Google Images Download

May 21, 2019

Contents

Link to GitHub repo

CHAPTER 1

Summary

This is a command line python program to search keywords/key-phrases on Google Images and optionally download images to your computer. You can also invoke this script from another python file.

This is a small and ready-to-run program. No dependencies are required to be installed if you would only want to download up to 100 images per keyword. If you would want **more than 100 images** per keyword, then you would need to install `Selenium` library along with `chromedriver`. Detailed instructions in the troubleshooting section.

CHAPTER 2

Compatibility

This program is compatible with both the versions of python - 2.x and 3.x (recommended). It is a download-and-run program with no changes to the file. You will just have to specify parameters through the command line.

CHAPTER 3

Installation

The guide provides detailed instructions on how to install the library.

3.1 Installation

Link to Documentation Homepage

You can use **one of the below methods** to download and use this repository.

3.1.1 Install using pip

```
$ pip install google_images_download
```

3.1.2 Manually install using CLI

```
$ git clone https://github.com/hardikvasa/google-images-download.git
$ cd google-images-download && sudo python setup.py install
```

3.1.3 Manually install using UI

Go to the repo on github ==> Click on 'Clone or Download' ==> Click on 'Download ZIP' and save it on your local disk.

CHAPTER 4

Usage

The following section provides details on using the library - from CLI or by standard imports.

4.1 Usage

Link to GitHub repo

Link to Documentation Homepage

4.1.1 Using the library from Command Line Interface

If installed via pip or using CLI, use the following command:

```
$ googleimagesdownload [Arguments...]
```

If downloaded via the UI, unzip the file downloaded, go to the 'google_images_download' directory and use one of the below commands:

```
$ python3 google_images_download.py [Arguments...]
OR
$ python google_images_download.py [Arguments...]
```

4.1.2 Using the library from another python file

If you would want to use this library from another python file, you could use it as shown below:

```
from google_images_download import google_images_download

response = google_images_download.googleimagesdownload()
absolute_image_paths = response.download({<Arguments...>})
```

CHAPTER 5

Arguments

This section provides all the arguments/parameters/options you can provide to this library.

5.1 Input Arguments

Link to GitHub repo

Link to Documentation Homepage

Argument	Short hand	Description
config_file	cf	You can pass the arguments inside a config file. This is an alternative to passing arguments on the command line directly. Please refer to the config file format below • If 'config_file' argument is present, the program will use the config file and command line arguments will be discarded • Config file can only be in **JSON** format • Please refrain from passing invalid arguments from config file. Refer to the below arguments list

Continued on next page

Table 1 – continued from previous page

Argument	Short hand	Description
keywords	k	Denotes the keywords/key phrases you want to search for. For more than one keywords, wrap it in single quotes. Tips: • If you simply type the keyword, Google will best try to match it • If you want to search for exact phrase, you can wrap the keywords in double quotes (“”) • If you want to search to contain either of the words provided, use **OR** between the words. • If you want to explicitly not want a specific word use a minus sign before the word (-)
keywords_from_file	kf	Denotes the file name from where you would want to import the keywords. Add one keyword per line. Blank/Empty lines are truncated automatically. Only file types ‘.txt’ or ‘.csv’ are allowed.
prefix_keywords	pk	Denotes additional words added before main keyword while making the search query. The final search query would be: <prefix keyword> <keyword> So, for example, if the keyword is ‘car’ and prefix_keyword is ‘red,yellow,blue’, it will search and download images for ‘red car’, ‘yellow car’ and ‘blue car’ individually
suffix_keywords	sk	Denotes additional words added after main keyword while making the search query. The final search query would be: <keyword> <suffix keyword> So, for example, if the keyword is ‘car’ and suffix_keyword is ‘red,yellow,blue’, it will search and download images for ‘car red’, ‘car yellow’ and ‘car blue’ individually

