Protar Documentation

Release 0.1

Daniel Wiesmann

April 26, 2016

Contents

Protar is a Django[1] based web application to explore Natura 2000 protected areas.

You can visit the app at http://www.protar.org/

The Protar repository is hosted on GitHub[2].

[1] https://docs.djangoproject.com
[2] https://github.com/geodesign/protar

CHAPTER 1

Data Management Plan

This section describes the data sources used for analysis in Protar, as well as the data access policies for the results of the same.

1.1 Data Sources

The Protar web application has two main data sources: The Natura 2000 protected area database and the Corine Land Cover dataset. Both datasets are published under a full and open access policy.

In addition to these two main datasets, statistical area boundaries are used to aggregate historic land cover in protected areas to regional and country levels. Finally, a base map layer is used to support the visualization of the data.

All data sources are described in some more detail below.

1.1.1 Natura 2000

The Natura 2000[3] dataset is a network of protected areas throughout Europe. It was established under the 1992 Habitats Directive and is the centrepiece of EU nature & biodiversity policy. The network currently covers about 18% of Europe's territory and consists of 27372 protected areas.

The data is contributed by regional authorities to a centralized database, which is managed by the European Environment Agency (EEA). The database consists of Geographic data and tabular data. The geographic data is available in two Geographic Information System (GIS) formats (sqlite and shapefile), and the tabular data as csv or excel tables.

According to the EEA terms of use[4], the re-use of the Natura 2000 dataset is permitted free of charge for commercial or non-commercial purposes, provided that the source is acknowledged and that the entire item is reproduced. The EEA policy follows the Directive 2003/98/EC of the European Parliament. The data can be accessed here[5].

1.1.2 Corine Land cover

The Corine Land Cover (CLC) dataset is a comprehensive and consistent land cover data layer for all of Europe.

It is available for four years: 1990, 2000, 2006, and 2012. Landcover *change* layers are available. In addition to these land cover layers. These represent the land cover change between each of the above years. The 2012 version of the dataset is still in production hand has not been finalized. Nevertheless, to take advantage of the most up to date data,

[3] http://ec.europa.eu/environment/nature/natura2000/index_en.htm
[4] http://www.eea.europa.eu/legal/copyright
[5] http://www.eea.europa.eu/data-and-maps/data/ds_resolveuid/52E54BF3-ACDB-4959-9165-F3E4469BE610

the latest available version (v18.4) is used in Protar. The data can be updated once the final version of the 2012 CLC is available.

The CLC dataset is published in various GIS formats, including both vector and raster files. The data is published under a full and open access policy and is distributed by the Copernicus Land Monitoring Programme. The data can be accessed here[6].

1.1.3 Regional Summary Boundaries

The analysis conducted in Protar intersects the Natura 2000 protected area boundaries with the CLC dataset to compute landcover and landcover change statistics for the available years. These values are computed for each of the roughly 27 thousand protected areas.

While this detailed information might be relevant for managing protected areas and learning more about them, the data is also aggregated to give overviews over broad trends on a regional and country level.

The geographical boundaries used for aggregation are derived from the boundaries of the statistical areas of the Nomenclature of Territorial Units for Statistics (NUTS)[7] geographical boundaries. These boundaries are distributed through the EEA data portal under the same open access policy as the CLC dataset. It can be used free of charge for commercial and non-commercial purposes. The data can be accessed here[8].

1.1.4 Base Maps

The geographical data used in Protar will be displayed on online maps in various parts of the web application. In these visualizations, basemaps are used to give context to the protected areas and the land cover data.

The basemaps used in Protar have been produced in a collaboration between CartoDB[9] and Stamen Design[10], and are described here[11]. The basemaps are designed specifically for data overlays and are therefore ideal for Protar's purpose.

The source code to reproduce the maps is available on GitHub[12], the source code and the basemap tiles are released under a Creative Commons CC3.0[13] License and have been derived from OpenStreetMap (OSM) data.

1.2 Protar Analysis Results

The main results of the analysis conducted in Protar are data on landcover and landcover change in all protected areas of the Natura 2000 network. This information is visualized in the web application which will be publicly accessible.

