Google OCR (Drive API v3) Documentation

Release 0.2.6

Hrishikesh Terdalkar

May 17, 2022

CONTENTS:

Perform OCR using Google's Drive API v3

- Free software: GNU General Public License v3
- Documentation: https://google-drive-ocr.readthedocs.io.

CHAPTER

ONE

FEATURES

- Perform OCR using Google's Drive API v3
- Class `GoogleOCRApplication()` for use in projects
- Highly configurable CLI
- Run OCR on a single image file
- Run OCR on multiple image files
- Run OCR on all images in directory
- Use multiple workers (`multiprocessing`)
- Work on a PDF document directly

1.1 Google OCR (Drive API v3)

Perform OCR using Google's Drive API v3

- Free software: GNU General Public License v3
- Documentation: https://google-drive-ocr.readthedocs.io.

1.1.1 Features

- Perform OCR using Google's Drive API v3
- Class `GoogleOCRApplication()` for use in projects
- Highly configurable CLI
- Run OCR on a single image file
- Run OCR on multiple image files
- Run OCR on all images in directory

- Use multiple workers (`multiprocessing`)
- Work on a PDF document directly

1.1.2 Usage

Using in a Project

Create a `GoogleOCRApplication` application instance:

```
from google_drive_ocr import GoogleOCRApplication

app = GoogleOCRApplication('client_secret.json')
```

Perform OCR on a single image:

```
app.perform_ocr('image.png')
```

Perform OCR on mupltiple images:

```
app.perform_ocr_batch(['image_1.png', 'image_2.png', 'image_3.png'])
```

Perform OCR on multiple images using multiple workers (`multiprocessing`):

```
app.perform_ocr_batch(['image_1.png', 'image_3.png', 'image_2.png'], workers=2)
```

Using Command Line Interface

Typical usage with several options:

```
google-ocr --client-secret client_secret.json \
--upload-folder-id <google-drive-folder-id>  \
--image-dir images/ --extension .jpg \
--workers 4 --no-keep
```

Show help message with the full set of options:

```
google-ocr --help
```

Configuration

The default location for configuration is `~/.gdo.cfg`. If configuration is written to this location with a set of options, we don't have to specify those options again on the subsequent runs.

Save configuration and exit:

```
google-ocr --client-secret client_secret.json --write-config ~/.gdo.cfg
```

Read configuration from a custom location (if it was written to a custom location):

```
google-ocr --config ~/.my_config_file ..
```

Performing OCR

Note: It is assumed that the `client-secret` option is saved in configuration file.

Single image file:

```
google-ocr -i image.png
```

Multiple image files:

```
google-ocr -b image_1.png image_2.png image_3.png
```

All image files from a directory with a specific extension:

```
google-ocr --image-dir images/ --extension .png
```

Multiple workers (`multiprocessing`):

```
google-ocr -b image_1.png image_2.png image_3.png --workers 2
```

PDF files:

```
google-ocr --pdf document.pdf --pages 1-3 5 7-10 13
```

Note: You must setup a Google application and download `client_secrets.json` file before using `google_drive_ocr`.

1.1.3 Setup Instructions

Create a project on Google Cloud Platform

Wizard: https://console.developers.google.com/start/api?id=drive

Instructions:

- https://cloud.google.com/genomics/downloading-credentials-for-api-access
- Select application type as "Installed Application"
- Create credentials OAuth consent screen –> OAuth client ID
- Save `client_secret.json`

1.2 Installation

1.2.1 Stable release

To install Google OCR (Drive API v3), run this command in your terminal:

```
$ pip install google_drive_ocr
```

This is the preferred method to install Google OCR (Drive API v3), as it will always install the most recent stable release.

If you don't have pip installed, this Python installation guide can guide you through the process.

1.2.2 From sources

The sources for Google OCR (Drive API v3) can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/hrishikeshrt/google_drive_ocr
```

Or download the tarball:

```
$ curl -OJL https://github.com/hrishikeshrt/google_drive_ocr/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```

1.3 Usage

1.3.1 Using in a Project

Create a `GoogleOCRApplication` application instance:

```
from google_drive_ocr import GoogleOCRApplication

app = GoogleOCRApplication('client_secret.json')
```

Perform OCR on a single image:

```
app.perform_ocr('image.png')
```

Perform OCR on mupltiple images:

```
app.perform_ocr_batch(['image_1.png', 'image_2.png', 'image_3.png'])
```

Perform OCR on multiple images using multiple workers (`multiprocessing`):

```
app.perform_ocr_batch(['image_1.png', 'image_3.png', 'image_2.png'], workers=2)
```

1.3.2 Using Command Line Interface

Typical usage with several options:

```
google-ocr --client-secret client_secret.json \
--upload-folder-id <google-drive-folder-id>  \
--image-dir images/ --extension .jpg \
--workers 4 --no-keep
```

Show help message with the full set of options:

```
google-ocr --help
```

Configuration

The default location for configuration is `~/.gdo.cfg`. If configuration is written to this location with a set of options, we don't have to specify those options again on the subsequent runs.

