Network Analysis Project

Andreas Klingenbrunn,
Johan Ødum Lundberg,
Mourad Karib,
Nicolai Herforth Rendtslev

December 2018

1 Problem

Originally, the focus of this project was on the research question:

"*What is the relationship between the different neighbourhoods of Berlin and the traffic between them?*"

However, the preliminary data analysis did not give any relevant insight into the relationship between the communities. It was therefore decided to work with the network as a whole. This lead to the research question:

"*How does Berlin react to the removal of edges and how well does it redistribute the traffic throughout this process?*"

2 Introduction

This report explores a traffic network of Berlin with a focus on the resilience of the network, specifically looking at how well the city is at redirecting cars in a scenario where roads are closed off. The network is represented as a directed graph, according to the direction of the roads in the city. The data is divided into six different parts, each representing their own neighbourhood of the city. The resilience is a measure of how much the graph can be changed or reduced before it loses some property or connection. Regarding the properties, the graph has two attributes: 'Capacity' which is the maximum number of cars a road can have, and 'Cars' which is the current number of cars on the road[1]. It is expected that as more edges are removed, more cars will get stuck in traffic and the number of components will increase. This will be the focus in the next sections.

3 Descriptive statistics

The following section explores properties of the graph 'Complete Network' which is a merger of the other networks(see Table 1). The graph has a total of 12.981 nodes and 33.217 out of 1.103.335.872 possible edges, making the average degree of the network equal to 2.55. The low density of the graph which is 0.02 means that the chance that two random nodes connects by an edge is only 0.02 pct. When two random nodes do connect, there is roughly a 50 pct. chance that they are connected both ways, as reflected by the reciprocity. This could be considered a low number as most roads usually connect both ways, however, as it turns out, many roads in Berlin are actually one way only.[2].

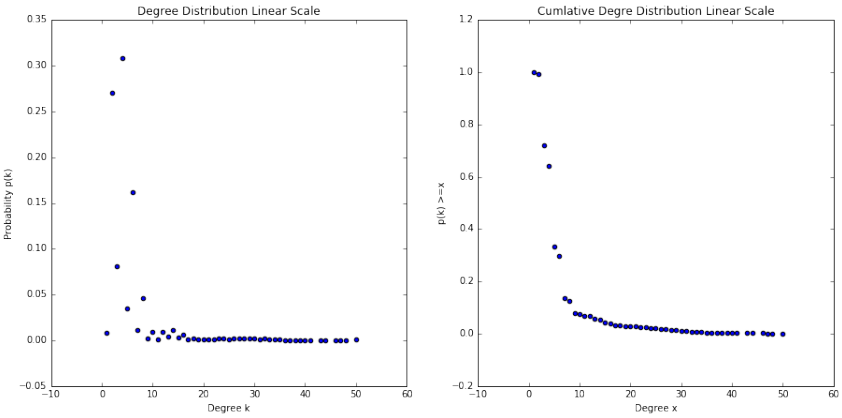
The average clustering coefficient(cc) indicates that ten pct. of the time the neighbours of a given node are connected to each other. The graph is one big weakly connected component composed of 129 strongly connected components, one of which connects 12.853 nodes, the other 128 are single nodes that do not connect to anything.

[1] 'Cars' is an attribute created for this project and is not a part of the actual data. It is set to a quarter of a road's capacity
[2] See Appendix A

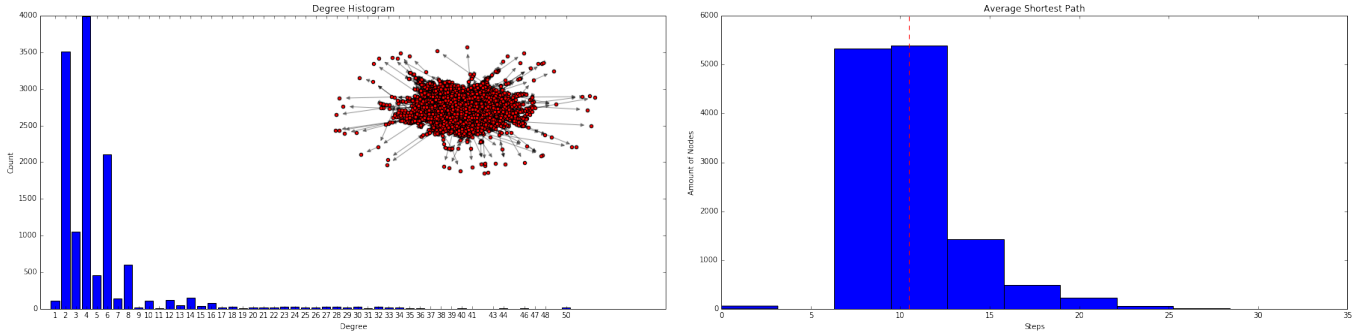
Table 1: Neighbourhood - Network Statistics