The visualizations are driven by a REST API[14], a Representational State Transfer Application Programming Interface. The API is also publicly accessible and provides structured access to the results of the Protar analysis. Protar's API root can be found here[15].

The Protar api is setup with Cross- Origin Resource Sharing (CORS) headers through the django-cors-headers[16] app, so it can readily be used from within other applications anywhere on the web.

[6] http://land.copernicus.eu/pan-european/corine-land-cover/clc-2012/
[7] https://en.wikipedia.org/wiki/Nomenclature_of_Territorial_Units_for_Statistics
[8] http://www.eea.europa.eu/data-and-maps/data/administrative-land-accounting-units
[9] https://cartodb.com/
[10] http://stamen.com/
[11] https://cartodb.com/basemaps/
[12] https://github.com/cartodb/cartodb-basemaps
[13] https://creativecommons.org/licenses/by/3.0/
[14] https://en.wikipedia.org/wiki/Representational_state_transfer
[15] http://www.protar.org/api/
[16] https://github.com/ottoyiu/django-cors-headers/

All results are published under the European Union Public License (EUPL) Version 1.1[17].

1.3 Protar logo

The protar logo is a derivative work of the Natura 2000 logo[18], which is released under the CC-BY-SA 3.0[19] creative commons license. The protar logo is therefore also released under the same license.

[17] https://github.com/geodesign/protar/blob/master/LICENSE
[18] https://en.wikipedia.org/wiki/Natura_2000#/media/File:Natura_2000_logo.png
[19] https://creativecommons.org/licenses/by-sa/3.0/

CHAPTER 2

Build Database

This section describes how to re-build the Protar PostGIS database from scratch. This presumes that the app and all its dependencies are installed and that the database settings are configured to use a PostGIS backend as specified below.

Building this database takes a substantial amount of resources, and the result is publicly available. In most cases it is therefore not necessary to rebuild this dataset. The description here is for documentation purposes and will be a guidance for potential future updates.

2.1 Configure Database

The Protar app works only PostGIS as database backends. To point the app to a specific database, specify the following environmental variables.

- Database name `DB_NAME` defaults to `protar`
- Database user `DB_USER` defaults to `postgres`
- Database host `DB_HOST` defaults to `localhost`
- Database port `DB_PORT` defaults to `5432`
- Database password `DB_PASSWORD` defaults to an empty string

2.2 Download and decompress

The first step is to download the data from the EEA. The sources are described in the Data Management Plan. After download, decompress all files for Corine and Natura into separate folders. The corine data should have one subfolder with the Legend information.

For the Natura data, download the shapefiles and the tabular data as csv. Decompress both the spatial data and the tabluar data into a single folder. For the Corine data, the sqlite versions are required. Download the Corine land cover and change data for the four landcover periods as spatialite and decompress all of those into one folder. Keep one of the *Legend* subfolders contained in the zip files of the sqlite verion of the corine data. The legend folder will be used to build the Corine data legend.

For Corine data, the version required is v18.5. For the Natura data, the required version is 7.

2.3 Parse Corine Data

The next step is to parse the corine vector data. For this, build the nomenclature and the legend objects first, then load the data into the app using scripts built into protar. First, set an environmental variable telling protar where the corine data sits (separate folders for the legend and the landcover data). The legend folder should contain a `clc_legend.xls` file which comes with the landcover sqlite files. The data folder should contain spatialite files for all land cover and land cover change steps. Then scripts can be called as follows:

```
export CORINE_DATA_DIRECTORY=/path/to/corine/data/Legend
./manage.py runscript corine.scripts.nomenclature
./manage.py runscript corine.scripts.rasterlegend

export CORINE_DATA_DIRECTORY=/path/to/corine/data
./manage.py runscript corine.scripts.load
```

The data volume of the vector format of the Corine land cover is quite substantial. There are 8191080 polygons if counting all years and including change data. The size of this table in PostGIS is about 30GB, and it requires another 30GB for the index.

