Project final report - Road Graphs

Carraro Eddie - 2121248
Gava Mattia - 2130893
Giacomin Marco - 2119285

Title: Exploring the correlation between a country’s history/geography and its road network graph

This reports details our attempt at exploring mesurable topological differences between road networks. We attempted to extract as many suitable features from each network as possible to be used by a clustering algorithm. Despite the numerous computational hurdles we faced, which forced us to downsize our project significantly, we managed to detect some interesting patterns in the data which neatly groups some closely related countries, along with highlighting some unexpected properties of seemingly uncorrelated one.

Motivation:

- Intuitively, countries with different histories, geographies and development timelines should have differently structured roads
- The road graphs of european countries, with thousands of years of gradual development history, should have more in common with each other rather than the ones found, for example, in the US, given their relatively rapid development
- Do measurable graph properties actually reflect this intuition?
- Are real-life road networks distinguishable from random networks?
- Data: the following graphs available from public repositories (snap.stanford.edu and networkrepository.com)
  - road-italy-osm (~7M nodes, 7M edges)
  - road-germany-osm (12M nodes, 12M edges)
  - road-great-britain-osm (8M nodes, 8M edges)
  - road-belgium-osm (1M nodes, 2M edges)

- road-luxembourg-osm (115k nodes, 115k edges)
- road-netherlands-osm (2M nodes, 2M edges)
- road-road-usa (24M nodes, 29M edges)
- road-roadNet-CA (2M nodes, 3M edges)
- road-roadNet-PA (1M nodes, 2M edges)
- road-roadNet-TX (1M nodes, 2M edges)
- road-asia-osm (12M nodes, 13M edges)

Methods used:

- Problem: Compute average centrality scores, motifs, clustering coefficient of road graphs and use them as features for clustering of said data. Confront features of real data with random graphs
- Algorithms: We attempted to use exact algorithms as seen in class for the calculation of features, with approximate solutions as a fallback. Unfortunately, given the size of studied graphs, even polynomial algorithms above a linear or quasi-linear complexity proved to be too expensive for our hardware to run. Given this, we limited ourselves to such algorithms for the feature extraction, which meant forgoing scores such as centrality.

Performed experiments:

1. We used the python language together with the graph analysis library networkx (networkx.org) which implements many common graph operations
2. Machine for experiments: We interchangeably used the personal computers of the members of this group (a linux pc with 8gb RAM, a windows pc with 8gb RAM, a windows pc with 12 gb RAM. None posess a GPU). We also attempted to use Google Colab, but its computing capabilities were on par with our hardware
3. Experiments:

a. For each graph: compute graph density, assortativity and average clustering coefficient

b. Save each computed statistic as a feature in a multi-dimensional space describing each graph
c. Generate some random graphs based on the actual road networks and compute the same statistics for them (aborted due to sickness of one member)
d. Employ a clustering algorithm to group together road networks that share similar characteristics
e. Verify if states with similar histories and geographies are grouped together in the same cluster and if they are distinguished from randomly generated networks

How the project evolved

For what concerns the **features to find**, doing some trials of execution in our machines, we noticed that the time of execution when finding the features we wanted to find first (closeness, betweenness and clustering coefficient) was very high when using the *networkx* package. So, we tried using *igraph* which is written in C (and so it should be faster than *networkx* for sure) and we noticed an improvement in the execution time, but it was not enough for the bigger graphs we have to deal with (which have a number of nodes and edges of the order of millions). To solve this issue, we attempted to use the academic cluster **Blade** and **cugraph** library in order to try the execution using a more powerful machine (which unfortunately didn't provide any speedup), finally settling on computing features with linear or quasi-linear complexities proportional to number of nodes in graphs.

For **random graphs**, the choice of model is crucial to accurately simulate the characteristics of the graphs we are analyzing. Specifically, we need to focus on random graphs that are connected. In fact, we verified that even the road network of Italy is connected, which suggests that the major parts of the country are connected with the islands. This connectivity might be achieved through existing marine routes, such as those in the Messina Strait.

Additionally, the nodes in these graphs should only be connected if their real-world distance is within a reasonable range.

