstrates how to not only create additional fields available in the output files, but how to modify the field menus in the tool to display those custom fields for users to select when configuring their output file structures.

3. **Additional Output Files** – The plugin architecture enables additional output files that alter or add to the existing, default set of output files. For instance, an All Science Journal Classification (ASJC) mapping file can be created, mapping each retrieved publication to the subject areas it falls within, using the default plugins.

Plugin Options

Plugins have their own Options in the Settings->Options area of the tool, under the heading of **plugin_params**. This holds any number of user-configurable options that will be made available to the plugin scripts.

Plugin Option Format

Each plugin option is in the form of **Name:Type:Value** whereby:

1. **Name** is the Option Name that will be made available to the plugin script(s)

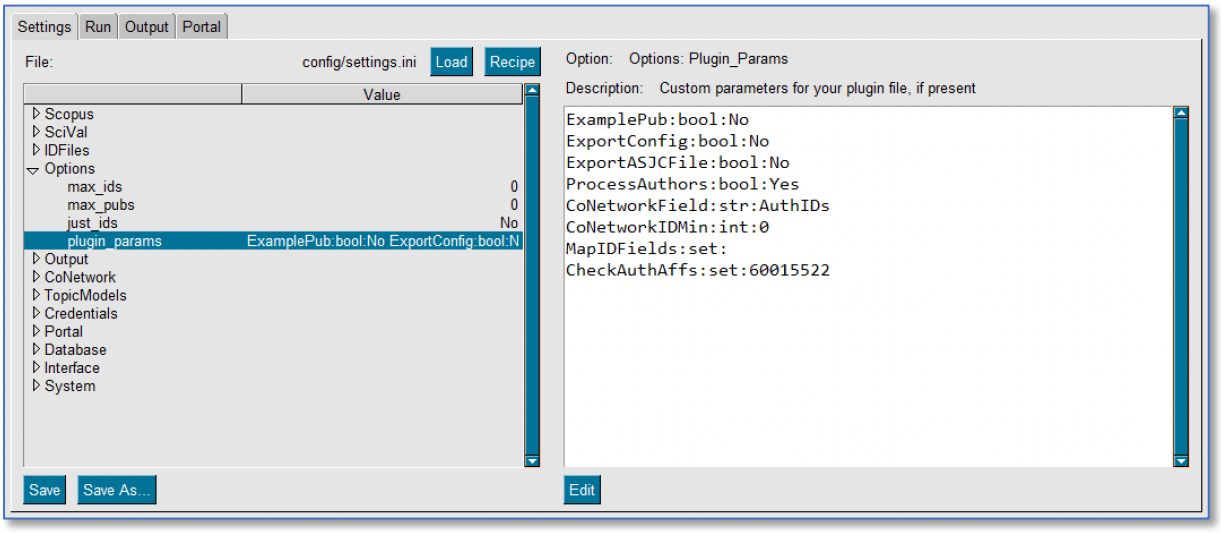
2. **Type** is the type of data the plugin will receive from that option, and will be interpreted thusly:

    - bool – this is a Boolean (True/False) value
    - str – this is a string (normal text) value
    - int – this is an integer (numeric) value
    - float – this is a floating-point number
    - set – this is a set (an unordered list of unique items)
    - list – this is a list (an ordered list of items (repeats are allowed))

3. **Value** is the value to pass along to the plugin script. For sets and lists, this can be a list of comma-delimited values, such as **Title,Abstract,Keywords** otherwise it should be a single value of the type indicated in the Type area for that plugin option.

As per the description above, each option below that ships with the Data Fetcher has a name, followed by a type, and finally a value, all separated by a colon (:). While the options below are the default ones that come “out of the box” with the Data Fetcher, this area is meant to be expanded upon and adjusted via customizations of the plugins available, as they are modified or enhanced.