A part of the Corine landcover geometries are not valid geometries[20]. Before computing the intersection, it is therefore necessary to clean the Coine data. The script to clean the data calls ST_MakeValid[21] on all geometries of the dataset. Run the script using the following command:

```
./manage.py runscript corine.scripts.clean
```

2.4 Parse Natura Data

To load the Natura 2000 protected areas into the database, specify the Natura data directory through an environment variable. The Natura data folder should contain one shapefile with the Natura data and a series of CSV files with the Natura tabular data. Then the spatial and tabular data can be loaded using the following command:

```
export NATURA_DATA_DIRECTORY=/path/to/natura/data
./manage.py runscript natura.scripts.load
```

The natura data consists of 27372 protected areas, the size of the table and index is around 1GB.

2.5 Compute Intersection

Once all load scrips have completed successfully, the intersection data can be built with a script as well.

The intersection script computes the landcover statistics for all protected areas. This geoprocessing step takes many hours of computations. Therefore, the asynchronous task manager Celery[22] is used to do the geoprocessing of the data. The computations are split into small batches of Natura sites, each of which is a separate Celery task.

To learn how to setup Celery, consult its documentation. Protar assumes a local RabbitMQ[23] instance as broker and a Redis[24] instance setup for the result backend. Both are expected to be running in the default locations. In that case, celery should work automatically out of the box. To start Celery use:

[20] http://postgis.net/docs/using_postgis_dbmanagement.html#OGC_Validity
[21] http://postgis.net/docs/ST_MakeValid.html
[22] http://www.celeryproject.org/
[23] https://www.rabbitmq.com/
[24] http://redis.io/

```
celery worker -A protar --loglevel=INFO
```

The environment variables to specify a custom broker backend is `BROKER_URL`, and `CELERY_RESULT_BACKEND` for the result backend. The concurrency of the Celery workers defaults to the number of available CPUs, but can be manually specified using the `CELERYD_CONCURRENCY` environment variable. A more detailed description of how to use Celery goes beyond the scope of this documentation, consult the Celery documentation for more details.

With Celery up and running, execute the following script to add tasks to the queue that will build the intersection data squentially:

```
./manage.py runscript natura.scripts.intersect
```

Due to the data volume of both the Corine and the Natura data, this intersection is a substantial task. On a server with 4 CPUs and SSD disks the intersection took roughly 20 hours to complete.

2.6 Dump the data

The protar frontend does not make any use of the Corine landcover geometries after computing the intersection. To use the data for running the app, it is therefore sufficient to use a database without the `corine_patch` table. To dump the data without the patches, use the following command:

```
pg_dump protar --exclude-table-data=corine_patch -F c -v -f protar.dump
```

2.7 Load raster data

The parsing of the raster version of the corine data is for visualization purposes only. It is a more manual process that is done through the admin utilities of the django-raster[25] package. To load the rasters, create one `RasterLayer` object for each raster through the admin, and link it to one `CorineLayer` object in the `corine` app. The raster layers span all of europe, and hence the parsing takes about 4 hours per layer and significant amounts of disk space are required during parsing.

If you want to use the Django shell to create those rasters, use something like the following command, this will create the raster objects and trigger the parsing through celery:

```
from raster.models import RasterLayer
from django.core.files import File
rst = File(open('/path/to/corine/data/g100_clc90_V18_5.tif', 'rb'))
lyr = RasterLayer.objects.create(
    name="CLC90 V18.5", datatype='ca', srid=3035, rasterfile=rst
)
```

With that, reference the raster layer as a corine layer in the corine app. The frontend interface expects one `CorineLayer` object for each available year (1990, 2000, 2006, and 2012):

```
from corine.models import CorineLayer
CorineLayer.objects.create(rasterlayer=lyr, year=1990)
```

[25] http://github.com/geodesign/django-raster

CHAPTER 3

Legal

3.1 Funding

This application has been developed within the MyGEOSS[26] project, which has received funding from the European Union's Horizon 2020 research and innovation programme.

3.2 License and Copyright

3.3 Disclaimer

3.4 App Stores

This is a web application not a mobile application. It is therefore not available on app stores.

[26] http://digitalearthlab.jrc.ec.europa.eu/mygeoss
[27] https://github.com/geodesign/protar/blob/master/LICENSE