Epylabel: Ensemble-labeling of infectious diseases time series

Andreas Hicketier¹, **Moritz Bach**¹, **Philip Oedi**¹, **Alexander Ullrich**¹, and **Auss Abbood**²

¹ Robert Koch Institute | Unit 32
² Robert Koch Institute | Unit ZIG 1

Cite

Hicketier A, Bach M, Oedi P, Ullrich A, and Abbood A (**2024**): *Epylabel: Ensemble-labeling of infectious diseases time series*, Zenodo. DOI: 10.5281/zenodo.12665040

This repository contains the code for the manuscript *Ensemble-labeling of infectious diseases time series to evaluate early warning systems* with which you can reproduce the manuscript's results and figures.

Project Information

This code was developed at the Robert Koch Institute as part of the project *Daten- und KI-gestütztes Frühwarnsystem zur Stabilisierung der deutschen Wirtschaft* funded by the Federal Ministry for Economic Affairs and Climate Action. The project launched 1st December 2021 and ends on 30th November 2024. Together with over a dozen research and industry partners, we work on preventing economic loss as seen during the COVID-19 pandemic with the help of early warning systems. These are not limited to infectious diseases but within a work package on early warning for infectious diseases, this code was developed. For more information on the project, visit the DAKI-FWS Website and the Webiste Digitale-Technologien of the German Federal Ministry for Economic Affairs and Climate Action.

Administrative and organizational information

This work was conducted by staff from Unit 32 | Surveillance with technical supervision by Alexander Ullrich and Auss Abbood from ZIG 1 | Information Centre for International Health Protection (INIG).
The publication of the code as well as the quality management of the metadata is done by department MF 4 | Domain Specific Data and Research Data Management. Questions regarding data management and the publication infrastructure can be directed to the Open Data Team of the Department MF4 at OpenData@rki.de.

Motivation

Early warnings systems (EWS) can help make informed public health decisions. Depending on the EWS, various evaluation strategies exist such as simulating data with outbreaks or using expert-labeled data. In the absence of ground truth knowledge about outbreaks, we can use post-hoc labeling methods. While these perform well for a selection of well-behaved disease time series, they do not perform as well on heterogeneous COVID-19 time series. To address this gap for evaluation, we propose an adaptive labeling method that produces useful labels on highly heterogeneous, non-stationary COVID-19 time series.

This repository allows you to use our self-developed ensemble labeling method. It helps detect various outbreak types like waves or short peaks as occurring on different spatial resolutions and uses a majority vote to assign outbreak labels post-hoc for evaluation of EWSs. This repository also contains evaluation experiments where our self-produced labels were used to train machine learning models, which we compared with traditional outbreak detection methods.

Installation

Our scripts make use of Python and R. Please make sure you have both programming languages installed. We also encourage users to use conda as an environment management tool for this repo. After installing Anaconda or Miniconda, run the following commands in a properly configured shell:

```
conda env create -f environment.yml
conda activate epylabel
```

Running the Code

Warning: This repo uses rpy2, a Python library that enables running R code and libraries in Python. As of now, this library is not supported for Windows and this repo may not work for you if you use Windows.

Reproduce Labels

To reproduce the labels presented in the manuscript run `python paper_labels.py` after the appropriate conda environment has been activated. Note, you need to navigate to the folder containing this script for it to work.

Generate Figures

You can also reproduce the figures from the manuscript using `python paper_plots.py`

Generating Docs

You can build the docs with Sphinx:

```
sphinx-build -b html docs/source/ docs/build/
```

Code

This repo is using a pipeline approach to compose the ensemble of labeling methods. Each labeling method inherits from the abstract class `Transformation` (see labeler.py). Theses Classes need to implement the `transform()` method that either return labels or transformed data.

The `Pipeline` class allows you to execute transform operations of various labeling methods successively.

Lastly, the `Ensemble` class implements the routine for the majority vote of each single labeling method in the ensemble. The code can be extended to use more labeling methods. Each method would only need to inherit from `Transformation`.

If another ensemble voting mechanism is desired, a new `Ensemble` class can be implemented where you specify your voting approach in the `transform()` method. This way, our code is open to new implementations and variations.

