Graph Signal Processing in the Data Analytics of Road Congestion

CONCEPT OF OPERATIONS

REVISION – 1

4 February 2019

CONCEPT OF OPERATIONS FOR

Graph Signal Processing in the Data Analytics of Road Congestion

Team 15 – GSP

Approved By:

_____

Project Leader Date

_____

John Lusher II, P.E. Date

_____

T/A Date

Change Record

Rev.	Date	Originator	Approvals	Description
1	2/4/19	Arash Abdolahzade		Draft Release

Table of Contents

List of Figures

1. Executive Summary

This project aims to tackle the present challenges of traditional machine learning algorithms for constructing predictive models to make predictions on traffic congestions and analyzing congestions' correlation. The proposed solution is to validate a novel approach, namely graphical signal processing, and confirm its ability to leverage the power of immense datasets (big data), which can be gathered from different sources, for building accurate prediction engines with high performances. The validation dataset if provided by the Zurich public transportation system; the quality of the dataset will be evaluated, and the data processing methodology will be developed. Also, this project will seek to modify the existing interface between the GSP (graphical signal processing) algorithm and the dataset to further simply the usage of the model. Finally, the model will be built using the GSP principles, and the performance will be evaluated based on the data generated throughout 2019.

2. Introduction

The purpose of this document is to showcase the potential benefits of applying graph signal processing techniques to building predictive models for traffic congestion. Accurate real-time predictions on traffic data is crucial for optimizing traffic flow in case of an accident, further improved path planning, and improvement of the road network design.

2.1 - Background

Data Science is an ever more crucial subject, especially since we are in the dawn of big data. Big data's typical definition includes the "three 'V's", which Ralph Jacobson, industrial data analytics portfolio manager at IBM, explains them to be, **v**olume, the fast-growing data in the order of petabytes, **v**elocity, the time-sensitivity, like fraud detection and stock market data, and lastly, **v**ariety, which points to the combination of structured and unstructured data that is available [1]. Developing real-time predictive models for traffic prediction is a vital aspect of formulating a robust traffic management strategy and optimizing the allocation of resources.

Dr. Nick Duffield, director of the Texas A&M Institute of Data Science, along with Dr. Krishna Narayanan, professor at the ECE (electrical and computer engineering) department at Texas A&M, in collaboration with students from the ECE department have developed a novel method for building predictive models to predict the spread of the traffic congestion clusters in road networks using graphical signal processing (abbreviated, GSP). The model was validated by using the dataset gathered from the Dallas-Fort Worth highway network, which was provided by the Texas A&M Transportation Institute. Dr. Duffield wishes to extend the application of mentioned GSP based model to other datasets, namely, the Zurich public transportation open dataset, as well as resolve some of the challenges that GSP based models face.

Practical usage of the predictive models for traffic management is highly dependent on accuracy of the model, as well as completion of timely predictions. Many machine learning algorithms can build such models; some of the existing methods, cited by Hasanzadeh et al. [2], include support vector regression (SVR) [3], gaussian process (GP) [4], and deep learning [5]. However, the methods mentioned above can prove to be either too computationally complex or fail to utilize the prior (to the congestion) data efficiently [2]. GSP model is aiming to proposition a solution to said issues. Although, to use the approach developed by Dr. Duffield's team, the prediction problem must be formulated such that a directed graph is constructed, where the vertices are some defined points of a given road, and the edges indicate whether it is possible to reach a vertex from the current one via that rode. Each of the vertices consist of a signal (here, average speed of vehicles on the road), which is to be predicted. Such rigid requirements for preparing the data renders current GSP models unpopular.

The dataset for validation is provided by VBZ (**V**erkehrs**b**etriebe **Z**ürich, translate to Zurich transport service, the company that provides bus for means of public transportation); it consists of a very large table-oriented dataset, reaching about 12 gigabytes of data, per given year. Making accurate predictions in time, can be the prove to be the pinnacle of the intelligent data, especially with emerging technologies like self-driving vehicles and intelligent cities. This project fits the definition of big data very well as the **v**olume of the data is massive, predictions on the data must be done in high **v**elocity, and the data is generated from **v**arious sensors and are asynchronous.

