# Tutorial for EDAMmapper

28.02.2017

## 1. Getting the EDAMmapper

The latest version of EDAMmapper is available at:
https://github.com/edamontology/edammap/releases

From there, we need those two files:
https://github.com/edamontology/edammap/releases/download/v0.1.1/edammap-0.1.1.jar
https://github.com/edamontology/edammap/releases/download/v0.1.1/processor-0.1.1.jar

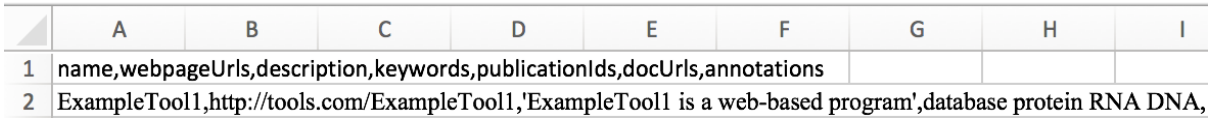
## 2. Using the EDAMmapper

### Input

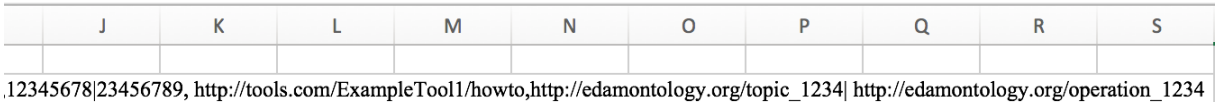
In essence, we need to give at least two files as an input. One of them is EDAM ontology file and the other is a file containing information about the tools what we want to annotate.

The latest EDAM ontology (currently, EDAM_1.16.owl) is available at:
http://edamontology.org/page

The second input file is a csv file containing information about tools what we want to annotate. An example of this file is following:

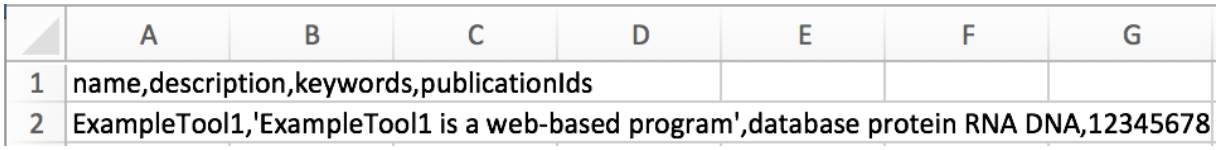


| | A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|---|
| 1 | name,webpageUrls,description,keywords,publicationIds,docUrls,annotations | | | | | | | | |
| 2 | ExampleTool1,http://tools.com/ExampleTool1,'ExampleTool1 is a web-based program',database protein RNA DNA, | | | | | | | | |



| | J | K | L | M | N | O | P | Q | R | S |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | |
| | 12345678\|23456789, http://tools.com/ExampleTool1/howto,http://edamontology.org/topic_1234\| http://edamontology.org/operation_1234 | | | | | | | | | |

Please note that different keywords should be separated by space not comma and description needs to be between ‘’. Adding annotations to csv file is important only if we want to generate benchmark file, for example to tune different parameters. This is further explained in benchmark section. In addition, it is not necessary to fill in all these fields. For example, following csv input is also functional:



| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | name,description,keywords,publicationIds | | | | | | |
| 2 | ExampleTool1,'ExampleTool1 is a web-based program',database protein RNA DNA,12345678 | | | | | | |

The simplest way to run this program is:

```
java -jar edammap-0.1.1.jar -e EDAM_1.16.owl -q tools.csv -t generic
```

Database

If we need to analyse the same inputs several times (e.g. to test different parameters) or we

want to calculate IDF file (more about it in IDF file section), then we need the contents of publications and web pages each time. In order to save some time, we are using MapDB library which allows to store information in an on-disk database. The unique key for each map entry will be an URL or a publication ID. As for the content, in case of webpages and docs, the value stored will be the string representing the plain text content of a website, and in case of publications, the stored value will be the serialized publication structure we have filled in during fetching. If a database file is specified to the automatic mapper, then contents of web pages and publications are read from there. If some are not present there and fetching is not disabled, they will be fetched from the web and put to the database. The database can also be filled beforehand, as described in here:

```
java -jar processor-0.1.1.jar --init-database new.db
java -jar processor-0.1.1.jar --add-all-missing-from-file new.db
biotools.xml biotools
```

