

VizuaMatix

Website: <https://www.vizuamatix.com/>

OCR Table Extraction



Implementation of Data Analyst Team

Author:- Pandula Pallewatta (DA Intern)

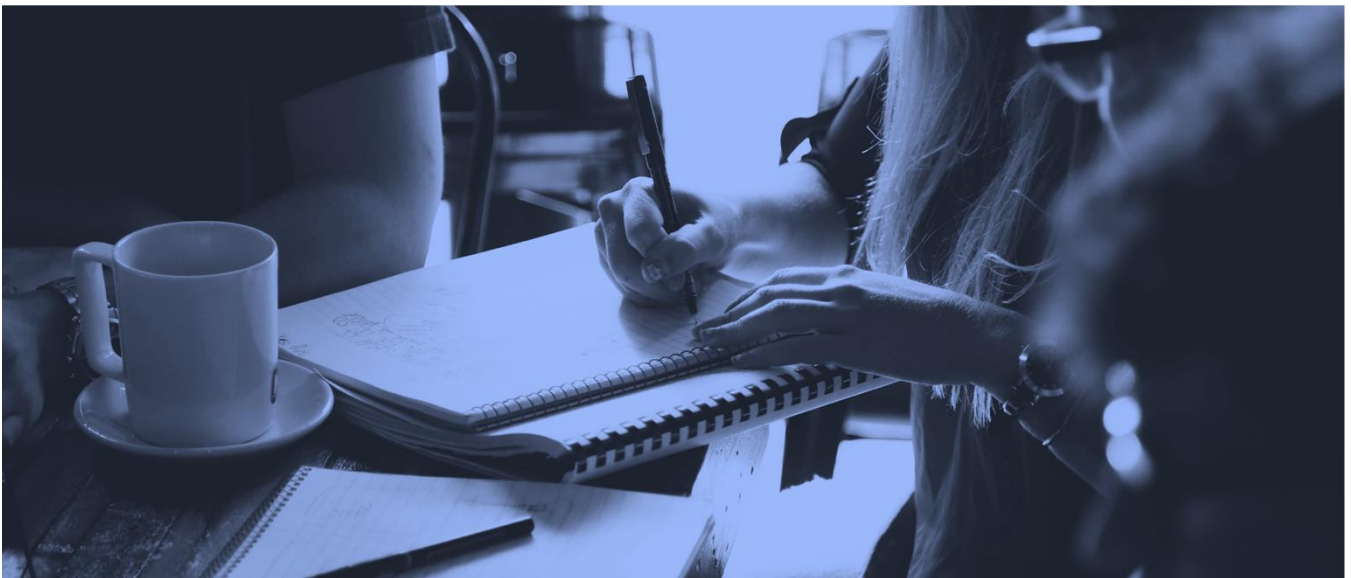


Introduction to Table Extraction Using OCR	2
Importance of Table Recognition.....	3
Industrial use cases.....	3
Method 1: Camelot Library.....	4
Prerequisites	6
Method 2: Using Tabula Library.....	9
Prerequisites	9
References.....	11

Introduction to Table Extraction Using OCR

To get started right away, just tap any placeholder text (such as this) and start typing to replace it with your own.

Some of the sample text in this document indicates the name of the style applied, so that you can easily apply the same formatting again. For example, this is the Content style.



Tables are typically used to show facts in a tidy manner. We see them all the time, from organizing our work by organizing data across tables to holding massive corporate assets. Every day, many businesses must deal with millions of tables. We need to use speedier approaches to help with such time-consuming duties of completing everything manually. Let's look at a couple scenarios where extracting tables is critical in **Industrial Business Cases**:

Importance of Table Recognition

“ Table Extraction is the task of detecting and decomposing table information in a document.”