Save configuration and exit:

```
google-ocr --client-secret client_secret.json --write-config ~/.gdo.cfg
```

Read configuration from a custom location (if it was written to a custom location):

```
google-ocr --config ~/.my_config_file ..
```

Performing OCR

Note: It is assumed that the `client-secret` option is saved in configuration file.

Single image file:

```
google-ocr -i image.png
```

Multiple image files:

```
google-ocr -b image_1.png image_2.png image_3.png
```

All image files from a directory with a specific extension:

```
google-ocr --image-dir images/ --extension .png
```

Multiple workers (`multiprocessing`):

```
google-ocr -b image_1.png image_2.png image_3.png --workers 2
```

PDF files:

```
google-ocr --pdf document.pdf --pages 1-3 5 7-10 13
```

Note: You must setup a Google application and download `client_secrets.json` file before using `google_drive_ocr`.

1.4 Setup Instructions

Create a project on Google Cloud Platform

Wizard: https://console.developers.google.com/start/api?id=drive

Instructions:

- https://cloud.google.com/genomics/downloading-credentials-for-api-access
- Select application type as "Installed Application"
- Create credentials OAuth consent screen –> OAuth client ID
- Save `client_secret.json`

1.5 google_drive_ocr

1.5.1 google_drive_ocr package

Submodules

google_drive_ocr.application module

Google OCR Application

Create a project on Google Cloud Platform

Wizard: https://console.developers.google.com/start/api?id=drive

Instructions:

- https://cloud.google.com/genomics/downloading-credentials-for-api-access
- Select application type as "Installed Application"
- Create credentials OAuth consent screen –> OAuth client ID
- Save client_secret.json

References

- https://developers.google.com/api-client-library/python/start/get_started
- https://developers.google.com/drive/v3/reference/
- https://developers.google.com/drive/v3/web/quickstart/python

class google_drive_ocr.application.**Status**(*value*)

Bases: enum.Enum

An enumeration.

SUCCESS = 'Done!'

ALREADY = 'Already done!'

ERROR = 'Something went wrong!'

class google_drive_ocr.application.**GoogleOCRApplication**(*client_secret: str*, *upload_folder_id: Optional[str] = None*, *ocr_suffix: str = '.google.txt'*, *temporary_upload: bool = False*, *credentials_path: Optional[str] = None*, *scopes: Optional[str] = None*)

Bases: object

Google OCR Application

Perform OCR using Google-Drive API v3

client_secret: str

upload_folder_id: str = None

ocr_suffix: str = '.google.txt'

temporary_upload: bool = False

credentials_path: str = None

scopes: str = None

get_output_path(*img_path: str*) → str

Get the output path

Output path is constructed by replacing the extension in `img_path` with `ocr_suffix`

Parameters **img_path** (*str*) – Path to the input image file

Returns Output path

Return type str

get_credentials() → google.oauth2.credentials.Credentials

Get valid user credentials

If no (valid) credentials are available, * Log the user in * Store the credentials for future use

Returns Valid user credentials

Return type Credentials or None

upload_image_as_document(*img_path: str*) → str

Upload an image file as a Google Document

Parameters **img_path** (*str*) – Path to the image file

Returns ID of the uploaded Google document

Return type str

download_document_as_text(*file_id: str*, *output_path: str*)

Download a Google Document as text

Parameters

- **file_id** (*str*) – ID of the Google document
- **output_path** (*str*) – Path to where the document should be downloaded

delete_file(*file_id: str*)

Delete a file from Google Drive

Parameters **file_id** (*str*) – ID of the file on Google Drive to be deleted

perform_ocr(*img_path: str*, *output_path: Optional[str] = None*) → *google_drive_ocr.application.Status*

Perform OCR on a single image

- Upload the image to Google Drive as google-document
- [Google adds OCR layer to the image]
- Download the google-document as plain text

Parameters

- **img_path** (*str or Path*) – Path to the image file
- **output_path** (*str or Path, optional*) – Path where the OCR text should be stored If None, a new file will be created beside the image The default is None.

Returns **status** – Status of the OCR operation

Return type *Status*

_worker_ocr_batch(*worker_arguments: dict*) → float

Worker to perform OCR on multiple files

Parameters **worker_arguments** (*dict*) – Arguments for the worker

Returns Time taken in seconds

Return type float

perform_ocr_batch(*image_files: list*, *workers: int = 1*, *disable_tqdm: Optional[bool] = None*)

Perform OCR on multiple files

Parameters

- **image_files** (*list*) – List of paths to image files
- **workers** (*int, optional*) – Number of workers The default is 1.
- **disable_tqdm** (*bool, optional*) – If True, the progress bars from tqdm will be disabled. The default is None.

google_drive_ocr.cli module

Console script for Google OCR (Drive API v3)

google_drive_ocr.cli.**main**()

google_drive_ocr.errors module

HTTP Errors

List of HTTP errors that can be fixed in most cases by trying again.