After conducting some research, we identified a few random graph models that may be useful for this task:

- **The Watts-Strogatz Model**: This model generates graphs with small-world properties, which are particularly useful for representing networks where nodes are grouped closely together and connected to one another.
- **The Random Geometric Graph**: In this model, N nodes are placed randomly in a metric space, following a specified probability distribution. Two nodes are connected by an edge if their distance is within a certain range, such as a specified neighborhood radius, r. The only problem with this approach is that it connects every pair of nodes under this radius, so the randomness factor is lost.

Unfortunately even this part of the project was left in a theoretical state, as the models we attempted to implement for the graph generation didn't terminate in a useful time. Of note: we predict that a random graph generation model will need to produce graphs that are remarkably close to the original ones in terms of features, as a random graph that's too distant from the originals will always be separated in its own cluster during the partition study and fail to produce interesting data.

Our **final evaluation** of the similarity between the road networks was therefore only performed on the features extracted from the original examined datasets. We extracted the three previously described features (density, assortativity, average clustering coefficient) and used the scikit-learn implementation for the KMeans problem to produce a clustering. Since the algorithm needs to be provided with a number of clusters to divide the data into, we also employed silhouette scoring to identify which k value produced the most distinct clusterings.

The final results

To conclude our work, despite the significant missing features due to lack of computational power, we computed the achievable features on our hardware, producing a vector of 3 elements for each graph, and repeated the KMeans clustering as described above for various numbers of clusters. Comparing the silhouette scores, the best values for k were:

- **K=2** with an average silhouette score of 0.73
- **K=5** with an average silhouette score of 0.46
- **K=6** with an average silhouette score of 0.53

These produced the following clusterings:

- **K=2**: Asia and Italy in one cluster, every other network in the rest

- **K=5**: Asia, Italy and Great Britain each in their separate cluster, followed by a cluster for the two american states of CA and PA, with the larger USA one and the remaining european ones in the last remaining one
- **K=6**: Asia, Italy and Great Britain in their individual cluster (plus the CA+PA one) as before, with the difference of having the USA and germany grouped together and the three geographically related networks of the netherlands, belgium and luxembourg in the last one

Some insights can be gained from this:

- Asia and Italy appear to have the most distinct networks among all the examined ones, possibly due to historical circumstances or different data gathering methods. Additionally, they're quickly separated by the introduction of additional clusters, meaning that they're quite different from each other too
- The american and german network are consistently grouped together. They share a relatively regular grid-like pattern that gets more sparse as it traverses the country (from east to west in the first, from south to north in the second)
- The canadian and pennsylvanian networks are sufficiently similar to be clustered together, as expected. The same is true for the netherlands, belgium and luxembourg. These are the networks that confirm our initial hypotheses the most.

Possible future work

The interesting results found at the end of our work suggest that a more thorough examination of the data (with additional features and graphs and the implementation of random graph generation) could discover more hidden properties in the road networks. Particularly, our originally intended scores (closeness/betweenness centrality) could be reintegrated, along with motif analysis and clustering of the graph themselves, to examine whether some parts of the region bear more similarity than others to other countries.

This methodology could also be easily applied to studying other related sets of graphs, particularly if they are numerous and present a reduced number of nodes.

Contributions and share of the work

- **Marco (1/3):** Suggested idea for the project, initial proposal, downloaded datasets, closeness centrality, networkx built-in features calculator, final report
- **Mattia (1/3):** Clustering coefficient, random graph generation, final computations and final report writing
- **Eddie (1/3):** tried different libraries for the execution, tried Blade and Colab to speed-up the executions, helped with reports and proposal

Repository structure

Our work is collected into a repository hosted at https://github.com/jolitti/road_graph. The code (in the src/ directory) is organized to allow the separation of the feature extraction and feature study sections of the processing. The src/featurecalc.py python script is written to allow easy extensibility with other graph feature calculators (stored in src/feature_calculators/). The datasets are not memorized in the repository and have been instead individually shared between group members. There are also various utility files (such as src/time_estimator.py) that have been used during development to produce an estimation of feature extraction time cost. The last file to be written, src/cluster.py, is responsible for clustering the extracted features and producing the information on the dataset similarities we discussed above.

References:
https://chih-ling-hsu.github.io/2020/05/15/Graph-Models#watts-strogatz-model
https://en.wikipedia.org/wiki/Random_geometric_graph
https://github.com/jolitti/road_graph