Continued on next page

Table 1 – continued from previous page

Argument	Short hand	Description
limit	l	Denotes number of images that you want to download. You can specify any integer value here. It will try and get all the images that it finds in the google image search page. If this value is not specified, it defaults to 100. **Note**: In case of occasional errors while downloading images, you could get less than 100 (if the limit is set to 100)
related_images	ri	This argument downloads a ton of images related to the keyword you provided. Google Images page returns list of related keywords to the keyword you have mentioned in the query. This tool downloads images from each of those related keywords based on the limit you have mentioned in your query This argument does not take any value. Just add '–related_images' or '-ri' in your query. **Note:** This argument can download hundreds or thousands of additional images so please use this carefully.
format	f	Denotes the format/extension of the image that you want to download. *Possible values: jpg, gif, png, bmp, svg, webp, ico, raw*
color	co	Denotes the color filter that you want to apply to the images. *Possible values: red, orange, yellow, green, teal, blue, purple, pink, white, gray, black, brown*
color_type	ct	Denotes the color type you want to apply to the images. *Possible values: full-color, black-and-white, transparent*

Continued on next page

Table 1 – continued from previous page

Argument	Short hand	Description
usage_rights	r	Denotes the usage rights/licence under which the image is classified. *Possible values:* • *labeled-for-reuse-with-modifications*, • *labeled-for-reuse*, • *labeled-for-noncommercial-reuse-with-modification*, • *labeled-for-nocommercial-reuse*
size	s	Denotes the relative size of the image to be downloaded. *Possible values: large, medium, icon, >400*300, >640*480, >800*600, >1024*768, >2MP, >4MP, >6MP, >8MP, >10MP, >12MP, >15MP, >20MP, >40MP, >70MP*
exact_size	es	You can specify the exact size/resolution of the images This value of this argument can be specified as `<integer, integer>` where the fist integer stands for width of the image and the second integer stands for the height of the image. For example, `-es 1024,786` **Note**: You cannot specify both 'size' and 'exact_size' arguments in the same query. You can only give one of them.
aspect_ratio	a	Denotes the aspect ratio of images to download. *Possible values: tall, square, wide, panoramic*
type	t	Denotes the type of image to be downloaded. *Possible values: face, photo, clip-art, line-drawing, animated*
time	w	Denotes the time the image was uploaded/indexed. *Possible values: past-24-hours, past-7-days, past-month, past-year*
time_range	wr	Denotes the time range for which you want to search the images The value of this parameter should be in the following format '{"time_min":"MM/DD/YYYY","time_max":"MM/DD

Continued on next page

Table 1 – continued from previous page

Argument	Short hand	Description
delay	d	Time to wait between downloading two images Time is to be specified in seconds. But you can have sub-second times by using decimal points.
url	u	Allows you search by image when you have the URL from the Google Images page. It downloads images from the google images link provided If you are searching an image on the browser google images page, simply grab the browser URL and paste it in this parameter It will download all the images seen on that page.
single_image	x	Allows you to download one image if the complete (absolute) URL of the image is provided
output_directory	o	Allows you specify the main directory name in which the images are downloaded. If not specified, it will default to 'downloads' directory. This directory is located in the path from where you run this code The directory structure would look like: `<output_directory><image_directory><i`
image_directory	i	This lets you specify a directory inside of the main directory (output_directory) in which the images will be saved If not specified, it will default to the name of the keyword. The directory structure would look like: `<output_directory><image_directory><i`
no_directory	n	This option allows you download images directly in the main directory (output_directory) without an image_directory The directory structure would look like: `<output_directory><images>`
proxy	px	Allows you to specify proxy server setting for all your requests You can specify the proxy settings in 'IP:Port' format