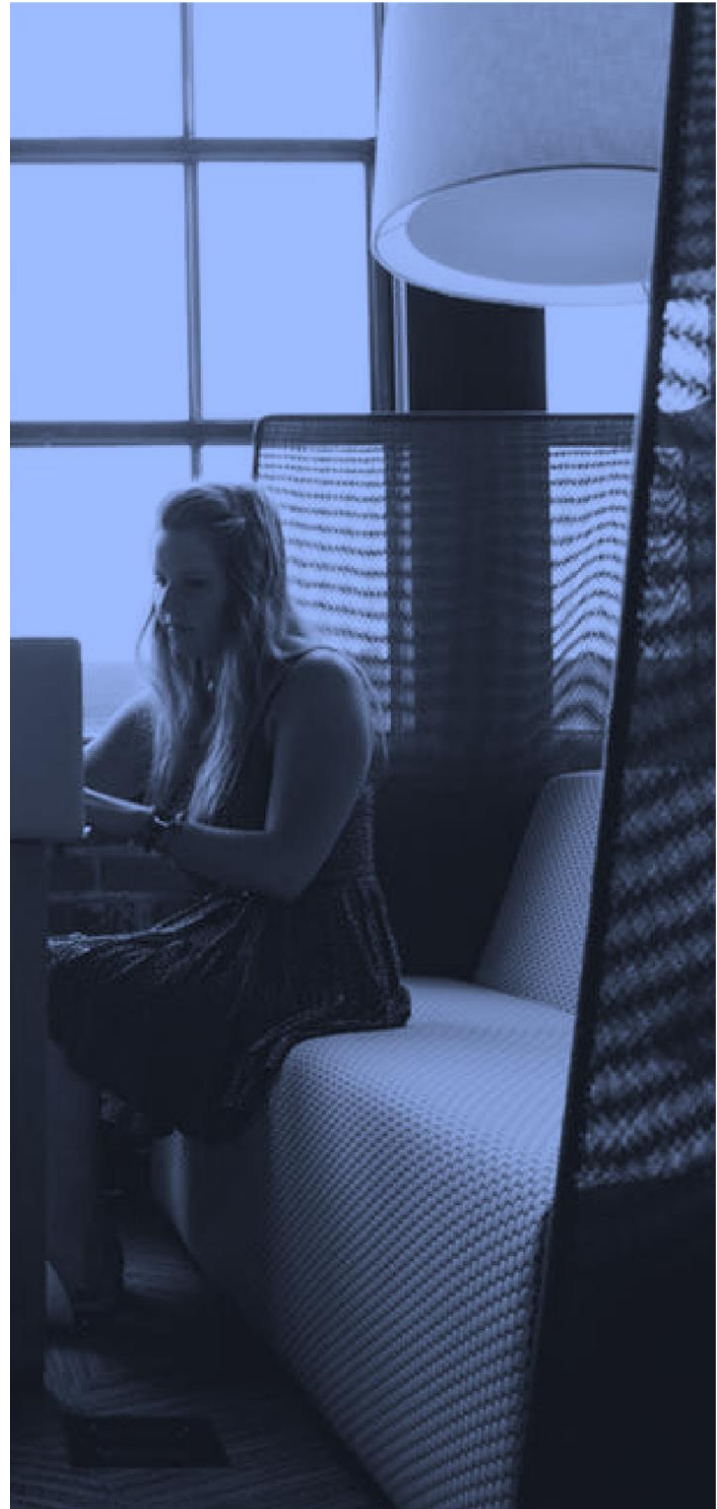
Industrial use cases

Quality Control:

In the industrial sector, tables are used to write down daily checklists and notes to track the progress of manufacturing lines. All of this may be easily documented in one place using table extraction.

Track Of Assets:

Every manufactured item is assigned a unique number, and tables are used to keep track of the products produced and delivered each day.



Method I: Camelot Library

Introduction

Camelot only works with text-based PDFs and not scanned documents.

- **Configurability:** Camelot gives you control over the table extraction process with tweakable settings.
- **Metrics:** You can discard bad tables based on metrics like accuracy and whitespace, without having to manually look at each table.
- **Output:** Each table is extracted into a **pandas DataFrame**, which seamlessly integrates into ETL and data analysis workflows. You can also export tables to multiple formats, which **include CSV, JSON, Excel, HTML, Markdown, and Sqlite.**

Documentation: - [Click Me](#)

Installation: -

PIP

To install Camelot from PyPI using `pip`, please include the extra `cv` requirement as shown:

```
$ pip install "camelot-py[base]"
```

Conda

[conda](#) is a package manager and environment management system for the [Anaconda](#) distribution. It can be used to install Camelot from the `conda-forge` channel:

```
$ conda install -c conda-forge camelot-py
```

Example Source Code: -

```
import camelot
import ghostscript
from camelot import read_pdf

# Get all the tables within the file
all_tables = read_pdf('/Users/Asus/Documents/Atmel Studio/OCR/Report_1.pdf', pages = 'all')

# Show the total number of tables in the file
print("Total number of table: {}".format(all_tables.n))

# print all the tables in the file
for t in range(all_tables.n):
    print("Table n°{}".format(t))
    print((all_tables[t].df).head())

for i in range(all_tables.n):
    all_tables[i].to_csv(f'PowerGeneration_{i+1}.csv', index=True)

for t in range(all_tables.n):
    print (all_tables[t].parsing_report)
    print ('\n')
```

Accuracy of each Table extractions

```
[5 rows x 21 columns]
{'accuracy': 99.18, 'whitespace': 27.08, 'order': 1, 'page': 1}

{'accuracy': 97.33, 'whitespace': 30.61, 'order': 2, 'page': 1}

{'accuracy': 99.55, 'whitespace': 23.75, 'order': 3, 'page': 1}

{'accuracy': 98.66, 'whitespace': 55.0, 'order': 1, 'page': 2}
```