Network	Nodes	Edges	Density	Reciprocity	Avg. Clustering Coefficient
Berlin Center	12.981	28.370	0.02	48.64	0.21
Mitte Center	397	871	0.55	52.12	0.20
Mitte-Friedrichschain	974	2184	0.23	52.47	0.19
Friedrichshain Center	224	523	1.05	56.21	0.18
Prenzlauerberg Center	352	749	0.61	52.07	0.21
Tiergarden Center	359	766	0.60	44.65	0.20
Complete Network	12.981	33.217	0.02	49.78	0.10

The probability of a node having a degree higher than eight is very low while the probability of a degree of three or less is very high, this is visualized in cumulative degree distribution plot on the right. When checking which function would be the best fit to this distribution the power law is ruled out as the R value is -.95 and the p value is 2.24, this does however not rule out an exponential fit. The two plots clearly show that most nodes have a low probability of connecting to many nodes, and considering that there is 12.981 nodes this explains the density



of 0.02. The same pattern is visualized in figure 1, where most nodes have a degree of eight or less and the majority have a degree of either two, four or six. To sum it up; most nodes only connect to six nodes or less, their neighbours often connect back to them and they rarely connect to their neighbours' neighbours, perhaps much like it is in the real world. The low density of the graph results in the average shortest path of the graph being about eleven steps, meaning two random nodes will have to traverse eleven edges to connect. This does not make the network very desirable as the flow of traffic will be long, inefficient and costly. Costly because crossing an edge takes time and requires fuel for the car, and the more time it takes, the more inefficient it gets.



(a) Degree Histogram

(b) Average Shortest Path

Figure 1: Degree Distribution & Avg. Shortest Path

4 Methods

The resilience of the network can be shown by ranking the edges by betweenness centrality. It is carried out by creating a random walker that traverses the graph in steps of eight over 5000 iterations. At each iteration, the walker starts at a random node and visits one of the node's neighbours, if any exists. This makes it possible to rank the edges of the network from most to least important based on betweenness centrality. Once the random walk is finished, the edges are removed one by one, starting with the most important one, this causes the graph to split into several components.

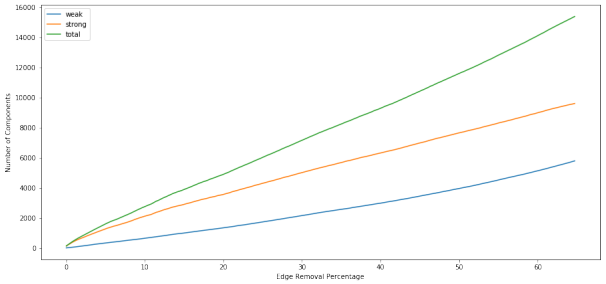


Figure 2: Edge removal sorted by edge betweenness not including redistribution

As mentioned earlier, each road has two properties; Capacity and Cars. To test how well the network redirects traffic, the data was cleaned such that all edges had a capacity and car value. Also, the roads that had undefined capacity was set to the mean value. When an edge is removed its cars are redistributed equally among the starting node's neighbours but only among those neighbours that have a path to the cars' destination; the end node. If a neighbour fails to receive extra cars, those cars are redistributed to the other possible choices of edges.
As seen on the two graphs on the right, the growth of components increases as more edges are removed. At each iteration, there is a chance that the removed edge will create a new weakly connected component, or it might create a new strongly connected component by disconnecting that component from the rest. The process continues until the graph consists of nothing else but small components, though if there exists an edge which the random walk did not visit, it will not be removed. This is the case in the plots where the edges removed stops at about 70 pct.