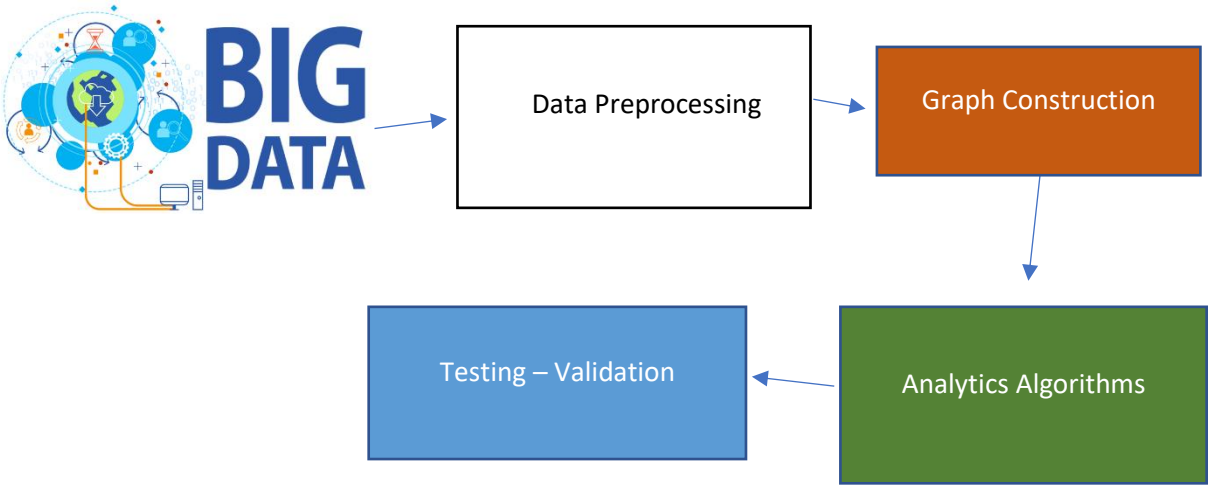


Figure 1 – High Level Representation of the Application Flow

2.2 – Overview

VBZ's open datasets consists of approximately three years of historic data about the movement of buses, including attributes like, line number, direction of travel, date, departure and arrival times, as well as GPS and GIS data. Datasets are asynchronous and are gathered daily and published weekly. Capturing data is still ongoing and the website is frequently updated; each week, approximately 250 mega-bytes of data is captured. As with any machine learning problem, there is a need for a test/validation set, as well as a training set, so, historic data from 2015 to 2018 will be used to train, and data captured in 2019 will be used to test the model. New information will be preprocessed to match the correct format required by the GSP model.

As a stretch goal for this project, once the preprocessing is done, and the GSP model is created, a recurrent neural network model, which has proven to be efficient when processing time-dependent data, can also be trained, and the performance of the two models can be compared.

2.3- Referenced Documents and Standards

[1] Jacobson, Ralph. “Big Data Spans Four Dimensions.” IBM Consumer Products Industry Blog, IBM, 23 Apr. 2013, www.ibm.com/blogs/insights-on-business/consumer-products/2-5-quintillion-bytes-of-data-created-every-day-how-does-cpg-retail-manage-it/.

[2] Hasanzadeh, Arman, et al. “A GRAPH SIGNAL PROCESSING APPROACH FOR REAL-TIME TRAFFIC PREDICTION IN TRANSPORTATION NETWORKS.” Electrical Engineering and Systems Science > Signal Processing, Cornell University, 19 Nov. 2017, arxiv.org/abs/1711.06954.

[3] G. Ristanoski, W. Liu, and J. Bailey. Time series forecasting using distribution enhanced linear regression. In Pacific-Asia Conference on Knowledge Discovery and Data Mining, pages 484–495. Springer, 2013.

[4] J. Zhou and A. KH. Tung. Smiler: A semi-lazy time series prediction system for sensors. In Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, pages 1871–1886. ACM, 2015.

[5] Y. Lv, Y. Duan, W. Kang, Z. Li, and F. Y. Wang. Traffic flow prediction with big data: A deep learning approach. IEEE Transactions on Intelligent Transportation Systems, 16(2):865–873, 2015

3. Operating Concept

3.1 - Scope

The scope of this project will be including preprocessing and applying the GSP method developed by Dr. Duffield's team. The deliverables of this project will consist of:

- a proposed methodology for preprocessing and data cleaning of the VBZ raw datasets
- developing a methodology for defining the graph required for the GSP model
- applying the GSP based method proposed by Dr. Duffield
- capturing performance metrics of the model

Ultimately, as a stretch goal, an off-the-shelf predictive model, namely a recurrent neural network model will be trained, and for validation, the performance metrics and the two models will be compared.

