CORE-SG: Efficient Computation of Multiple MSTs for Density-Based Methods

This repository contains the code corresponding to the techniques proposed and discussed in the manuscript **"CORE-SG: Efficient Computation of Multiple MSTs for Density-Based Methods"**, submitted to VLDB'22, as well as the instructions to reproduce the results reported in the paper.

We compare our proposal with the previous state-of-the-art approach for the computation of multiple density-based Minimum Spanning Trees (MSTs) based on Relative Neighborhood Graphs (RNGs) [3], proposed in the context of HDBSCAN* [1]. The RNG-based approach has been re-implemented in Python/Cython and is also available in this repository.

This implementation uses Cython to improve the performance of many operations that would be otherwise very inefficient if implemented with only Python. This short guide includes the 4 steps needed to reproduce the results reported in the paper: 1. Installation 2. Getting the Data 3. Running Experiments 4. Generating Figures

We have also included a troubleshooting section to help with issues that can potentially occur during installation.

1. Installation

The first step for installing the project consists of downloading the code from the repository publicly available at Github. Note that if this repository has been already downloaded as ZIP file, the clonning of the Github repository is not necessary.

```
git clone git@github.com:antoniocavalcante/coresg.git
```

Next, change to the project folder and create a virtual environment that will contain the packages and dependencies needed to run the CORE-SG project.

```
python -m venv env
```

After the virtual environment has been created, it must be activated so the installation of the dependencies is done in the virtual environment.

```
source env/bin/activate
```

Next, run the following command to install the required dependencies listed in the `requirements.txt` file.

```
python3 -m pip install -r requirements.txt
```

After the required dependencies are installed, build the Cython modules with the following command.

```
python3 setup.py build_ext --inplace
```

At this point, the installation should be ready for use. In case of errors, check out the troubleshooting section at the end of this README page.

2. Getting the Data

1. Download the data from this Google Drive URL
2. Unzip the data besides the git repository.

```
.
├── coresg/
├── data/
    ├── dataset-handl/
    ├── dataset-real/
```

The `dataset-handl` folder contains the CSV files corresponding to the synthetic datasets constructed with the generator proposed by Handl et al. [2], and the `dataset-real` folder contains the CSV files corresponding to the real datasets. The detailed description of the parameters used to generate the synthetic datasets in our experimental evaluation, as well as description of the real datasets, is included in the paper.

3. Running Experiments

The bash script files `experiments_real.sh` and `experiments_handl.sh`, located at the root directory of `coresg`

In order to run the experiments for **ALL** strategies considered in the paper, run the following command:

```
./experiments_handl.sh "../data/dataset-handl" "ALL"
```

If only the experiments regarding **CORE-SG** should be executed, then the following command should be run:

```
./experiments_handl.sh "../data/dataset-handl" "CORE"
```

Similarly, to run experiments regarding only the incremental construction of the **CORE-SG**, run the following command:

```
./experiments_handl.sh "../data/dataset-handl" "CORE_INC"
```

In order to run experiments with real data, the directory and names of the data are encoded inside the script

```
./experiments_real.sh
```

4. Generating Figures

The directory `plots` contains a collection of Python, Gnuplot and Bash scripts that process the results runtimes measured in our experiments and generate the figures included in the paper.

```
coresg/
├── plots/
    ├── gnuplot/
    ├── python/
    ├── results/
    ├── sh/
```

The sub-directory `results` already contains the results corresponding to the experiments reported in the manuscript.

```
plots/sh/charts.sh
```

This will generate the `.eps` image files in the `plots/gnuplot/figs` subdirectory. In order to generate Figures corresponding to new runs of the experiments (as per the instructions in this guide), the `.results` files generated at the root directory of this project must be copied to the `plots/results` subdirectory, and then the `mergefiles.sh` script in the `plots/sh` must be executed before generating the charts with `charts.sh`.

- Next Technical Steps

- Build and publish a Python package that can be easily installed with `pip`.
- Integrate **CORE-SG** into existing density-based methods already broadly available, *e.g.* DBSCAN, HDBSCAN*, OPTICS.

- Troubleshooting

`fatal error: numpy/arrayobject.h: No such file or directory`

Create a symbolic link from the Numpy installation in the Python version being used (e.g. 3.8) to `/usr/include/numpy`

```
sudo ln -s  /usr/local/lib/python3.8/dist-packages/numpy/core/include/numpy /usr/include/numpy
```

References

[1] Ricardo J. G. B. Campello, Davoud Moulavi, Arthur Zimek, Joerg Sander. Hierarchical Density Estimates for Data Clustering, Visualization, and Outlier Detection. ACM Trans. Knowl. Discov. Data 10(1): 5:1-5:51 (2015)

[2] Julia Handl, Joshua Knowles. Cluster generators for large high-dimensional data sets with large numbers of clusters. https://personalpages.manchester.ac.uk/staff/Julia.Handl/generators.html

[3] Antonio Cavalcante Araujo Neto, Jorg Sander, Ricardo Campello, Mario Nascimento. Efficient Computation and Visualization of Multiple Density-Based Clustering Hierarchies. IEEE Transactions on Knowledge and Data Engineering (TKDE), 2019.