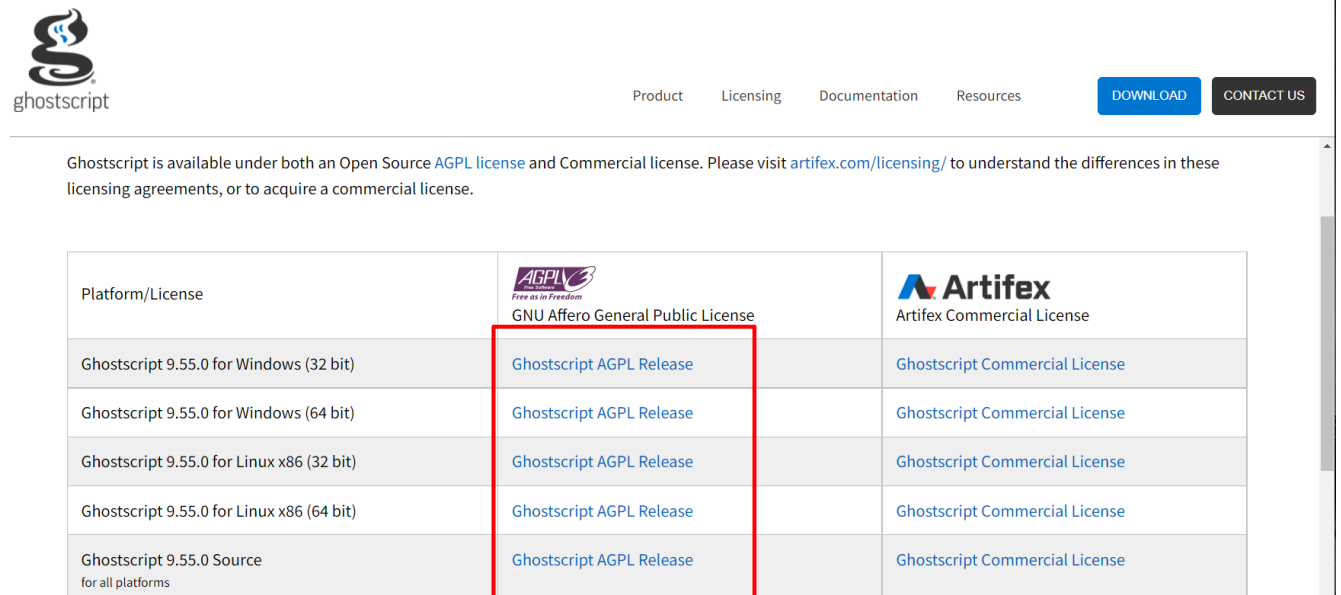
Prerequisites

1) Packages to be Installed

install ghostscript (This Package is needed to be installed with additional DLL file)

- First got this site and install Ghostscript DDL (I Windows/Linux).

Link:- [Ghostscript](#)





ghostscript

Product Licensing Documentation Resources

[DOWNLOAD](#) [CONTACT US](#)

Ghostscript is available under both an Open Source [AGPL license](#) and Commercial license. Please visit [artifex.com/licensing/](#) to understand the differences in these licensing agreements, or to acquire a commercial license.

Platform/License	 GNU Affero General Public License	 Artifex Commercial License
Ghostscript 9.55.0 for Windows (32 bit)	Ghostscript AGPL Release	Ghostscript Commercial License
Ghostscript 9.55.0 for Windows (64 bit)	Ghostscript AGPL Release	Ghostscript Commercial License
Ghostscript 9.55.0 for Linux x86 (32 bit)	Ghostscript AGPL Release	Ghostscript Commercial License
Ghostscript 9.55.0 for Linux x86 (64 bit)	Ghostscript AGPL Release	Ghostscript Commercial License
Ghostscript 9.55.0 Source for all platforms	Ghostscript AGPL Release	Ghostscript Commercial License

Select necessary package for your OS.

- Then install **ghostscript** according to your python environment.

camelot-py (Use the Pip installation in Page 4 to install Table Extraction model in Python environment)

1) Working with Camelot

➔ Specify page numbers

By default, Camelot only uses the first page of the PDF to extract tables. To specify multiple pages, you can use the `pages` keyword argument:

```
>>> camelot.read_pdf('your.pdf', pages='1,2,3')
```

The `pages` keyword argument accepts pages as comma-separated string of page numbers. You can also specify page ranges — for example, `pages=1,4-10,20-30` or `pages=1,4-10,20-end`.

➔ Reading encrypted PDFs

To extract tables from encrypted PDF files you must provide a password when calling `read_pdf()`.

```
>>> tables = camelot.read_pdf('foo.pdf', password='userpass')
>>> tables
<TableList n=1>
```

Currently Camelot only supports PDFs encrypted with ASCII passwords and algorithm code 1 or 2. An exception is thrown if the PDF cannot be read. This may be due to no password being provided, an incorrect password, or an unsupported encryption algorithm.