IDF (inverse document frequency weights) file

The purpose of this file is to raise the importance of more meaningful words. For example, a large number of bioinformatics articles contain a word "sequence". However, this word definitely is not the most important word for describing all these articles. Instead, more meaningful words for describing a certain article/tool tend to occur less frequently. In this file, each word has a specific value which describes how common a term is in a collection of documents and in a sense, this value indicates how much information each word provides. We can construct it with following command:

```
java -jar processor-0.1.1.jar --make-query-idf biotools.xml biotools db.db
biotools.idf
```

Biotools content

For constructing aforementioned file and database, we need to get the content from Biotools Registry (https://bio.tools/) The latest version can be downloaded from: https://bio.tools/api/tool. However, this can also be done with attached python script (get_all_pages_from_biotools_xml.py) which we can run on a command line with following command:

```
python3 get_all_pages_from_biotools_xml.py
```

This program downloads all the content in biotools in xml format. However, content for every page is in a separate xml file. To combine all the xml files in a folder we can use another attached python file (combine_xml.py) with following command:

```
python combine_xml.py *.xml > biotools_combined.xml
```

This program outputs combined xml file from all xml files in a given folder. Yet, newly generated file is not suitable for EDAMmapper (a problem which arose with changes in biotools xml format and will be fixed in next EDAMmapper version). Firstly, this needs to be converted to different format. A file transform.xslt will do that work with a command:

```
xsltproc transform.xslt biotools_combined.xml >
biotools_combined_formatted.xml
```

**Running the EDAMmapper**

Now, you we run the programm like this:

```
java -jar edammap-0.1.1.jar -e EDAM_1.16.owl -q tools.csv -t generic -d db.db
--query-idf biotools.idf -b topic operation data format -m 5 -r results.html
-k benchmark.html
```

Some more common parameters and their meaning

"`--fetching-disabled`" -- do not download information from internet for those articles and webpages that are not in db.db
"`-b topic operation data format`" – use all ontology branches
"`-m 5`" – give 5 best suggestions for each branch
"`--obsolete`" – give also obsolete terms
"`-r results.html -k benchmark.html`" – output files

In the case of Biotools we can add following:

```
--disable-query-idf-branches data format
--good-score-data 0.77
--good-score-format 0.70
--bad-score-data 0.65
--bad-score-format 0.63
```

In order to avoid entering command line parameters we can use configuration file:
https://raw.githubusercontent.com/edamontology/edammap/master/edammapper/options.conf

We just need to remove (#) in front of parameters what we want to change from default and also remove "--" from the starting row as well as from the following row. After that we can run program like this:

```
java -jar edammap-0.1.1.jar @options.conf
```

**Output**

Results file

Results file can be outputted in two different formats: plain text and HTML. The plain text is more suitable as an input for other tools and HTML is better for displaying the results in more intuitive way. The example HTML is shown in figure 1.

On the left hand side, we can see the query content. First, the tool name (PASTA) followed by a short description. It has one publication, with PMID 24848016, attached to it. We can see the publication title, MeSH, EFO, GO terms and abstract. The full text is not presented, only its character count is printed. The tool has also one documentation URL attached, which is also brought out. The homepage can be accessed by clicking on the name (PASTA). Also, clicking on the Publication header goes to the corresponding journal article and clicking on terms goes to the individual term web pages. For EFO and GO terms, their count (how many times they were found in the full-text) is also brought out.