Continued on next page

Table 1 – continued from previous page

Argument	Short hand	Description
similar_images	si	Reverse Image Search or 'Search by Image' as it is referred to on Google. Searches and downloads images that are similar to the absolute image link/url you provide.
specific_site	ss	Allows you to download images with keywords only from a specific website/domain name you mention.
print_urls	p	Print the URLs of the images on the console. These image URLs can be used for debugging purposes This argument does not take any value. Just add '–print_urls' or '-p' in your query.
print_size	ps	Prints the size of the images on the console The size denoted the actual size of the image and not the size of the image on disk This argument does not take any value. Just add '–print_size' or '-ps' in your query.
print_paths	pp	Prints the list of all the absolute paths of the downloaded images When calling the script from another python file, this list will be saved in a variable (as shown in the example below) This argument also allows you to print the list on the console
metadata	m	Prints the metada of the image on the console. This includes image size, origin, image attributes, description, image URL, etc. This argument does not take any value. Just add '–metadata' or '-m' in your query.
extract_metadata	e	This option allows you to save metadata of all the downloaded images in a JSON file. This file can be found in the `logs/` directory. The name of the file would be same as the keyword name This argument does not take any value. Just add '–extract_metadata' or '-e' in your query.

Continued on next page

Table 1 – continued from previous page

Argument	Short hand	Description
socket_timeout	st	Allows you to specify the time to wait for socket connection. You could specify a higher timeout time for slow internet connection. The default value is 10 seconds.
thumbnail	th	Downloads image thumbnails corresponding to each image downloaded. Thumbnails are saved in their own sub-directories inside of the main directory. This argument does not take any value. Just add '–thumbnail' or '-th' in your query.
thumbnail_only	tho	Downloads only thumbnails without downloading actual size images Thumbnails are saved in their own sub-directories inside of the main directory. This argument does not take any value. Just add '–thumbnail_only' or '-tho' in your query.
language	la	Defines the language filter. The search results are automatically returned in that language *Possible Values: Arabic, Chinese (Simplified), Chinese (Traditional), Czech, Danish, Dutch, English, Estonian. Finnish, French, German, Greek, Hebrew, Hungarian, Icelandic, Italian, Japanese, Korean, Latvianm, Lithuanian, Norwegian, Portuguese, Polish, Romanian, Russian, Spanish, Swedish, Turkish*
prefix	pr	A word that you would want to prefix in front of actual image name. This feature can be used to rename files for image identification purpose.
chromedriver	cd	With this argument you can pass the path to the 'chromedriver'. The path looks like this: "path/to/chromedriver". In windows it will be "C:\path\to\chromedriver.exe"

Continued on next page

Table 1 – continued from previous page

Argument	Short hand	Description
safe_search	sa	Searches for images with the Safe Search filter On And this filter will be Off by default if you do not specify the safe_search argument This argument does not take any value. Just add '–safe_search' or '-sa' in your query.
no_numbering	nn	When you specify this argument, the script does not add ordered numbering as prefix to the images it downloads If this argument is not specified, the images are numbered in order in which they are downloaded This argument does not take any value. Just add '–no_numbering' or '-nn' in your query.
offset	of	When you specify this argument, it will skip the offset number of links before it starts downloading images If this argument is not specified, the script will start downloading form the first link until the limit is reached This argument takes integer. Make sure the value of this argument is less than the value of limit
save_source	is	Creates a text file with list of downloaded images along with their source page paths. This argument takes a string, name of the text file.
no_download	nd	Print the URLs on the console without downloading images or thumbnails. These image URLs can be used for other purposes This argument does not take any value. Just add '–no-download' or '-nd' in your query.
silent_mode	sil	Remains silent. Does not print notification messages on the terminal/command prompt. This argument will override all the other print arguments (like print_urls, print_size, etc.)
ignore_urls	iu	Skip downloading of images whose urls contain certain strings such as wikipedia.org This argument takes a delimited set of values e.g. wikipedia.org,wikimedia.org

Continued on next page

Table 1 – continued from previous page

Argument	Short hand	Description
help	h	show the help message regarding the usage of the above arguments

Note: If `single_image` or `url` parameter is not present, then keywords is a mandatory parameter. No other parameters are mandatory.

CHAPTER 6

Examples

Many examples have been provided to help new users quickly ramp up the the usage.

6.1 Examples

Link to GitHub repo

Link to Documentation Homepage

Link to Input arguments or parameters

6.1.1 Config File Format

You can either pass the arguments directly from the command as in the examples below or you can pass it through a config file. Below is a sample of how a config file looks.