2) Exporting Extracted Tables

export the table as a CSV file using its `to_csv()` method. Alternatively you can use `to_json()`, `to_excel()`, `to_html()`, `to_markdown()` or `to_sqlite()` methods to export the table as JSON, Excel, HTML files or a sqlite database respectively.

Example Code

```
for i in range(tables.n):
    tables[i].to_csv(f'PowerGeneration_{i+1}.csv', index=True)

# export individually as Excel (.xlsx extension)
tables[0].to_excel("foo.xlsx")

# or export all in a zip
tables.export("foo.csv", f="csv", compress=True)

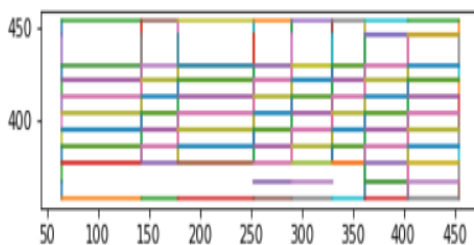
# export to HTML
tables.export("powergen.html", f="html")
```

Also, we can find the Structure of the Extracted Table using this Code.

```
camelot.plot(all_tables[2], kind='grid').show()
```

<ipython-input-11-5879e9a8949f>:2: UserWarning: Matplotlib is currently using module://ipykernel.pylab.backend_inline, which is a non-GUI backend, so cannot show the figure.

```
camelot.plot(all_tables[2], kind='grid').show()
```



Method 2: Using Tabula Library

Introduction

Tabula-Py

tabula-py is a simple Python wrapper of **tabula-java**, which can read tables in a PDF. You can read tables from a PDF and convert them into a pandas DataFrame. **tabula-py** also enables you to convert a PDF file into a CSV, a TSV or a JSON file.

Prerequisites

1) Required Environments

- ➔ Java 8+
- ➔ Python 3.6+

Installation

Ensure you have a Java runtime and set the PATH for it.

```
pip install tabula-py
```

Example Code:

tabula-py enables you to extract tables from a PDF into a DataFrame, or a JSON. It can also extract tables from a PDF and save the file as a CSV, a TSV, or a JSON.

```
from tabula import read_pdf
from tabulate import tabulate
import pandas as pd
import io

# Read the only the page n°6 of the file
import tabula

tabula.convert_into('/Users/Asus/Documents/Atmel Studio/OCR/Report_1.pdf', "Plants_Distribution.csv",
                    output_format="csv", pages = "1")
```

Converting Specific Pdf's

```
# convert PDF into CSV file
tabula.convert_into("test.pdf", "output.csv", output_format="csv", pages='all')

# convert all PDFs in a directory
tabula.convert_into_by_batch("input_directory", output_format='csv', pages='all')
```

```
In [8]: pdf_path = "/Users/Asus/Documents/Atmel Studio/OCR/Report_1.pdf"
tabula.read_pdf(pdf_path, stream=True)
```

'pages' argument isn't specified. Will extract only from page 1 by default.

```
Out[8]: [
      Unnamed: 0  Energy Generation  Unnamed: 1  \
0  Complex wise generation and energy mix      NaN      NaN
1      K-M Complex 5.38 GWh      11.83%      NaN
2      Mahaweli Complex 9.29 GWh      20.45% Hydro      15.39
3      Samanala Complex 0.72 GWh      1.58%      NaN
4      Thermal Complex 4.14 GWh      9.11%      NaN
5      Coal 11.23 GWh      24.72% Thermal      29.00
6      IPP Thermal 13.64 GWh      30.02%      NaN
7      Solar PV (Bulk) 0.32 GWh      0.69% Solar(Bulk)      0.32
8      Wind 0.73 GWh      1.60% Wind      0.73

      Unnamed: 2  Unnamed: 3
0      NaN      NaN
1      NaN      NaN
2      GWh      33.87%
3      NaN      NaN
4      NaN      NaN
5      GWh      63.84%
```

References

- 1) **Analytics Vidhya. 2022. Camelot - An Amazing Python Library to Extract Tabular Data from PDFs.** [online] Available at: <https://www.analyticsvidhya.com/blog/2020/08/how-to-extract-tabular-data-from-pdf-document-using-camelot-in-python> [Accessed 1 February 2022].
- 2) **Table Extraction with OCR (Demo)** · <https://github.com/Pandula1234/Flask-Projects-Dcau/tree/main/Table%20Extraction%20with%20OCR> [Accessed 29 January 2022].
- 3) **F. Du, “Extract tabular data from PDF with Camelot Using Python,”** 13-Jan-2019. [Online]. Available: <https://www.youtube.com/watch?v=LoiHI-IB3IY>. [Accessed 31 January 2022].