Case Study Extractor v1.2 Manual

Wilhelmina Kinderziekenhuis - Department of metabolic diseases

Program characteristics:

Operating System: Windows
Version: 1.2
Programmer: Tjardo Maarseveen
Primary Email-address: tdmaarseveen@gmail.com
Github Repository: https://github.com/levrex/DeepPhenotypingHPO
Github Wiki: https://github.com/levrex/DeepPhenotypingHPO/wiki

Table of Contents

Program characteristics: **1**

Installing Anaconda **3**

Windows systems: 3

Linux / Windows (dev) systems: 3

Setting up custom environment **3**

Option 1: Create environment with conda & pip 3

1. Build New Kernel **4**

2. Adding additional packages to the kernel **4**

3. Scraping Case Studies **4**

3.1 Scrape Case Study with Link 5

3.2 Access Case Study through library 5

3.2.1 Change Library 5

3.3 Parse downloaded html 5

3.3.1 Parse HTML 6

3.4. Failed Screening 6

3.4.1. Take screenshots if image extraction fails 6

3.4.2. Increase time to connect 6

4. Screening **7**

5. Phenotyping **7**

5.1. Select different PhenoTool 7

5.2. Apply stringent or lenient entity linking 7

6. Annotating **8**

7. Deep Phenotyping **8**

7.1 Another example: Pairwise distance table of IARS1 9

1.Installation

1.1. Installing Anaconda

If you are a developer, it might be convenient to install Jupyter notebooks. Notebooks allow you to run & test blocks of codes rather than running the pipeline all at once. If you want to use notebooks we suggest that you install Anaconda. However, if you prefer to use plain python (with Pycharm for example) then the Miniconda implementation would suffice.

Windows systems:

To create the custom environment for this tool you need to install either Miniconda or Anaconda:

1. Install Miniconda with python version 3.6+.
2. Install Anaconda with python version 3.6+. This additionally installs Jupyter Notebook & Spyder editor. It also installs Anaconda Prompt, which you can find in the windows search bar. Use this Anaconda prompt to run the commands mentioned below.

Linux / Windows (dev) systems:

Prerequisite: conda environment (with jupyter notebook). Use the terminal to run the commands mentioned below.

Install Jupyter Notebook:

$ conda install -c anaconda notebook

1.2. Setting up custom environment

Before the tool can be employed, you need to install the dependencies.

Create environment with conda & pip

prerequisite: conda

$ conda env create -f my_env.yml

$ conda activate my_env

Install the required packages in custom environment

$ pip install -r requirements.txt

Deactivate environment:

$ conda deactivate

2. Adding additional packages to the

If you want to include additional modules:

conda activate my_env

pip install seaborn

conda deactivate

3. Scraping Case Studies

There are 3 different ways to scrape the case studies from a website. Either: Scrape Case studies with link (3.1), access Case studies through a library (3.2) or by parsing a manually downloaded html (3).

To get an overview of the pipeline, you can consult the wiki at Github. If you want to carefully adjust the pipeline, we advise to examine the notebook file [**HPO extraction tool.ipynb**] rather than the CaseStudyExtractor script. Here you can run each module independently.

3.1 Scrape Case Study with Link

Users can provide the case study link to the extractor as follows:

```
python src/CaseStudyExtractor.py
https://www.wjgnet.com/1007-9327/full/v26/i15/1841.htm
```

3.2 Access Case Study through library

Certain case studies are locked behind a paywall, this requires access to a digital library. Users can provide a file [**login_details.txt**] in the root folder with username and password to automatically gain access.

The file should contain a dictionary with username and password structured as follows:

```
{'USERNAME':'1094140',

'PASSWORD':DeepPhenotyping123'}
```

Now, you can use the following command to access a case study through a library:

```
python src/CaseStudyExtractor.py
https://onlinelibrary.wiley.com/doi/full/10.1002/humu.23205 -login
1
```

3.2.1 Change Library

Disclaimer: This function currently only allows access to the Library of University of Utrecht library. You can manually change the LIBRARY parameter in the *scrapingCaseStudyLOGIN()* function from [**DeepPhenotyping_functions.py**]. Currently the following default is applied: LIBRARY='https://login.proxy.library.uu.nl/login?auth=uushibboleth&url='