3.2 - Operational Description and Constraints

The model will be able to predict the traffic congestions based on the real time input data. Any dataset that match the characteristics of the VBZ dataset, namely, inclusion of arrival and departure times is going to be able to be preprocessed and used as training data for a GSP predictive engine. As a hard constraint, the GSP model's input data must be transformed into a directed, weighted graph. Also, due to the described envisioned use cases, the predictions must run in a reasonable time.

3.3 - System Description

As with any big data/machine learning project, a series of subsystems are required to construct a final model. This project will consist of two major subsystems detailed below,

Data Preprocessing: Data preprocessing is the major part of any data science project, and it usually takes the most amount of time to complete. Data preprocessing is a series of steps to prepare a workable framework for the analytics algorithm to run. In many cases, the accuracy and performance of the model is heavily dependent on the preprocessing of the data. It consists of:

- **Data Cleaning**: This stage seeks to unify the dataset by filling in missing values or eliminating that datapoint. It is particularity challenging since the VBZ dataset signals are time dependent and elimination of datapoints can have immense ramifications.
- **Data Transformation and Reduction**: Data transformation consist of performing mathematical transformation to reduce the chance of introducing meaningless bias generated from the relative differences in values, the datapoints are normalized to a certain standard. Also, to lower the dimension of the data, which in turn reduces the

needed computation, data reduction techniques, namely PCA (principle component analysis) will be performed.

The main challenge of this project lies in construction of a graph with adherence to the points discussed above.

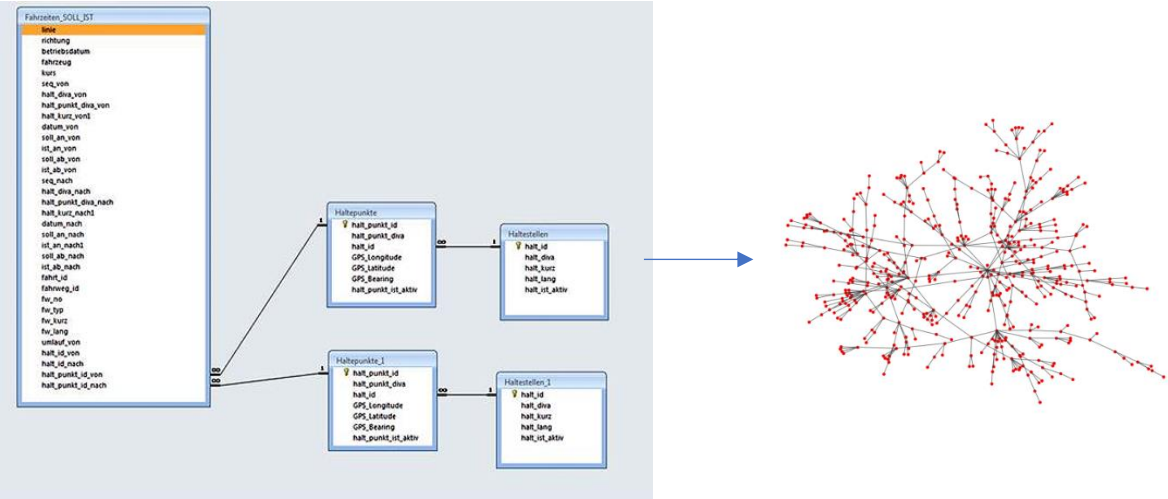


Figure 2 – Conforming DB to a Desired Data Structure

Analytics Algorithm and Performance Metrics: Once the graph is created, the next step is to apply the GSP based model technique; the algorithm API (python) will be used to run the analytics algorithm on the data. However, minor tweaks to the existing methods are to be expected. For example, in this section, code will be developed to decompose the initial graph into smaller disjoint graphs. Once the model is trained, the performance of will be evaluated using a split on the available dataset.

3.4 - Modes of Operations

Since the intent of this project is extension and validation of theories of the proposed GSP-based model to the VBZ dataset, any discussion about modes of operation will have to be an expectation and suggestion. Modes of operation for any application based on the findings of this project can be training, validation/testing, making predictions; training can be attributed training a new model, i.e. when a new network is constructed, and the model is not trained yet. Testing and validation can be engaged to periodically test the response of the system against an artificially simulated distress and study the outcome. And finally, prediction mode allows real-time predictions on the traffic signals (such as congestion, average velocity…).