Default Plugin Options Table



Value	Description	Value Options	Additional Notes
ExamplePub	Boolean: Export example publication JSON file	Available values are: Yes; No Default: No	If enabled, this will create a JSON file that fully expands the first publication retrieved by the tool during a fetch. This shows the structure of the data passed to

Value	Description	Value Options	Additional Notes
			plugins during plugin development.
ExportConfig	Boolean: Export the tool's configuration as a JSON file at the start of each run	Available values are: Yes; No Default: No	If enabled, will write out the tool configuration on each run. This shows the structure and contents of the configuration that is available to plugins.
ExportASJCFile	Boolean: Export a file that maps publications to ASJC codes	Available values are: Yes; No Default: No	If enabled, this creates a mapping file, with one row per publication-ASJC code combination. *i.e.,* if a publication has three ASJC codes, this will produce three lines in the file
ExportFingerprint	Boolean: Export a file with fingerprint concepts listed for each publication (requires subscription to Elsevier's Fingerprint Engine)	Available values are: Yes; No Default: No	If enabled, this creates a mapping file, one row per publication-concept combination. *i.e.,* if a publication has 20 concepts, this will produce 20 lines in the file
ProcessAuthors	Boolean: Enables/Disables author processing in the plugin	Available values are: Yes; No Default: Yes	This option enables author processing within the plugin. Generally, should be remain Yes
CoNetworkField	String: This indicates the field used to create a co-occurrence network	Options are any valid publication field. Default: AuthIDs	The field indicated here will be used to create the co-network file. Note this can follow JMESpath syntax, so something like this would work: Affiliations[*].Name Several examples are listed in **Settings->Output->field_aliases**
CoNetworkIDMin	Integer: Minimum number of co-network id nodes that must be in the provided ID list (if given)	Options are 0 or positive integer. Default: 0 (all)	If set to something like 2, this would require that both Node1 and Node2 or the collaboration must appear in the ID list provided (use_idfile must be true as that's the list that's examined)
MapIDFields	set: One or more fields in the publication record with which to create a separate mapping file (*i.e.,* AuthID->pubID, AffID->PubID, SDG->PubID, *etc.*)	Default: Blank comma-delimited list of any valid JMESpath publication fields (e.g. Authors[*].EID). Aliases are allowed.	Any fields listed here will generate separate mapping output files, one for each field, that maps the field requested to the corresponding publication ID. The file will be named map_{field}.txt

Value	Description	Value Options	Additional Notes
CheckAuthAffs	Set: comma-delimited list of Affiliation IDs	List of valid IDs, separated by commas. Default: 60015522 (Elsevier's Aff ID)	This option drives the value of the AffAuth custom plugin author field. If used, this field will indicate whether each author listed in the author expansion file lists any of the indicated Aff IDs on that particular paper. In the default example, this would highlight Elsevier authors in the default search's list of Elsevier publications.

Customizing Plugins

The Data Fetcher plugin architecture uses a set of 'hooks' or points of interception during processing where custom code can be inserted.

Plugin Files

Plugin files are located in the plugins folder under the main installation folder of the tool. Within this folder are three items to examine:

1. **hooks.py** – This is the default plugin file that ships with the tool. You can modify this file, or copy it to a new file and direct the tool to load your file by adjusting the following setting: **Settings->System->plugin_file**

2. **extensions** – This is a sub-folder containing library functions useful in creating plugins. In particular, the file **utility.py** in the extensions folder is used by the main **hooks.py** file to perform various functions.

3. **PluginManager.py** – This is a copy of the Python module within the Data Fetcher that loads the plugin file and manages the execution of the functions within it. You can examine this file to understand the interaction of the main system with the plugins, and you can execute the file to perform a basic test of the existing plugins or modifications you have made to them. There are several prerequisites to using this file successfully:

    a. You must have Python installed (3.10+ matches the tool). In addition, running the script may generate ModuleNotFound errors. You will have to install the noted modules. As of this writing, **orjson** and **jmespath** are required at a minimum.

    b. Read through the comments in PluginManager.py, as you must first run the tool configured to produce test output compatible with it.