On the right hand side, we can see found matches, grouped by branch and ordered by score. From top to bottom, the branches are topic, operation, data and format. For each branch, we

can see the top 3 matches. First, we have the matched concept label, followed by the best matched concept part, followed by the best matched query part. In case the best matched concept part is not a label (i.e. the preferred name), the content of the concept part is brought out in parentheses after the label. Such as the narrow synonym "Mutation" for label "Genetic variation". Clicking on the matched concept label takes to the corresponding EDAM URI. If the matched label text is stricken through, this means the concept is obsolete (we did not match any obsolete concepts in the top 3). Scores are in the last column and colours are set by the score limits given as parameters and mean the following: green for good match, yellow for mediocre match and red for bad match.

If we want upload a tool to Biotools, then we should neglect those annotations which are marked as obsolete. To omit obsolete terms we just need to avoid giving following parameter `--obsolete`.

Benchmark file

As mentioned above, we may want to evaluate how well annotating works with certain parameters and test different ones or compare obtained annotations to those which were added manually. For this purpose, EDAMmapper can generate benchmark file (an example is shown in figure 2). In this file, we can see true positives, false positives and false negatives. True positives are represented in green and these are terms which were added by both - the automatic mapper and manual annotation. Yellow marks false positives which are terms that were found by the automatic mapper but were not used to annotate the tool. False negatives, represented in red, are terms that were used in manual annotation but what EDAMmapper failed to find. We can also see, that some manual annotations are stricken through, those are obsolete terms.

Extracting relevant information from the results

When we have generated results html file, we would also like to extract relevant information from it and and upload tools to Biotools. To achieve that, we can use python program parsing_results_html.py. The idea of this script is to find description and annotations from html document. However, a lot more information is needed for uploading, therefore this program should be combined with others to add all the necessary information. Unfortunately, this program is also very slow, because it fetches EDAM ontology terms from webpages. In the next version of EDAMmapper, those issues will be resolved and those scripts would then be unnecessary. Command for using this program is following:

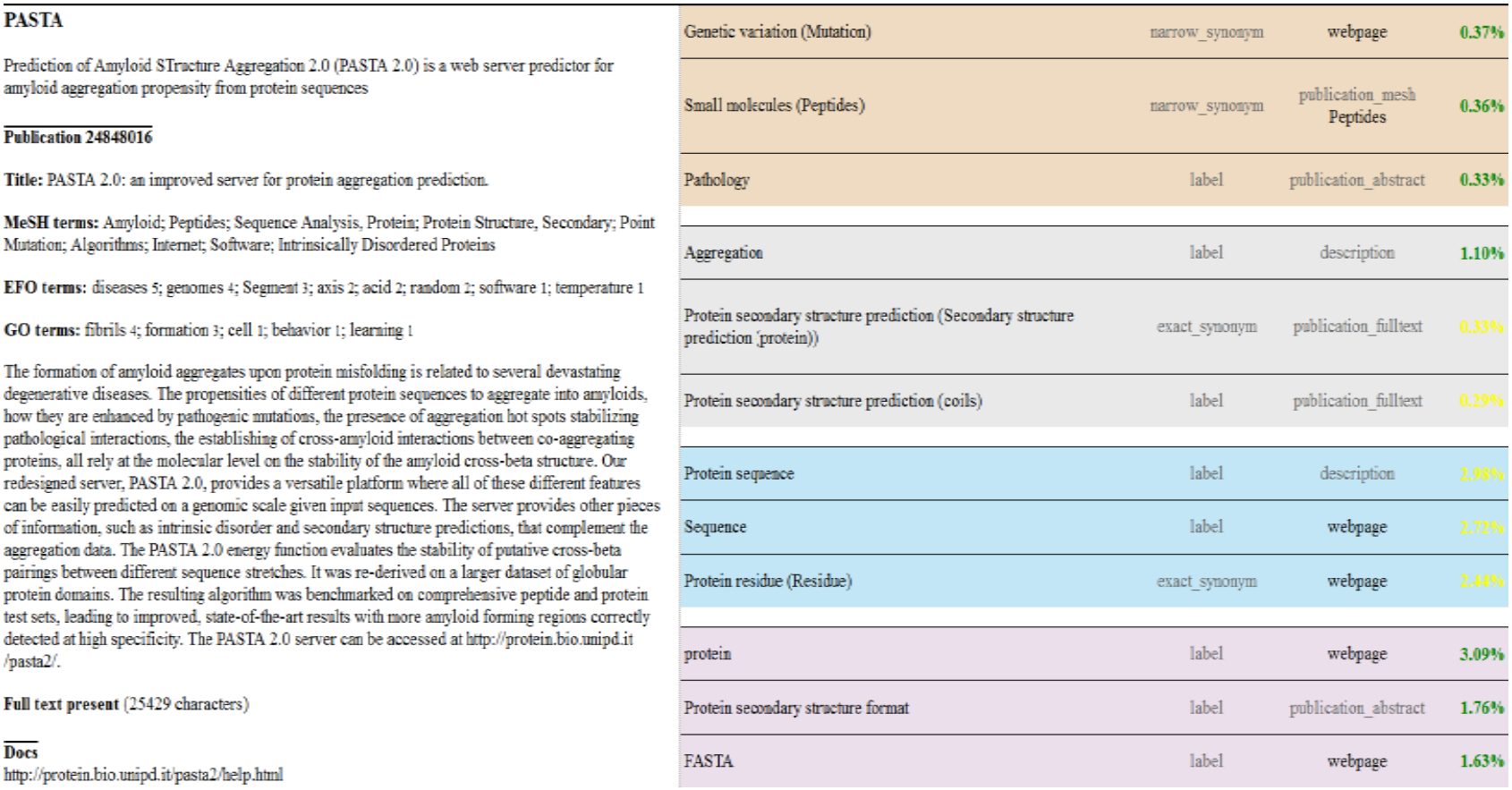
```
python3 parsing_results_html.py –t tool_type –l file_location –o output_file.json
```