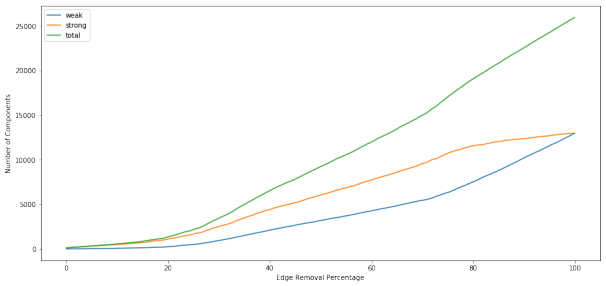


Figure 3: Unsorted edge removal not including redistribution

The idea behind this process was to get insight into how the network was put together when removing the most central edges first. Figure 3 clearly shows that removing edges randomly results in the component growth starting slower, which could be due to the edges that are removed at first barely have any relevance in connecting our big component. On the other hand, figure 2 displays a quick dissolution of the graph's components from the very beginning of the edge removal process. The total number of components is linear, since the removal of an edge, in most cases, cannot generate more than one new component.
The second attempt at testing the graph resilience was to redistribute the cars throughout the network as roads were closed. This was done to see how well our network redistributes traffic in case of a given road being shut down.

A road is considered to be shut down when its edge representation is removed or when the amount of cars for a given road exceeds its capacity. If a car suddenly cannot get to the desired location, it is considered stuck. The main idea with counting stuck cars in relation to redistributions and edges removed was to see how long it would take before too many cars get stuck. The limit is set to 50%, so when half of the total amount of cars are stuck, the network is considered out of order. Looking at Figure 4, we can actually see that the amount of stuck cars is not necessarily linear. Instead the evolution is a more bent curve which then grows into what looks more like an exponential growth. Looking at the plot itself, it is seen that half of the cars are stuck after around 7000 redistributions. This is considered a rather good result seeing as it implies that the network is doing a good job at redistributing cars throughout the city which also makes good sense in relation to the kind of network we are working with, since if a road closes down in a city, the city should be able to have other paths leading to the destination.

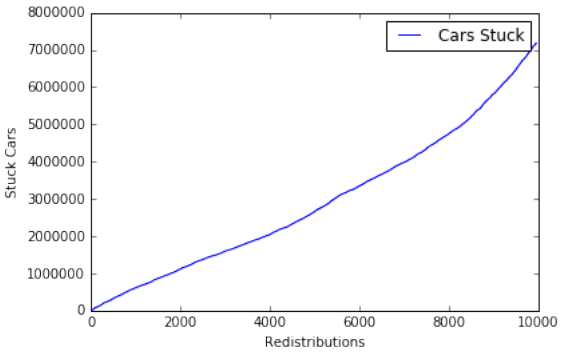


Figure 4: Redistribution vs Stuck Cars

5 Evaluation

The following section discuss the limitations of the applied methods and answers the research question.

5.1 Error Analysis

The cars are redistributed in equal proportions and thus the method does not consider the difference in edge capacity. Obviously a better solution would be to assign most of the cars to the edge with the highest capacity and thus delaying the breakdown of the other edges.

Furthermore, 'Cars' was not an original part of the data, hence why any analysis based on this attribute should be considered biased. The cars on each edge was, as mentioned previously, set to the quarter of the edge capacity, this is of course also naive and does not represent any real scenario. The purpose was only to test the hypothetical scenario seeing as there were no resources to replicate a realistic case.

As the algorithm only determines the number of components after a redistribution, any case where the redistribution fails could result in the removal of several edges before recalculating the amount of components. This presents a minor inaccuracy in figure 2.

5.2 Conclusion

In conclusion the results show that the city of Berlin initially splits up into smaller, mostly strongly, connected components when edges are removed, if the edge removal process is starting at the most central edges in the network. Regarding the redistribution it is shown that the network does a good job of redirecting cars through the city when central edges are removed. This is supported by the fact that it took around 70% of the total amount of redistributions before half of the total amount of cars were stuck.