Hook Functions and Parameters

As processing of a query or ID file progresses, at certain times the Plugin Manager will determine if a hook function has been registered for that point in the workflow. If so, it will be called with one or more parameters, including, at a minimum:

1. **state** – This is an initially-empty convenience data structure that plugins can use to store data between hook invocations. Examples of using the state dictionary are in hooks.py

2. **lock** – The Plugin Manager maintains a thread lock it passes to each hook. The Data Fetcher uses multithreading for improved API throughput. Importantly, hook invocations are **NOT** inherently thread-safe. Modifying local variables and passed data structures is safe. However, modifying the state dictionary, or performing file operations, should be handled inside the lock context manager. Examples of lock usage are in the hooks.py file.

3. **data** – Depending on the hook, this data will be a publication, or author, or affiliation, etc. See the comments for each hook to determine whether the data structure is read-only or alterable within the hook function.

How to perform data analysis via Microsoft Excel

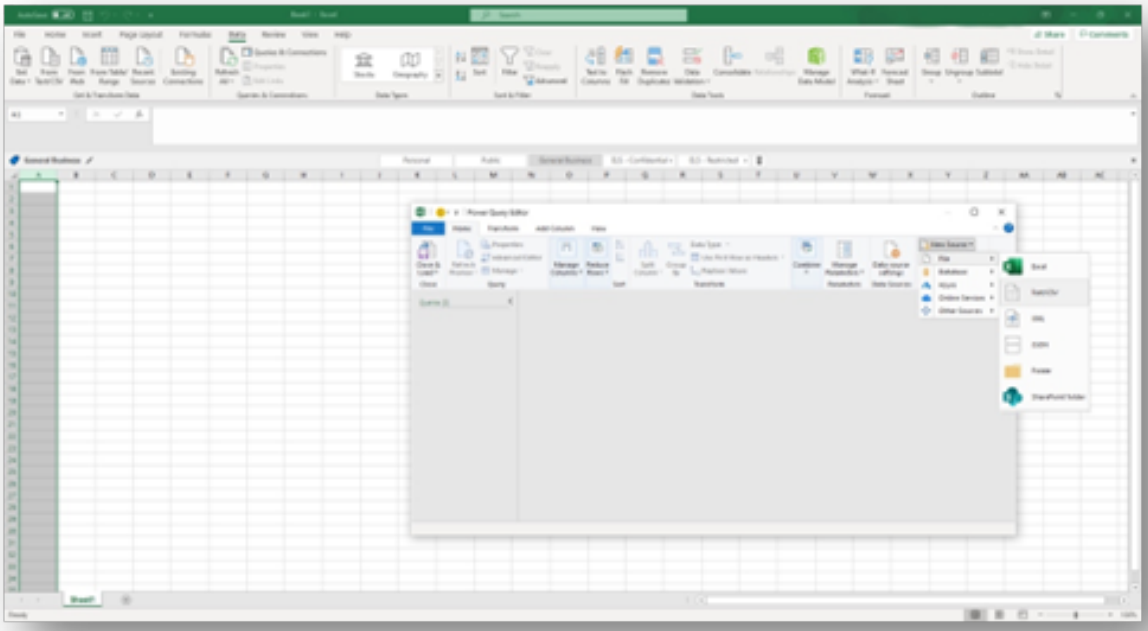
Note: The Data Fetcher has the ability to directly export Excel-formatted workbooks created from each executed job. The instructions in this section are intended for situations in which you are importing the data into an existing workbook, or other use cases in which the default workbook export file isn't appropriate.