This program outputs a json file ready to be added to biotools. Please note that tool type, file location and output file name are given as arguments (flag –t for tool type and –l for location and –o for output file). -t argument can be omitted and then default value for tool type ("Web application") is used. To save time in uploading the tools (not to upload one at a time) we can use a program bulkpost with following command:

```
python bulkpost.py list.json
```

This programs uploads the list of tool in json format to bio.tools and also outputs two files, one containing those tools which failed to upload and the other which contains tool names with reasons why uploading failed (for example, they might have been added already).

For further reading I attached also thesis of EDAMmapper and it can be found from following link (file name is “jaaniso_informaatika_2016.pdf”):
http://kodu.ut.ee/~ejaaniso/edammap/



**PASTA**

Prediction of Amyloid STructure Aggregation 2.0 (PASTA 2.0) is a web server predictor for amyloid aggregation propensity from protein sequences

**Publication 24848016**

**Title:** PASTA 2.0: an improved server for protein aggregation prediction.

**MeSH terms:** Amyloid; Peptides; Sequence Analysis, Protein; Protein Structure, Secondary; Point Mutation; Algorithms; Internet; Software; Intrinsically Disordered Proteins

**EFO terms:** diseases 5; genomes 4; Segment 3; axis 2; acid 2; random 2; software 1; temperature 1

**GO terms:** fibrils 4; formation 3; cell 1; behavior 1; learning 1

The formation of amyloid aggregates upon protein misfolding is related to several devastating degenerative diseases. The propensities of different protein sequences to aggregate into amyloids, how they are enhanced by pathogenic mutations, the presence of aggregation hot spots stabilizing pathological interactions, the establishing of cross-amyloid interactions between co-aggregating proteins, all rely at the molecular level on the stability of the amyloid cross-beta structure. Our redesigned server, PASTA 2.0, provides a versatile platform where all of these different features can be easily predicted on a genomic scale given input sequences. The server provides other pieces of information, such as intrinsic disorder and secondary structure predictions, that complement the aggregation data. The PASTA 2.0 energy function evaluates the stability of putative cross-beta pairings between different sequence stretches. It was re-derived on a larger dataset of globular protein domains. The resulting algorithm was benchmarked on comprehensive peptide and protein test sets, leading to improved, state-of-the-art results with more amyloid forming regions correctly detected at high specificity. The PASTA 2.0 server can be accessed at http://protein.bio.unipd.it /pasta2/.