3.5 - Users

An analytics engine can be created based on the theories that this project seeks to validate. Generally, any decision that involves spatio-temporal data can benefit from accurate real-

time predictions. Since some of the theories used in this model are mathematically involved, a solid background in mathematics, signal processing, machine learning, specifically PCA (principle components analysis), and graphs is required; however, a high-level analytics engine can be considered, where a well-designed API allows for “plug-and-play” usage of these theories, although it most likely will be limited to the VBZ dataset.

3.6 - Support

A high-level documentation will be provided that outlines the low-level description of the steps that was required to complete the preprocessing of the data, as well as construction of the graph.

4. Scenarios

Path planning optimization and safety

Companies that offer location services products can benefit significantly from real-time predictive models to optimize their path suggestion algorithms. Smart path planning is crucial for autonomous vehicles, as it allows for more intelligent routing; that can lead to an ever so harmonious flow of traffic, and especially with the introduction of 5G, and recent advancements in self-driving vehicles, the GSP model can future-proof the transportation system.

Cab/ Ridesharing fair optimization

Ride sharing companies, like Uber, can integrate this method in their system to adjust fares to optimize revenue. Gaining insight from data and having access to predictions about the potential delays can provide path planning agents to suggest well informed alterations to optimize the time and distance of ride. The public transportation can also utilize the applications based on this method to find oversights in scheduling and find a supplemental solution.

Road Design – Service Optimization

Studying the traffic patterns, combined with gaining insights from the state the network when an unforeseeable event occurs, i.e. an accident, can lead to well-informed adjustments to the network and traffic flow. Also, leveraging a data-driven approach to monitoring the traffic flow allow for public transportation services to be more reliable, and potentially drive the cost of services down.

5. Analysis

5.1 - Summary of Proposed Improvements

- The Zurich public transport system will employ the GSP based model to reduce their traffic pattern predication models. The system will be benefiting from the data generated for free, or next to no cost, to improve their service. Public transportation users will experience less delays. VBZ can market its service as a superior alternative to personal means of transportation, which can result in less emissions and ultimately, pollution reductions.
- With the ability to forecast traffic signals (i.e. average vehicle velocity in a given node, in this paper, “traffic signal” is not to be confused with traffic control signals), emergency vehicles can benefit from enhanced route selection.
- VBZ can start to serve as a pioneer in intelligent transportation system; other public transport systems can adopt a similar pattern to optimize their networks. As the network increases its size, the predictions can be made with higher accuracy, as there is more data available to train and test the models.

5.2 - Disadvantages and Limitations

GSP models can compete with classical machine learning algorithms in complexity, as in general, they are less computationally expensive. However, the current GSP models can only be applied on highly processed datasets; constructing a graph, along with formulating the traffic signals that is in question. Also, the preprocessing method that will be employed to construct the graph from the VBZ dataset, may not necessarily apply to any other dataset.

A common limitation that many big data problems face is incorporation of data streams from different sources which can prove to be extremely challenging. Unstructured datasets are the hardest to conform to a specific format, and the issue of synchronization impacts the preprocessing stage significantly.

5.3 - Alternatives

There has been an adequate number of models implemented to solve traffic congestion prediction. However, as mentioned in the introduction, when dealing with a massive dataset, such as the VBZ dataset, model selection become a delicate matter. Many of the traditional machine leaning models are too computationally expensive. Sequence models like, LSTM (long-short term memory) and recurrent neural networks, can serve as an alternative to applying GSP based models to large datasets. Deep learning-based models are viable especially with the utilization of computation parallelism, and the proven exceptional performance of RNN models. Alternatively, the methods of GSP can be modified to relax some of the hard constraints of the GSP model.

Graph Signal Processing in the Data Analytics of Road Congestion

FUNCTIONAL SYSTEM REQUIREMENTS

REVISION – 1

17 February 2019

FUNCTIONAL SYSTEM REQUIREMENTS

FOR

Graph Signal Processing in the Data Analytics of Road Congestion

Team 15 – GSP

Approved By:

_____

Project Leader Date

_____

John Lusher II, P.E. Date

_____

Minjeong Kim Date

Change Record

Rev.	Date	Originator	Approvals	Description
1	2/17/19	Arash Abdolahzade		Draft Release

Table of Contents

List of Figures

List of Tables

1. Introduction

1.1. Purpose and Scope

The Graph Signal Processing (GSP) is an alternative analytics method to the state-of-the-art machine learning algorithms, namely Long-Short term memory (LSTM) recurrent neural networks (RNN). The advantage of GSP over other methods include retaining and using the spatio-temporal (belonging to both space and time or to space-time) dependencies, and faster training time. The scope of this project covers the preprocessing a dataset containing three years of historical data of the Zurich public transportation (VBZ), constructing the network graph required for the analytics algorithm, and applying the GSP model to get some results. The figure below shows the relations between the different components of this project.

