Reproducibility review of: Geoparsing: Solved or Biased? An Evaluation of Geographic Biases in Geoparsing

Daniel Nüst, Eleni Tomai

2022-06-10

This report is part of the reproducibility review at the AGILE conference. For more information see https://reproducible-agile.github.io/. This document is published on OSF at https://osf.io/3dsmv/. To cite the report use

> Nüst, Daniel, and Eleni Tomai. 2022. Reproducibility Review of: Geoparsing: Solved or Biased? An Evaluation of Geographic Biases in Geoparsing. OSF. https://doi.org/10.17605/OSF.IO/3DSMV

Reviewed paper

> *Liu, Z., Janowicz, K., Cai, L., Zhu, R., Mai, G., and Shi, M.: Geoparsing: Solved or Biased? An Evaluation of Geographic Biases in Geoparsing, AGILE GIScience Ser., 3, 9, https://doi.org/10.5194/agile-giss-3-9-2022*

Summary

The article presents an evaluation of geoparsing performance using a number of different datasets and methods from various sources. Though preprocessing steps and a core analysis step based on proprietary software could not be evaluated, one of two toponym resolution models could be executed successfully. The provided notebooks for exploratory analysis, calculating statistical values, and geographic bias evaluation could be run and the outputs match the data and figures presented in the paper. Therefore, this reproducibility report can confirm a partially successful reproduction of a complex pipeline, for which authors provide reasonable but improvable documentation and share all details (code, data) of their computational workflow.

Reproducibility reviewer notes

The authors provide a link to a GitHub repository at https://github.com/zilongliu-geo/Geoparsing-Solved-Or-Biased. The README gives detailed instructions how to reproduce the data processing steps. These instructions were updated after initial contacts with the first reproducibility reviewer (ET), before a second reviewer (DN) took over the task due to technical issues.

The preprocessing was conduced with ArgGIS PRO, for which I do not have a license. Therefore, I continue with the given preprocessed data in the step "Deploying Geoparsers".

First, let's get the code and data:

```
https://github.com/zilongliu-geo/Geoparsing-Solved-Or-Biased
```

Deploying Geoparsers

The README includes links to used software projects for installation instructions. I had to reinstall spaCy several times, as I first installed the latest version (3.3.0) instead of the used one (2.1) and then realised that NeuroTPR needs a Conda environment.

```
conda create -n agile-019 python=3.6

# for NeuroTPR
conda activate agile-019
conda install keras -c conda-forge
pip install git+https://www.github.com/keras-team/keras-contrib.git
pip install neurotpr

# for spaCy
pip install spacy==2.1.9 --ignore-installed
python -m spacy download en_core_web_lg

# NeuroTPR pretrained model
wget https://github.com/geoai-lab/NeuroTPR/blob/master/PretrainedModel.zip?raw=true
```

This gives me the following environment from `pip freeze` (updated to include pandas, see below):

```
blis==0.2.4
certifi==2022.5.18.1
charset-normalizer==2.0.12
click==8.0.4
cymem==2.0.6
emoji==1.7.0
en-core-web-lg @ https://github.com/explosion/spacy-models/releases/download/en_core_web_lg-2.1.0/en_core_web_lg-2.1.0.tar.gz
idna==3.3
importlib-metadata==4.8.3
importlib-resources==5.4.0
joblib==1.1.0
keras @ file:///home/conda/feedstock_root/build_artifacts/keras_1652925255303/work/keras-2.9.0-py2.py3-none-any.whl
keras-contrib @ git+https://www.github.com/keras-team/keras-contrib.git@3fc5ef709e061416f4bc8a92ca3750c824b5d2b0
murmurhash==1.0.7
neurotpr==0.0.9
nltk==3.6.7
numpy==1.19.5
pandas==1.1.5
plac==0.9.6
preshed==2.0.1
protobuf==3.19.4
python-dateutil==2.8.2
pytz==2022.1
regex==2022.4.24
requests==2.27.1
six==1.16.0
spacy==2.1.9
srsly==1.0.5
tensorflow-hub==0.12.0
thinc==7.0.8
tqdm==4.64.0
typing_extensions==4.1.1
urllib3==1.26.9
wasabi==0.9.1
zipp==3.6.0
```

Installation seems to have worked, the tools for the toponym recognition models should be available now.