Below, you can find a shortened and commented version of `paper_labels.py` to illustrate how generating labels with our ensemble approach works.

```
import pandas as pd

from epylabel.labeler import (Bcp,Changerate,Ensemble,Shapelet,WaveFinder)
from epylabel.pipeline import Pipeline
from paper_labels import StandardForm

# Instatiate single labeling methods with adequate parameters
cr = Changerate()
bcp = Bcp()
wv = WaveFinder()
sp = Shapelet()

# Instatiate ensemble
ens = Ensemble(n_min=2)

# Download RKI COVID-19 data
data_rki_url = (
    "https://raw.githubusercontent.com/robert-koch-institut/"
    "COVID-19_7-Tage-Inzidenz_in_Deutschland/main/"
    "COVID-19-Faelle_7-Tage-Inzidenz_Deutschland.csv"
)
data_rki = pd.read_csv(data_rki_url)

# Rearrange data
data_wide = Pipeline([StandardForm()]).transform(data_rki)
data_wide_faelle = Pipeline(
    [
        StandardForm("Faelle_neu"),
    ]
).transform(data_rki)

# Label data with single labeling methods
bcp_labels = Pipeline(
    [
        cr,
        bcp,
    ]
).transform(data_wide_faelle)
sp_labels = Pipeline([sp]).transform(data_wide)
wv_labels = Pipeline([wv]).transform(data_wide)

# Combine labeling methods in ensemble
bcp_sp_wv_labels = Pipeline([ens]).transform(bcp_labels, sp_labels, wv_labels)
```

Data

The code in this repository depends on reported COVID-19 cases in Germany. The main function `paper_labels.py`, which is more closely explained in the next section, downloads data from the Robert Koch Institute's Open Data Repository on GitHub for which it then produces the labels as described in the manuscript.

There are three datasets that will be downloaded to build timeseries of newly reported cases. New cases are in the CSV's column `Faelle_neu`. Region identifiers which are named `Bundesland_id` for federal countries and `Landkreis_id` for counties, are renamed to `location` by the script. The reporting date `Meldedatum` is renamed to `target` and the case numbers to `value`. Without a regional stratification, i.e., timeseries for Germany only, the column `location` gets the value 0. Age stratification of the data is ignored.

The repository is using the latest data form the RKI "7-Tage-Inzidenz der COVID-19-Fälle in Deutschland" dataset provided on Github:

> https://github.com/robert-koch-institut/COVID-19_7-Tage-Inzidenz_in_Deutschland

All versions of the currently daily updated data, are also published on Zenodo.org:

> Robert Koch-Institut (**2024**): 7-Tage-Inzidenz der COVID-19-Fälle in Deutschland, Berlin: Zenodo. DOI: 10.5281/zenodo.7129007

Description	URL
COVID-19 cases in Germany per county	https://raw.githubusercontent.com/robert-koch-institut/COVID-19_7-Tage-Inzidenz_in_Deutschland/main/COVID-19-Faelle_7-Tage-Inzidenz_Landkreise.csv
COVID-19 cases in Germany per federal state	https://raw.githubusercontent.com/robert-koch-institut/COVID-19_7-Tage-Inzidenz_in_Deutschland/main/COVID-19-Faelle_7-Tage-Inzidenz_Bundeslaender.csv
COVID-19 cases in Germany without startification	https://raw.githubusercontent.com/robert-koch-institut/COVID-19_7-Tage-Inzidenz_in_Deutschland/main/COVID-19-Faelle_7-Tage-Inzidenz_Deutschland.csv

After the transformation, the data has the following structure:

Column	Datatype	Description
value	integer	Number of reported COVID-19 cases
target	string	Reporting date (yyyy-mm-dd)
location	string	The five-digit community identification code for counties, two-digit code for federal countries, and a 0 for the whole of Germany

Formatting

Data is downloaded as a comma-separated .csv file. The character encoding is UTF-8. Values are separated by ",".

Metadata

To increase findability, the provided repository is described with metadata. The metadata is distributed to the relevant platforms via GitHub Actions.
Versioning and DOI assignment are performed via Zenodo.org. The metadata prepared for import into Zenodo are stored in the zenodo.json. Documentation of the individual metadata variables can be found at https://developers.zenodo.org/representation.

> metadata/zenodo.json

Collaborate

If you want to participate in our project, feel free to fork this repo and send us pull requests. To make sure everything is working please use pre-commit. It will run a few tests and lints before a commit can be made. To install pre-commit, run

```
pre-commit install
```

Guidelines for Reuse of the Code

Open source code from the RKI is available on Zenodo.org, GitHub.com and OpenCoDE:

- https://zenodo.org/communities/robertkochinstitut
- https://github.com/robert-koch-institut
- https://gitlab.opencode.de/robert-koch-institut

License

The "Epylabel: Ensemble-labeling of infectious diseases time series" code is licensed under the MIT License.

The code provided in the repository is freely available, with the condition of attributing the Robert Koch Institute as the source, for anyone to process and modify, create derivatives of the dataset and use them for commercial and non-commercial purposes.
Further information about the license can be found in the LICENSE file of the dataset.