3.3 Parse downloaded html

If the solution presented in 3.2 doesn't succeed in extracting the case study then users have the option to download the webpage manually. Follow the following steps to download the case study:

1. Open preferred browser
2. Navigate to case study of interest
3. Use key combination on webpage (CTRL + S) to download the html
4. Only save the HTML by indicating that you don't want to save the complete webpage in the file explorer (see figure 3.3.a)

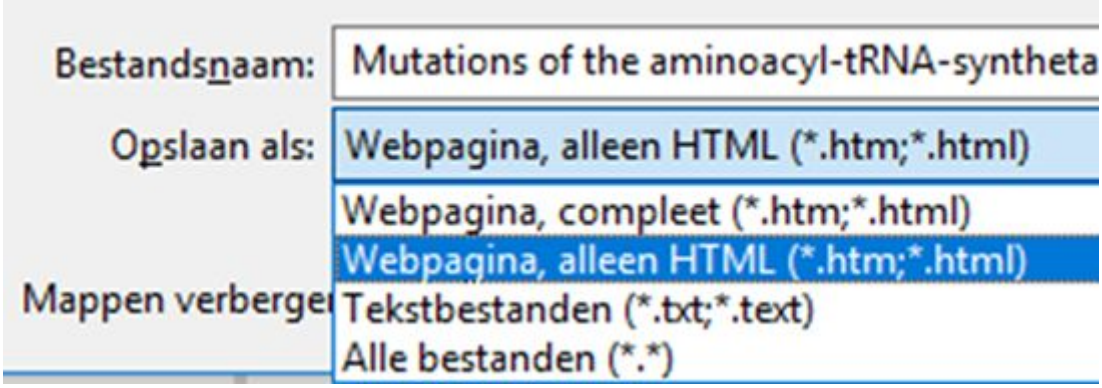

Figure 3.3.a: Save the HTML of the case study, do not save the complete webpage.

3.3.1 Parse HTML

The manually downloaded case studies can then be parsed with the Case Study Extractor via the -html flag. Be sure to provide a path to the correct directory, see example below:

```
python src/CaseStudyExtractor.py -html
PhenoTool/results/downloaded_html/Orenstein_2017_Wiley.html
```

3.4. Failed Screening

In this section we highlight potential problems with our tool & possible solutions.

3.4.1. Take screenshots if image extraction fails

Certain journals prove difficult with regard to extracting the figures. Hence, we have added a screenshot functionality that you activate as follows:

```
python src/CaseStudyExtractor.py
https://www-karger-com.proxy.library.uu.nl/Article/FullText/4
99701 -screenshots 1
```

3.4.2. Increase time to connect

Sometimes the content of the case study isn't extracted because it takes too long for a case study to render in the browser. This problem can be caused by the fact that there is not enough time scheduled (default = 10 seconds). If you have a weak wifi connection, then you might consider increasing this value:

```
 python src/CaseStudyExtractor.py
https://www-sciencedirect-com.proxy.library.uu.nl/science/article/p
ii/S0002929716301987?via%3Dihub -time 30
```

4. Screening

The first rudimentary screening aims to color the phenotypic rich regions. This screening step utilizes multiple reading frames to define the regions. Users have the possibility to adjust the parameters:

1. **BIN SIZE** = number of subsentences to qualify as 1 region (default 3)
2. **MIN POWER** = minimal number of phenotypes required to qualify as phenotypic rich region (default 3)

3. **FRAMES** = Number of "open" reading frames to alleviate the bias of initial binning (default 5). To get an idea of the you can see an example with 6 frames in figure 4.0.a

You can adjust the screening parameters with the 'intercept' flag, for example:

```
python src/CaseStudyExtractor.py
www.wjgnet.com/1007-9327/full/v26/i15/1841.htm -intercept 1 1 1
```