Spatially-Explicit Geoparsing Performance Evaluation

Running Toponym Recognition Models

First, running toponym recognition models **spaCy**.

```
cd Geoparsing-Solved-Or-Biased/

python scripts/toponym-recognition-GeoCorpora-spaCy.py
Traceback (most recent call last):
  File "Geoparsing-Solved-Or-Biased/scripts/toponym-recognition-GeoCorpora-spaCy.py", line 7, in <module>
    import pandas as pd
ModuleNotFoundError: No module named 'pandas'

# install pandas with
pip install pandas
# and run again:
[..]
FileNotFoundError: [Errno 2] No such file or directory: '/home/daniel/git/reproducible-agile/reviews-2022/reports\\data\\evaluation-corpora\\
```

Going back to "Downloading Datasets" in the README, I access the required data and run again. After some fiddling with the paths and the directory from which I run the script, it works after changing line 15 of `toponym-recognition-GeoCorpora-spaCy.py` to

```
dir_geocorpora = os.path.join(os.path.dirname(os.getcwd()), 'data', 'evaluation-corpora', 'original-datasets', 'geocorpora.tsv')
```

I also change the output folder in line 64f.:

```
dir_geocorpora_results = os.path.join(os.path.dirname(os.getcwd()), 'geoparsed-results-reproduced')
df_geocorpora_poi.to_csv(os.path.join(dir_geocorpora_results, 'geocorpora_geotagged_results_spaCy.csv'))
```

```
cd data/evaluation-corpora/original-datasets
wget -O geocorpora.tsv https://github.com/geovista/GeoCorpora/raw/master/geocorpora_1544784178012.tsv
cd ../..

cd scripts
python toponym-recognition-GeoCorpora-spaCy.py

# /home/daniel/miniconda3/envs/agile-019/lib/python3.6/site-packages/pandas/core/indexing.py:670: SettingWithCopyWarning:
# A value is trying to be set on a copy of a slice from a DataFrame
#
# See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
#  iloc._setitem_with_indexer(indexer, value)
```

The code runs for about a minute and creates output files, which match the ones provided by the authors:

```
diff geoparsed-results-reproduced/geocorpora_geotagged_results_spaCy.csv geoparsed-results/geocorpora_geotagged_results_spaCy.csv
# no output, files are the same.
```

Now the toponym recognition model with **NeuroTPR** needs to be run, though I have problems:

```
python toponym-recognition-GeoCorpora-NeuroTPR.py
Traceback (most recent call last):
# [...]
ModuleNotFoundError: No module named 'tensorflow'

# install Tensorflow with
pip install tensorflow==1.15
# and try again:

python toponym-recognition-GeoCorpora-NeuroTPR.py
Traceback (most recent call last):
# [..]
  File "/home/daniel/miniconda3/envs/agile-019/lib/python3.6/site-packages/keras/backend.py", line 39, in <module>
    from tensorflow.python.eager.context import get_config
ImportError: cannot import name 'get_config'
```

I'm optimistic this part of the workflow could be made to work as well but don't want to go down the rabbit hole of updating all depencies and finding working versions for the different tools. Also, the paths need to be fixed in the same manner in the second script.

Running Toponym Resolution Models

```
# get data, run following command in data/evaluation-corpora/original-datasets/
wget https://raw.githubusercontent.com/milangritta/Geocoding-with-Map-Vector/master/data/lgl.xml
# repeat for files GeoVirus.xml and WikToR.xml

# get patch data - directory "data-patches" needed to be created first, run following commands in data/evaluation-corpora/data-patches/
wget https://raw.githubusercontent.com/google-research-datasets/mlg_evaldata/main/lgl_patches.tsv
# repeat for files GeoVirus_patches.tsv and WikToR_patches.tsv

# run notebook
jupyter notebook toponym-resolution-results-Edinburgh-Geoparser-integration.ipynb

# needs geophy
pip install geopy
# run notebook again
```