You can pass more than one record through a config file. The below sample consist of two set of records. The code will iterate through each of the record and download images based on arguments passed.

```
{
    "Records": [
        {
            "keywords": "apple",
            "limit": 5,
            "color": "green",
            "print_urls": true
        },
        {
            "keywords": "universe",
            "limit": 15,
            "size": "large",
            "print_urls": true
        }
```

(continues on next page)

(continued from previous page)

```
    ]
}
```

6.1.2 Code sample - Importing the library

- If you are calling this library from another python file, below is the sample code

```
from google_images_download import google_images_download   #importing the library

response = google_images_download.googleimagesdownload()   #class instantiation

arguments = {"keywords":"Polar bears,baloons,Beaches","limit":20,"print_urls":True}
↪#creating list of arguments
paths = response.download(arguments)   #passing the arguments to the function
print(paths)   #printing absolute paths of the downloaded images
```

6.1.3 Command line examples

- If you are passing arguments from a config file, simply pass the config_file argument with name of your JSON file

```
$ googleimagesdownload -cf example.json
```

- Simple example of using keywords and limit arguments

```
$ googleimagesdownload --keywords "Polar bears, baloons, Beaches" --limit 20
```

- Using Suffix Keywords allows you to specify words after the main keywords. For example if the `keyword = car` and `suffix keyword = 'red,blue'` then it will first search for `car red` and then `car blue`

```
$ googleimagesdownload --k "car" -sk 'red,blue,white' -l 10
```

- To use the short hand command

```
$ googleimagesdownload -k "Polar bears, baloons, Beaches" -l 20
```

- To download images with specific image extension/format

```
$ googleimagesdownload --keywords "logo" --format svg
```

- To use color filters for the images

```
$ googleimagesdownload -k "playground" -l 20 -co red
```

- To use non-English keywords for image search

```
$ googleimagesdownload -k "" -l 5
```

- To download images from the google images link

```
$ googleimagesdownload -k "sample" -u <google images page URL>
```

- To save images in specific main directory (instead of in 'downloads')

```
$ googleimagesdownload -k "boat" -o "boat_new"
```

- To download one single image with the image URL

```
$ googleimagesdownload --keywords "baloons" --single_image <URL of the images>
```

- To download images with size and type constrains

```
$ googleimagesdownload --keywords "baloons" --size medium --type animated
```

- To download images with specific usage rights

```
$ googleimagesdownload --keywords "universe" --usage_rights labeled-for-reuse
```

- To download images with specific color type

```
$ googleimagesdownload --keywords "flowers" --color_type black-and-white
```

- To download images with specific aspect ratio

```
$ googleimagesdownload --keywords "universe" --aspect_ratio panoramic
```

- To download images which are similar to the image in the image URL that you provided (Reverse Image search).

```
$ googleimagesdownload -si <image url> -l 10
```

- To download images from specific website or domain name for a given keyword

```
$ googleimagesdownload --keywords "universe" --specific_site example.com
```

===> The images would be downloaded in their own sub-directories inside the main directory (either the one you provided or in 'downloads') in the same folder you are in.

6.1.4 Library extensions

The downloading algorithm does a good job of keeping out corrupt images. However it is not ideal. There are still some chances of getting one-off corrupt image that cannot be used for processing. Below script will help clean those corrupt image files. This script was ideated by @devajith in Issue 81.

```
import os
from PIL import Image

img_dir = r"path/to/downloads/directory"
for filename in os.listdir(img_dir):
    try :
        with Image.open(img_dir + "/" + filename) as im:
            print('ok')
    except :
        print(img_dir + "/" + filename)
        os.remove(img_dir + "/" + filename)
```

CHAPTER 7

Troubleshooting

This section proviedes troubleshooting guide for commonly seen issues.

7.1 Troubleshooting Errors/Issues

Link to GitHub repo

Link to Documentation Homepage

7.1.1 SSL Errors

If you do see SSL errors on Mac for Python 3, please go to Finder —> Applications —> Python 3 —> Click on the 'Install Certificates.command' and run the file.