6 Appendix A

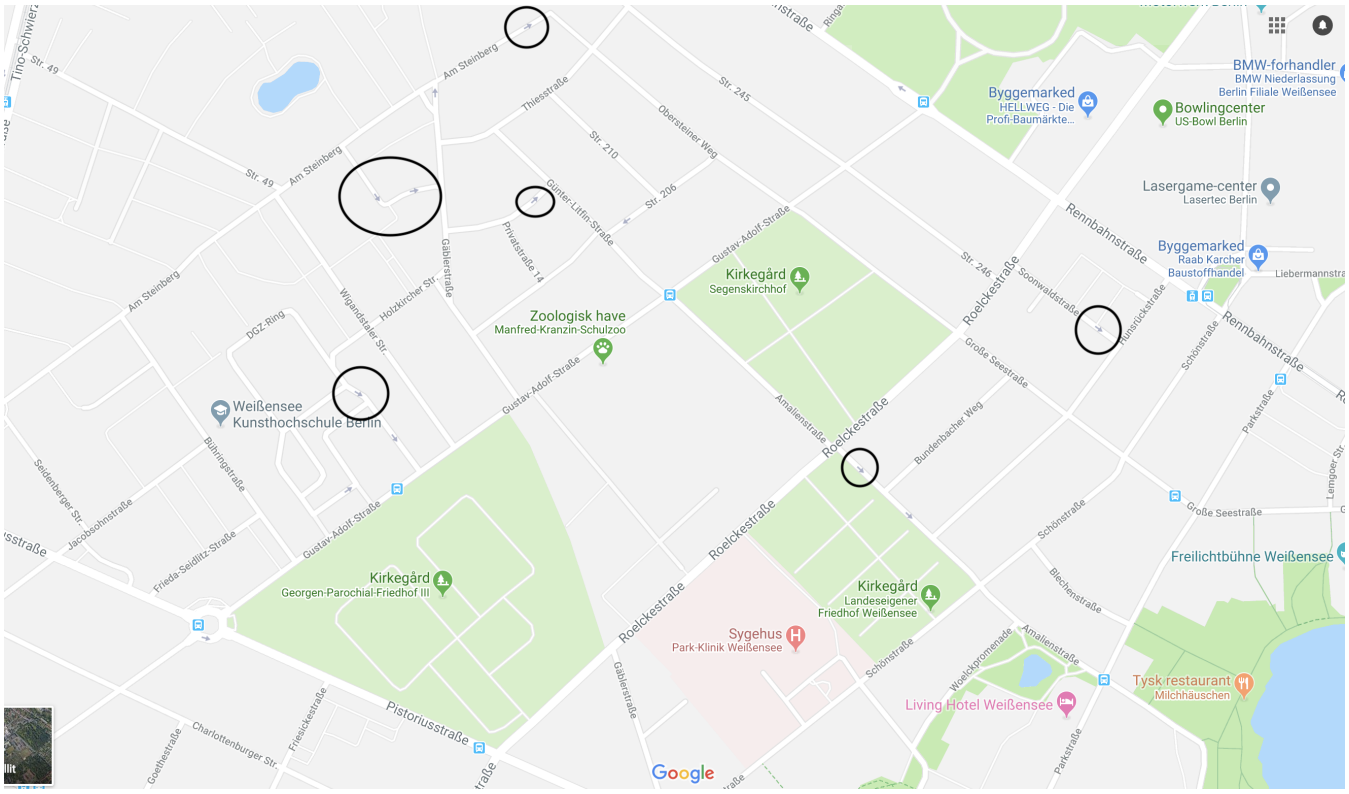


Figure 5: One Way Roads - Berlin

7 Appendix B

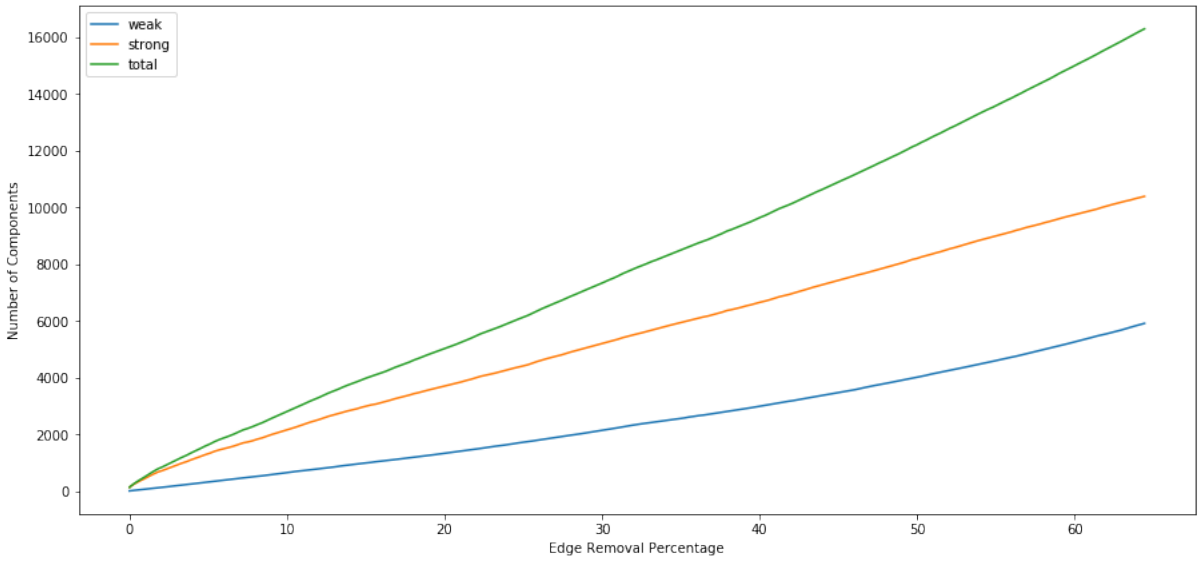


Figure 6: Edge removal sorted by betweenness