This notebook needs some files from running Edinburgh Geoparser, which was skipped as the steps are not documented in detail. *This part of the workflow could note be reproduced.*

Exploratory Analysis on Geoparsing Performance Indicators

I run a Jupyter server to open the notebooks:

```
pip install jupyter

# in Geoparsing-Solved-Or-Biased/scripts
jupyter notebook exploratory-analysis-recall.ipynb

# installing missing libs
pip install matplotlib seaborn
```

After fixing the paths, I get matching results for **Figure 1 (a) and (b)**:

```
knitr::include_graphics(here::here("019", "figure1a.png"))
```

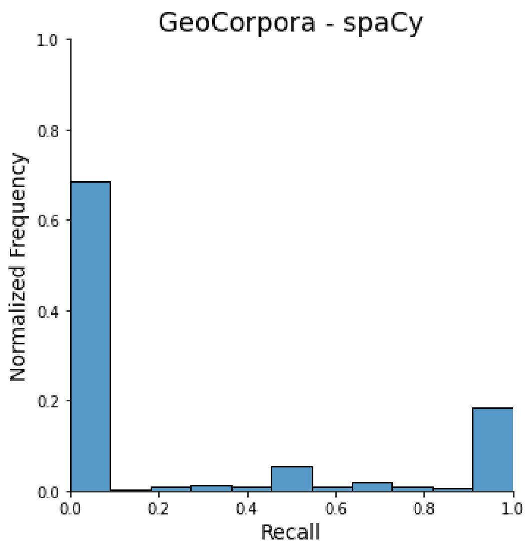

Figure 1: Figure 1 (a)

```
knitr::include_graphics(here::here("019", "figure1b.png"))
```

In the same manner, after fixing paths to work on Linux, I can execute all notebook cells and recreate **Figure 2 (a) - (f)** when running `jupyter notebook exploratory-analysis-mdned.ipynb`.

Calculating the Standard Deviation of MdnED for Highly Ambiguous Toponyms

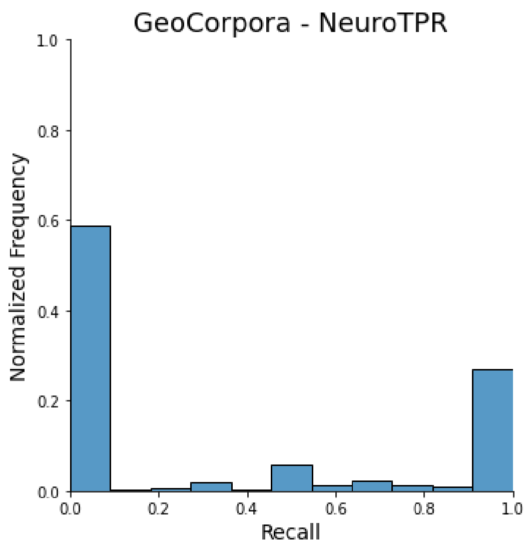

Figure 2: Figure 1 (b)

```
jupyter notebook standard-deviation-toponym-resolution-ambiguity.ipynb
```

After fixing paths, the calculated statistics match the values in **Table 1**:

```
# Edinburgh
Washington County      median_error_distance
std             807.704385
Clinton      median_error_distance
std                    NaN
Springfield      median_error_distance
std                    NaN
Greenville      median_error_distance
std             407.517261
Kingston      median_error_distance
std             680.481725
Georgetown      median_error_distance
std                    NaN
Hamilton      median_error_distance
std            4105.574962
Jefferson County      median_error_distance
std                    NaN
Newport      median_error_distance
std            5705.194849
Franklin County      median_error_distance
std                    NaN

# CamCorder
Washington County      median_error_distance
std             692.177184
Clinton      median_error_distance
std              539.43373
Springfield      median_error_distance
std            2770.634287
Greenville      median_error_distance
std             514.681684
Kingston      median_error_distance
std             226.447057
Georgetown      median_error_distance
std            2742.785136
Hamilton      median_error_distance
std             622.237286
Jefferson County      median_error_distance
std             384.380587
Newport      median_error_distance
std             145.050226
Franklin County      median_error_distance
std             414.985655
```

Spatial Autocorrelation Analysis

This part of the workflow requires the proprietary software ArcGIS pro, for which I do not have access.