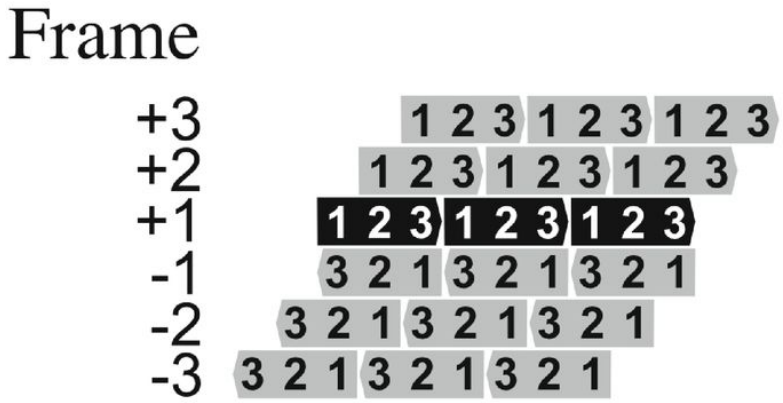

Figure 4.0.a: 6 different reading frames used in the biological context to encompass the different open reading frames

5. Phenotyping

During phenotyping a more exhaustive and stringent screening is performed. There are two parameters that can be changed by researchers: select the PhenoTool (1), perform a stringent or lenient entity linking (2).

5.1. Select different PhenoTool

Users have the possibility to choose their preferred phenotypic screening tool. There are three candidates to choose from: Clinphen, txt2hpo and Neural Concept Recognizer (NCR). By default the NCR is used, but if we want to use the txt2hpo instead we could use the 'pheno'-flag :

```
python src/CaseStudyExtractor.py
https://www.wjgnet.com/1007-9327/full/v26/i15/1841.htm -pheno
txt2hpo
```

5.2. Apply stringent or lenient entity linking

During entity linking we define and assign regions within the narrative to patients. It is important to understand that not all phenotypes mentioned in the narrative concern a patient. For example, phenotypes can refer to a family member or phenotypes refer to other literature.

Researchers have the option to apply a stringent entity linking by using the 'strict'-flag, setting this parameter to 1 will disrupt the reign of the patient within the narrative more easily:

```
python src/CaseStudyExtractor.py
https://www.wjgnet.com/1007-9327/full/v26/i15/1841.htm -strict 1
```

6. Annotating

We have an experimental entity linking step in place for creating the phenotypic patient profiles.

```
python src/CaseStudyExtractor.py
https://www-sciencedirect-com.proxy.library.uu.nl/science/article/pii/S0002929716301987?via%3Dihub -login 1 -el 1
```

It's important to note that we don't perform entity linking. Meaning that we don't appoint any of the phenotypes to a certain patient.

Disclaimer: Sadly, we couldn't find a single entity linking tool in literature that was capable of differentiating between entities, which is why we developed our own. However, we want to emphasize that this step is very rudimentary.

7. Deep Phenotyping

For Deep Phenotyping we experimented with different strategies as there is currently no consensus. The best way to adjust / experiment with the different strategies is by examining the notebook examples in the root directory: [**HPO_clustering_Freq.ipynb**], [**Phrank.ipynb**]. However, users could also take a look at our python script that incoorporates some of these functions.

To employ the deep phenotyping on the actin sample (ACTB/ACTG1) you can run the following command:

```
python src/DeepPhenotyping.py -sample actin -visual TSNE -fig 1
-phe 1
```

Other parameters:

- `data: path to file with phenotypic profiles`
- `metric: distance metric (jaccard, dice, hamming, hrss, occurrence)`
- `sample: Dimension reduction method to visualize the data (e.g: TSNE, PCA)`
- `visual : Dimension reduction method to visualize the data (e.g: TSNE, PCA)`
- `fig : Whether or not to save the intermediary figures (e.g: 0 or 1 (default))`

  `This concerns: Elbowplot & cluster membership plot`

- `phe : Calculate top 10 specific phenotypes per cluster (e.g: 0 (False) or 1 (True))`

7.1 Another example: Pairwise distance table of IARS1

When you want to apply the deep phenotyping strategy on a new dataset you can supply it with the data flag. In this case we supply an IARS1 sample. Here we already calculated the pairwise distance between patients with respect to the Hybrid relative semantic similarity (HRSS), therefore we set the 'metric' flag to none:

```
python src/DeepPhenotyping.py -data
data/deepdata/distance_matrix_IARS_hrss.csv -metric none -visual
TSNE -fig 1
```