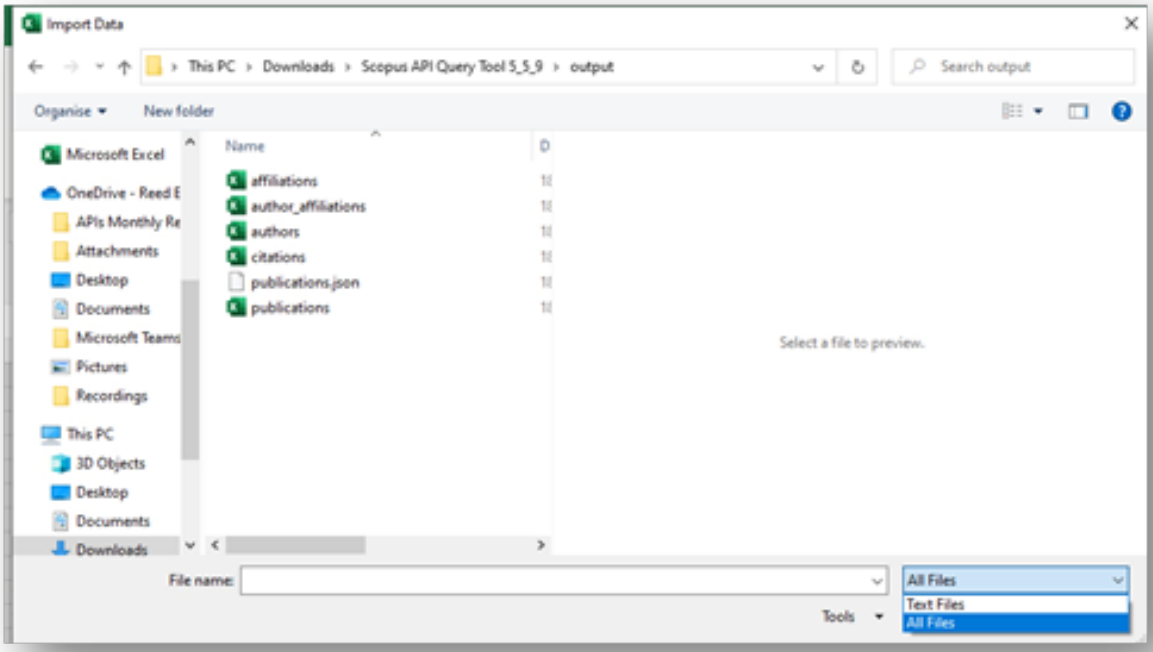
Importing text file output data into existing Workbooks

To see the data output from the last fetch request, navigate to the Output folder within the Data Fetcher folder on your Desktop. The files indicated in the Settings->Output area of the Settings tab will be available there by default.

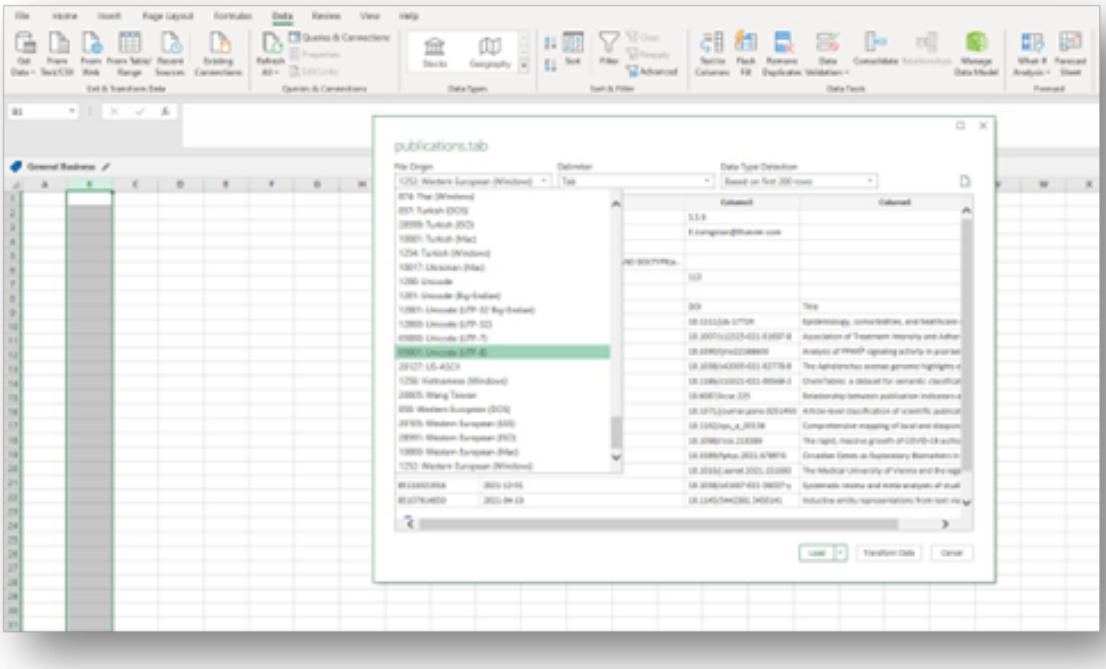
The output files are encoded using Unicode UTF-8. This is the native encoding of the Scopus database and the communications that take place between any script and the Elsevier APIs. Because Excel may not use this encoding (Especially on PCs, where it defaults to Window's own encoding), we may need to manually override Excel's encoding. This can be done by taking the following steps to customize Excel to open API files in a way that all characters appear correctly.