Geographic Bias Evaluation

```
jupyter notebook representation-bias-measurement.ipynb
```

This fails at first, because the data files are stored with Git LFS, which I must install first and then fetch the actual data files of about 10GB (!).

```
git lfs install
git lfs fetch
git lfs checkout

ls -l -h data/grids-100sqkm-admin0-natural-earth/
total 8,9G
-rw-r--r-- 1 daniel daniel 1,5G Mai 27 13:23 grids_summary_GeoCorpora_poi.csv
-rw-r--r-- 1 daniel daniel 1,5G Mai 27 13:23 grids_summary_GeoNames_poi.csv
-rw-r--r-- 1 daniel daniel 1,5G Mai 27 13:23 grids_summary_GeoVirus_poi.csv
-rw-r--r-- 1 daniel daniel 1,5G Mai 27 13:23 grids_summary_GeoWiki_poi.csv
-rw-r--r-- 1 daniel daniel 1,5G Mai 27 13:23 grids_summary_LGL_poi.csv
-rw-r--r-- 1 daniel daniel 1,5G Mai 27 13:23 grids_summary_WikToR_poi.csv
```

Afterwards I could execute all cells in the notebook. It is not well documented how, but the statistical values calculated in the notebook seem to be aggregated in **Table 2** and **Table 3** in the paper for *Spatial Misalignment* (`sm_` objects the notebook) and *Spatial Diversity Misalignment* (`sdm_` objects in the notebook), with some difference due to rounding:

```
# summary statistics of sm_geowiki
count    181.000000
mean       0.471373
std        0.218703
min        0.000000
25%        0.309278
50%        0.494639
75%        0.628910
max        0.865160

# summary statistics of sdm_geowiki
count    178.000000
mean      -0.223448
std        0.146347
min       -1.000000
25%       -0.303085
50%       -0.216342
75%       -0.143549
max        0.428039

# sm_wiktor
count    124.000000
mean       0.892736
std        0.096244
min        0.492557
25%        0.866875
50%        0.916329
75%        0.956252
max        1.000000

# sdm_wiktor
count    124.000000
mean      -0.812131
std        0.166701
min       -1.000000
25%       -1.000000
50%       -0.833592
75%       -0.702824
max       -0.261689

# sm_geocorpora
count    110.000000
mean       0.910579
std        0.128534
min        0.350839
25%        0.895627
50%        0.957071
75%        0.987779
max        0.998944

# sdm_geocorpora
count    110.000000
mean      -0.906054
std        0.127746
```

```
min      -1.000000
25%      -1.000000
50%      -1.000000
75%      -0.860386
max      -0.431835
```

The values in the paper's tables are based on the data provided in the GitHub repository.

Recreated figures are stored in the OSF repository of this report. Changed code and notebooks are in this commit: https://github.com/reproducible-agile/Geoparsing-Solved-Or-Biased/commit/265fb8cd4c91616be4747d301eac400f99c98ae2.

Comments to the authors

- Please add information about data sizes and processing times you have experienced on your machine, so that users are informed about what resources they should have prepared.
- Consider providing some kind of automated/scripted way to install the required software in the actually used version.
- Please provide a clear mapping between figures in the paper and which scripts create them, e.g., in the README file.
- Use OS-agnostic ways to construct paths, and be explicit in which directory you run scripts.
- Before submissions, I strongly recommend you re-run your whole workflow on a new machine, or as a colleague not involved in the project to do so; this will greatly improve the level of documentation and stability of your workflow, also for your own benefit (e.g., undocumented use of GitLFS).
- Even when using ArcGIS, consider using a Python script to control the workflow to increase reproducibility.
- You rely on distributed data sources, which are published under open licenses though. Consider having a script to download all of these files, or just saving a copy into your own repository, and documenting the different licenses clearly - it makes other people's lives a lot easier.