7.1.2 googleimagesdownload: command not found

While using the above commands, if you get `Error: -bash: googleimagesdownload: command not found` then you have to set the correct path variable.

To get the details of the repo, run the following command:

```
$ pip show -f google_images_download
```

you will get the result like this:

```
Location: /Library/Frameworks/Python.framework/Versions/2.7/lib/python2.7/site-
↪packages
Files:
  ../../../bin/googleimagesdownload
```

together they make: `/Library/Frameworks/Python.framework/Versions/2.7/bin` which you need add it to the path:

```
$ export PATH="/Library/Frameworks/Python.framework/Versions/2.7/bin"
```

7.1.3 [Errno 13] Permission denied creating directory 'downloads'

When you run the command, it downloads the images in the current directory (the directory from where you are running the command). If you get permission denied error for creating the *downloads directory*, then move to a directory in which you have the write permission and then run the command again.

7.1.4 Permission denied while installing the library

On MAC and Linux, when you get permission denied when installing the library using pip, try doing a user install.

```
$ pip install google_images_download --user
```

You can also run pip install as a superuser with `sudo pip install google_images_download` but it is not generally a good idea because it can cause issues with your system-level packages.

7.1.5 Installing the chromedriver (with Selenium)

If you would want to download more than 100 images per keyword, then you will need to install 'selenium' library along with 'chromedriver' extension.

If you have pip-installed the library or had run the setup.py file, Selenium would have automatically installed on your machine. You will also need Chrome browser on your machine. For chromedriver:

Download the correct chromedriver based on your operating system.

On **Windows** or **MAC** if for some reason the chromedriver gives you trouble, download it under the current directory and run the command.

On windows however, the path to chromedriver has to be given in the following format:

`C:\\complete\\path\\to\\chromedriver.exe`

On **Linux** if you are having issues installing google chrome browser, refer to this CentOS or Amazon Linux Guide or Ubuntu Guide

For **All the operating systems** you will have to use '–chromedriver' or '-cd' argument to specify the path of chromedriver that you have downloaded in your machine.

If on any rare occasion the chromedriver does not work for you, try downgrading it to a lower version.

7.1.6 urlopen error [SSL: CERTIFICATE_VERIFY_FAILED]

Reference to this issue

Use the below command to install the SSL certificate on your machine.

```
cd /Applications/Python\ 3.7/
./Install\ Certificates.command
```

CHAPTER 8

Workflow

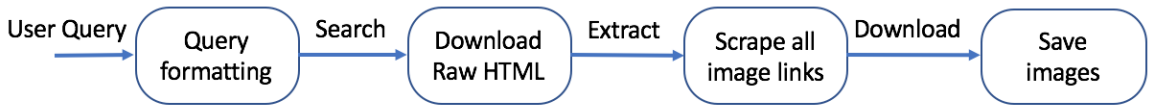
Workflow showcases the algorithm used within this module to download the images.

8.1 Workflow

Link to GitHub repo

Link to Documentation Homepage

Below diagram represents the algorithm logic to download images.



CHAPTER 9

Contribute

Anyone is welcomed to contribute to this script. If you would like to make a change, open a pull request. For issues and discussion visit the Issue Tracker.

The aim of this repo is to keep it simple, stand-alone, backward compatible and 3rd party dependency proof.

CHAPTER 10

Disclaimer

Warning: This program lets you download tons of images from Google. Please do not download or use any image that violates its copyright terms. Google Images is a search engine that merely indexes images and allows you to find them. It does NOT produce its own images and, as such, it doesn't own copyright on any of them. The original creators of the images own the copyrights.

Images published in the United States are automatically copyrighted by their owners, even if they do not explicitly carry a copyright warning. You may not reproduce copyright images without their owner's permission, except in "fair use" cases, or you could risk running into lawyer's warnings, cease-and-desist letters, and copyright suits. Please be very careful before its usage!

Index