To begin your import, navigate to the Data heading, click Get Data, then Power Query Editor, selecting New Source and then Text/CSV. Then, navigate to the Data Fetcher > Output folders, change file type to All Files, and import the Publications file as an example.





Change the File Origin to UTF-8 and click OK.



The encoding should be displaying correctly now, but there is one more adjustment to make. For columns with data comprised of numbers, click the 123 icon in the column header and click Text followed by Replace current. Now, all number values should be displayed as left-aligned, and the column header should show an ABC icon.

At this point, you can click Close and Load, which turns the spreadsheet into a table automatically. Rename the file once processed and save it in a secure location, outside of the Output folder.

Note: You will need to adjust the formatting in this manner each time you generate new output files, and the output files will automatically update when a new request is run. It is very important that you save as and rename the output files you want to because rerunning the data fetcher will overwrite the output files.

Data Fetcher Local Caching

While you generally don't need to do anything to the Data folder, it is good to know what it's for. The Data folder is where the program stores its local cache, web session information (if enabled), and local publication database (if enabled). Do not clear the cache if you don't have to so requests run more quickly, and you stay within your quota limits. The cache comes pre-set to expire every 24 hours as that is how frequently the Scopus database is updated, though you can set that to any desired duration.

API Key Quotas

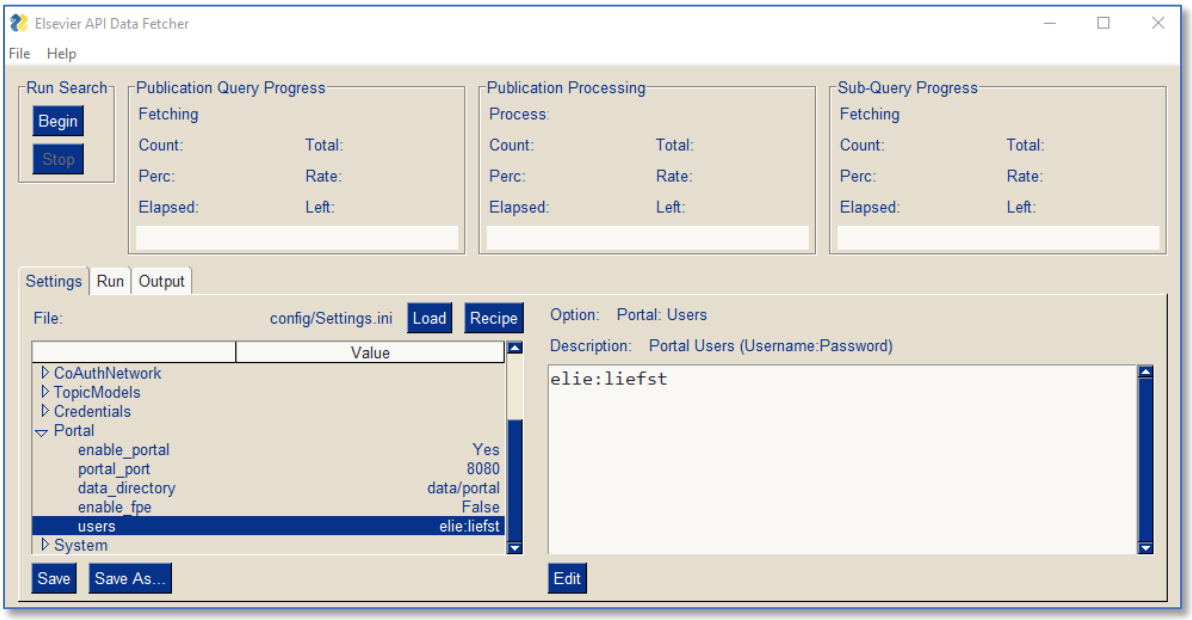
Quotas limit the number of requests a given API key can make to a particular Scopus or other API endpoint and are reset every seven days. You can view the quotas and throttling rates of our suite of APIs on https://dev.elsevier.com/api_key_settings.html. Quota limits are unique to each API, there is not a single global setting for a given API Key. If you find that the quota of a particular API is not enough for your project needs, please write API Support with:

- your API Key
- full institution name
- a brief description of your use case
- API name and requested quota increase

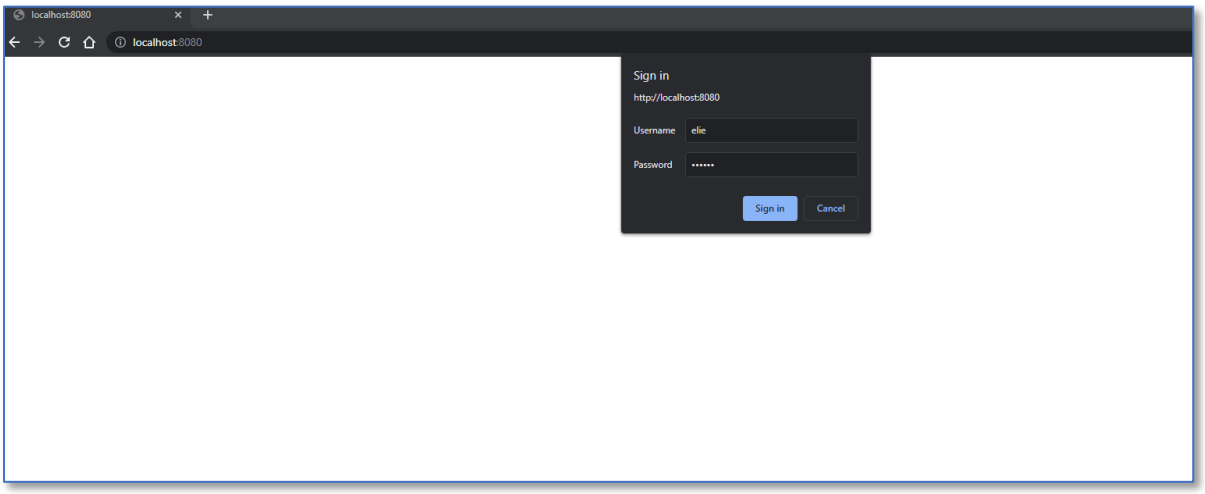
Accessing the Data Fetcher via Browser

Users have the ability to use the tool through their browser, if that capability is enabled in the Data Fetcher settings. This can be useful for a variety of use cases, such as a group of users who work on the same project and share a set of credentials. It can also be useful for Mac users who want to use the tool on a machine that is not natively supported.