Provides a `@retry` decorator, which applies exponential backoff to a function.

google_drive_ocr.errors.**retry**(*attempts: int = 4*, *delay: int = 1*, *backoff: int = 2*, *hook: Optional[Callable[[int, Exception, int], Any]] = None*) → Callable

Decorator to Retry with Exponential Backoff (on Exception)

A function that raises an exception on failure, when decorated with this decorator, will retry till it returns True or number of attempts runs out.

The decorator will call the function up to `attempts` times if it raises an exception.

By default it catches instances of the Exception class and subclasses. This will recover after all but the most fatal errors. You may specify a custom tuple of exception classes with the `exceptions` argument; the function will only be retried if it raises one of the specified exceptions.

Additionally you may specify a hook function which will be called prior to retrying with the number of remaining tries and the exception instance; This is primarily intended to give the opportunity to log the failure. Hook is not called after failure if no retries remain.

Parameters

- **attempts** (*int, optional*) – Number of attempts in case of failure. The default is 4.
- **delay** (*int, optional*) – Intinitial delay in seconds The default is 1.

- **backoff** (*int, optional*) – Backoff multiplication factor The default is 2.
- **hook** (*Callable[[int, Exception, int], Any], optional*) – Function with the parameters *(tries_remaining, exception, delay)* The default is None.

Returns Decorator function

Return type Callable

Raises

- **ValueError** – If the `backoff` multiplication factor is less than 1.
- **ValueError** – If the number of `attempts` is less than 0.
- **ValueError** – If the initial `delay` is less than or equal to 0.

google_drive_ocr.utils module

Utility Functions

google_drive_ocr.utils.**get_files**(*topdir: str*, *extn: str*) → Generator[str, None, None]

Search `topdir` recursively for all files with extension `extn`

extension is checked with `str.endswith()`, instead of the supposedly better `os.path.splitext()`, in order to facilitate the search with multiple dots in the `extn`

i.e. `>>> get_files(topdir, ".xyz.txt")` wouldn't have worked as expected if `splitext()` was used.

Parameters

- **topdir** (*str*) – Path of the directory to search files in
- **extn** (*str*) – Extension to look for

Returns Matching file paths

Return type Generator[str, None, None]

google_drive_ocr.utils.**list_to_range**(*list_of_int: List[int]*) → List[Tuple[int, int]]

Convert a list of integers into a list of ranges

A range is tuple (start, end)

Parameters **list_of_int** (*List[int]*) – List of integers

Returns List of ranges

Return type List[Tuple[int, int]]

google_drive_ocr.utils.**extract_pages**(*pdf_path: str*, *pages: Optional[Iterator[Tuple[int, int]]] = None*) → Set[str]

Extract pages from a PDF file as image files

Pages are saved in the same directory as the PDF file, with the suffix `.page-[number].jpg`

Parameters

- **pdf_path** (*str*) – Path to the PDF file
- **pages** (*Iterator[Tuple[int, int]], optional*) – Page ranges to extract. If None, all pages will be extracted. The default is None.

Returns Set of paths to extracted pages

Return type Set[str]

Module contents

Google OCR (Drive API v3).

1.6 Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

1.6.1 Types of Contributions

Report Bugs

Report bugs at https://github.com/hrishikeshrt/google_drive_ocr/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

Write Documentation

Google OCR (Drive API v3) could always use more documentation, whether as part of the official Google OCR (Drive API v3) docs, in docstrings, or even on the web in blog posts, articles, and such.

Submit Feedback

The best way to send feedback is to file an issue at https://github.com/hrishikeshrt/google_drive_ocr/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

1.6.2 Get Started!

Ready to contribute? Here's how to set up *google_drive_ocr* for local development.

1. Fork the *google_drive_ocr* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/google_drive_ocr.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv google_drive_ocr
$ cd google_drive_ocr/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 google_drive_ocr tests
$ python setup.py test or pytest
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

1.6.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 3.5, 3.6, 3.7 and 3.8, and for PyPy. Check https://travis-ci.com/hrishikeshrt/google_drive_ocr/pull_requests and make sure that the tests pass for all supported Python versions.

1.6.4 Tips

To run a subset of tests:

```
$ pytest tests.test_google_drive_ocr
```

1.6.5 Deploying

A reminder for the maintainers on how to deploy. Make sure all your changes are committed (including an entry in HISTORY.rst). Then run:

```
$ bump2version patch # possible: major / minor / patch
$ git push
$ git push --tags
```

Travis will then deploy to PyPI if tests pass.

1.7 Credits

1.7.1 Development Lead

- Hrishikesh Terdalkar <hrishikeshrt@linuxmail.org>

1.7.2 Contributors

None yet. Why not be the first?

1.8 History

1.8.1 0.2.0 (2021-06-29)

- PDF file support

1.8.2 0.1.0 (2021-06-14)

- First release on PyPI.

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

g

INDEX

Symbols

A

C

D

E

G

L

M

O

P

R

S

T

U