**Full text present** (25429 characters)

**Docs**
http://protein.bio.unipd.it/pasta2/help.html

| | | | |
|---|---|---|---|
| Genetic variation (Mutation) | narrow_synonym | webpage | 0.37% |
| Small molecules (Peptides) | narrow_synonym | publication_mesh Peptides | 0.36% |
| Pathology | label | publication_abstract | 0.33% |
| Aggregation | label | description | 1.10% |
| Protein secondary structure prediction (Secondary structure prediction 'protein)) | exact_synonym | publication_fulltext | 0.33% |
| Protein secondary structure prediction (coils) | label | publication_fulltext | 0.29% |
| Protein sequence | label | description | 2.98% |
| Sequence | label | webpage | 2.72% |
| Protein residue (Residue) | exact_synonym | webpage | 2.44% |
| protein | label | webpage | 3.09% |
| Protein secondary structure format | label | publication_abstract | 1.76% |
| FASTA | label | webpage | 1.63% |

**Figure 1.** Results in HTML

**Figure 2.** Benchmark

**PASTA**

Prediction of Amyloid STructure Aggregation 2.0 (PASTA 2.0) is a web server predictor for amyloid aggregation propensity from protein sequences

**Publication 24848016**

**Title:** PASTA 2.0: an improved server for protein aggregation prediction.

**MeSH terms:** Amyloid; Peptides; Sequence Analysis, Protein; Protein Structure, Secondary; Point Mutation; Algorithms; Internet; Software; Intrinsically Disordered Proteins

**EFO terms:** diseases 5; genomes 4; Segment 3; axis 2; acid 2; random 2; software 1; temperature 1

**GO terms:** fibrils 4; formation 3; cell 1; behavior 1; learning 1

The formation of amyloid aggregates upon protein misfolding is related to several devastating degenerative diseases. The propensities of different protein sequences to aggregate into amyloids, how they are enhanced by pathogenic mutations, the presence of aggregation hot spots stabilizing pathological interactions, the establishing of cross-amyloid interactions between co-aggregating proteins, all rely at the molecular level on the stability of the amyloid cross-beta structure. Our redesigned server, PASTA 2.0, provides a versatile platform where all of these different features can be easily predicted on a genomic scale given input sequences. The server provides other pieces of information, such as intrinsic disorder and secondary structure predictions, that complement the aggregation data. The PASTA 2.0 energy function evaluates the stability of putative cross-beta pairings between different sequence stretches. It was re-derived on a larger dataset of globular protein domains. The resulting algorithm was benchmarked on comprehensive peptide and protein test sets, leading to improved, state-of-the-art results with more amyloid forming regions correctly detected at high specificity. The PASTA 2.0 server can be accessed at http://protein.bio.unipd.it /pasta2/.

**Full text present** (25429 characters)

**Docs**
http://protein.bio.unipd.it/pasta2/help.html

| | | | | | |
|---|---|---|---|---|---|
| | Genetic variation (Mutation) | | narrow_synonym | webpage | **0.37%** |
| | Small molecules (Peptides) | | narrow_synonym | publication_mesh Peptides | **0.36%** |
| | Pathology | | label | publication_abstract | **0.33%** |
| | | Sequence analysis | | | |
| | | ~~Protein-protein interactions~~ | | | |
| | Aggregation | | label | description | **1.10%** |
| | Protein secondary structure prediction (Secondary structure prediction (protein)) | | exact_synonym | publication_fulltext | **0.33%** |
| | Protein secondary structure prediction (coils) | | label | publication_fulltext | **0.29%** |
| | | ~~Protein-protein interaction prediction (from protein sequence)~~ | | | |
| Protein sequence | | | label | description | **2.98%** |
| | Sequence | | label | webpage | **2.72%** |
| | Protein residue (Residue) | | exact_synonym | webpage | **2.44%** |
| | | Sequence features | | | |
| | protein | | label | webpage | **3.09%** |
| | Protein secondary structure format | | label | publication_abstract | **1.76%** |
| FASTA | | | label | webpage | **1.63%** |
| | | MIME HTML format for Web pages, which can include external resources, including images, Flash animations and so on. | | | |