centrality including redistribution

8 Appendix C

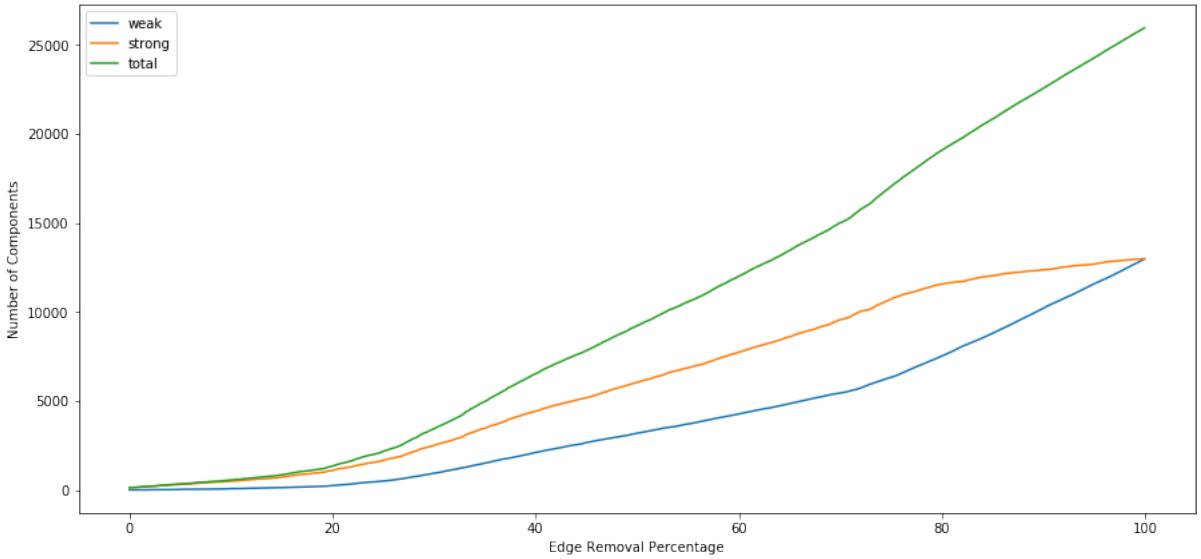


Figure 7: Unsorted edge removal including redistribution

9 Appendix D