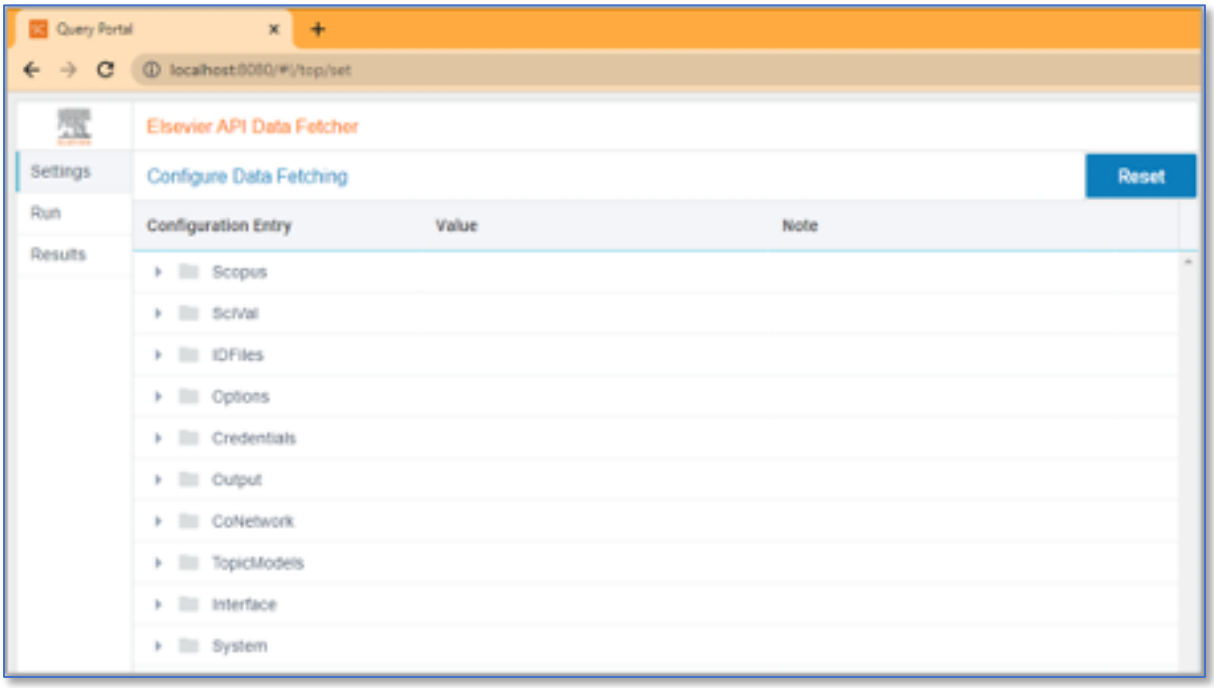
You can turn on the browser functionality by navigating to the Portal settings and changing the default value from No to Yes on the enable_portal setting. Then, you will need to go to the users setting and set a username and password for yourself, such as Jane:3lsevi3r. Make sure to save your settings and then close and restart the application.



Once your application has restarted, you can navigate to your browser and type in “localhost:8080”. This will trigger a log-in screen, and you will enter the username and password you specified in the user setting.



Once you are logged in, the browser should show the Settings, Run, and Results tab.



The browser contains a streamlined collection of settings. If you refer to the Settings section of this manual, settings available in the browser extension are bolded.

Useful Resources and Tips

1. **Metadata Field Mappings –** Read about data tags for Scopus APIs here and here

2. **Constructing Queries –** You can copy and paste the advanced search version of searches you make on Scopus.com into the Data Fetcher. This is a fast way to construct and run requests in the tool. For quality control, you can even run a

comparison between the number of results returned in Scopus.com and the number returned via the API-powered Data Fetcher.

3. **Prepping before large queries –** Limit the number of publications in "Options" to a small number (e.g., 10) the first time you run a query to make sure that the outputs are exactly as you need them before running large queries

Who to contact for support

As noted previously, this is pre-production software provided “as is”. The Data Fetcher team will answer access-related queries and associated permissions adjustments for subscribing users. Our team will answer questions, suggestions, or ideas **as time allows.**

Our email address is datafetchersupport@elsevier.com.