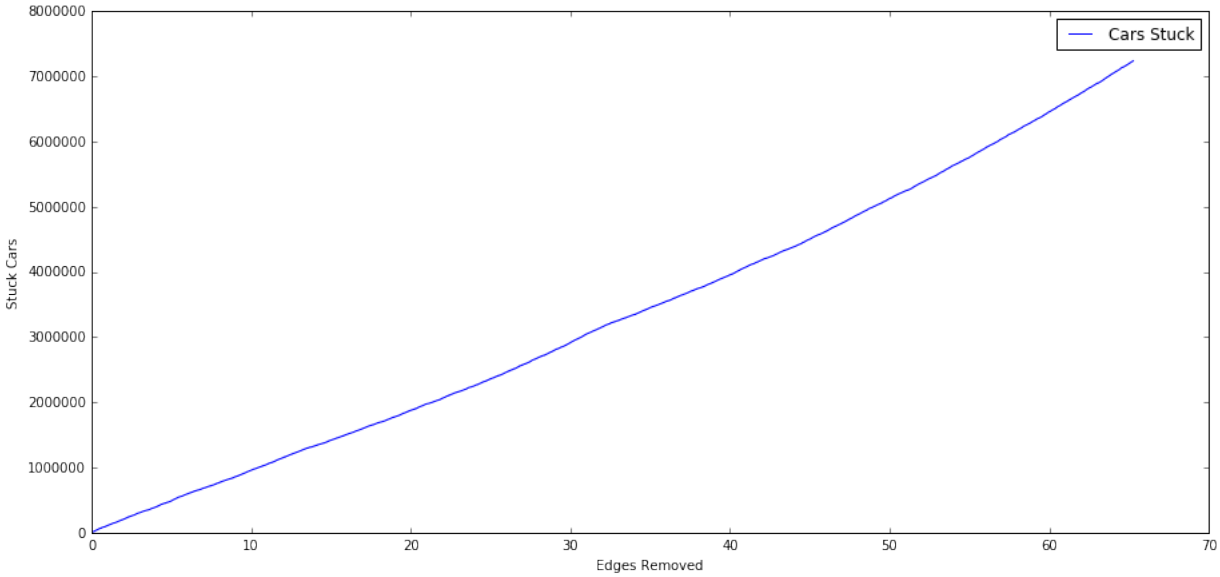


Figure 8: Redistribution - Stuck cars vs. # edges removed

10 Appendix E

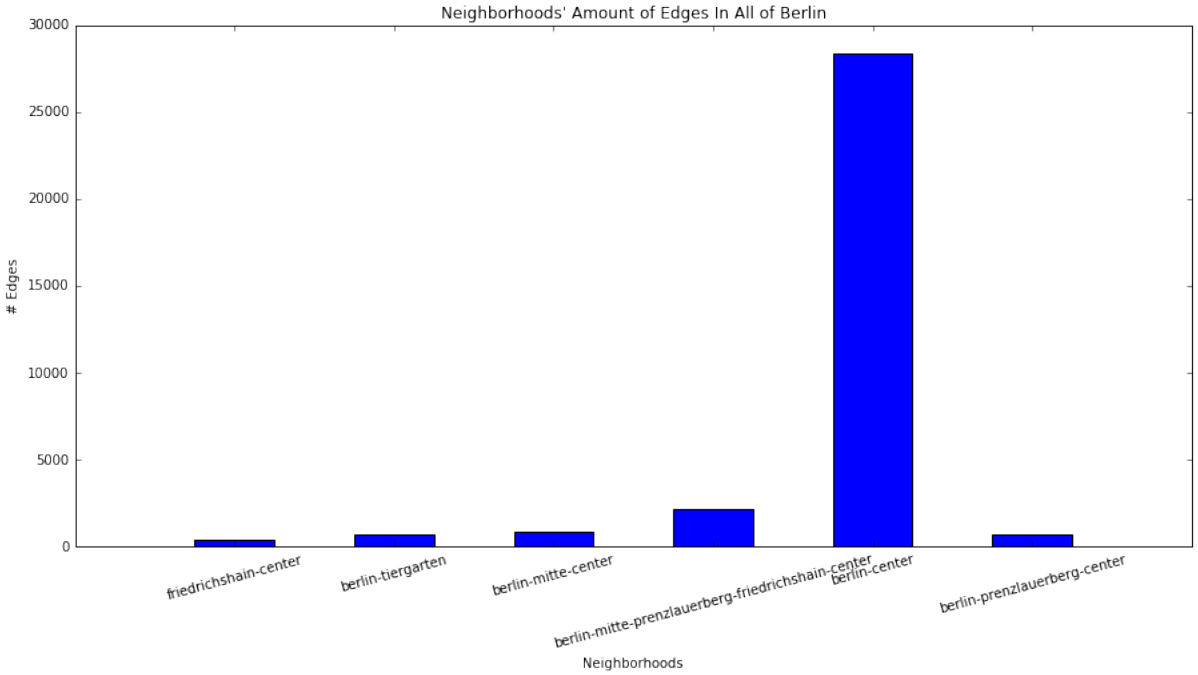


Figure 9: Edge count for all areas of the network