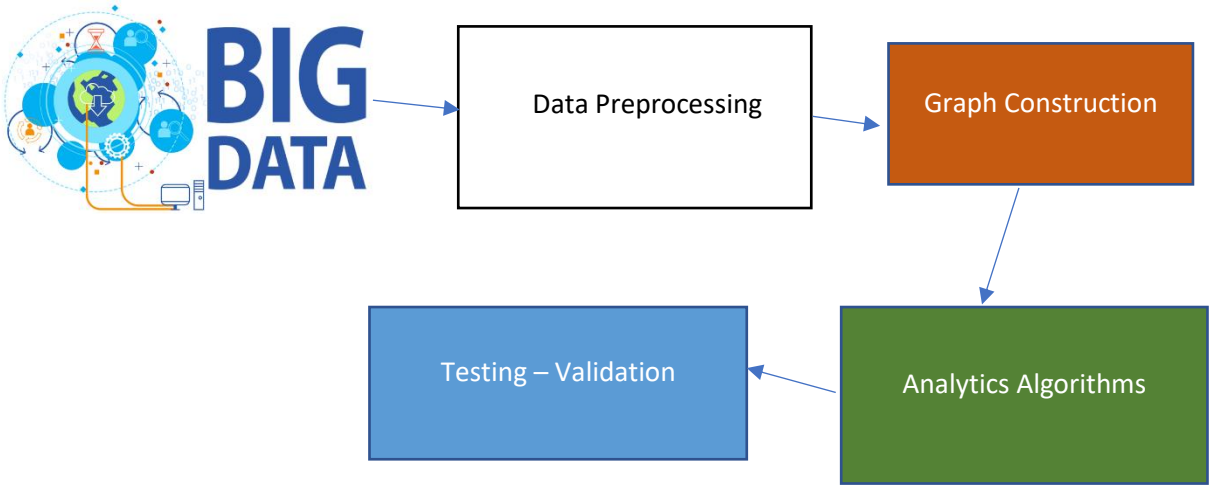


Figure 1. GSP Model Overview

The existing GSP method for dealing with big data requires some hard constraints to be satisfied, such as the input graph must be able to be partitioned with a moderately low complexity; it is heavily dependent on the type of data and the resulting graph. In this project, some of these issues may be attempted to be fixed.

1.2. Responsibility and Change Authority

Dr. Duffield has the responsibility to dictate changes and adjust the project specifications. I, Arash Abdollahzadeh, am responsible for ensuring that the project is meeting the specifications mandated by the project sponsor, Dr. Duffield.

2. Applicable and Reference Documents

2.1. Applicable Documents

The following documents, of the exact issue and revision shown, form a part of this specification to the extent specified herein:

Document Number	Revision/Release Date	Document Title
IEEE - arXiv:1712.00468	V1 - 26 Mar 2018	Graph Signal Processing: Overview, Challenges and Applications

Table - 1

2.2. Reference Documents

The following documents are reference documents utilized in the development of this specification. These documents do not form a part of this specification and are not controlled by their reference herein.

Document Number	Revision/Release Date	Document Title
IEEE - arxiv.org/abs/1711.06954	19 Nov 2017	A Graph Signal Processing Approach for Real-time Traffic Prediction in Transportation Networks

Table - 2

2.3. Order of Precedence

In the event of a conflict between the text of this specification and an applicable document cited herein, the text of this specification takes precedence without any exceptions.
All specifications, standards, exhibits, drawings or other documents that are invoked as “applicable” in this specification are incorporated as cited. All documents that are referred to within an applicable report are considered to be for guidance and information only, except ICDs that have their relevant documents considered to be incorporated as cited.

3. Requirements

Here, the analytical method that is being used to gain insight on the dataset, as previously mentioned, is the ‘system’, and will be referred to as such. The term VBZ, is referring to the open dataset from the Zurich public transportation network.

3.1. System Definition

The GSP analytics model consists of two subsystems, namely, data preprocessing and the application of the GSP method to the processed data. The input for the system is provided on the VBZ website in a form of relational databases, consisting of individual datapoints and keys that map each entry to another table, providing more information about that datapoint. Figure below shows the relations between different parts of the system, including the processes of raw uncleansed dataset into a graph suitable for the GSP method.

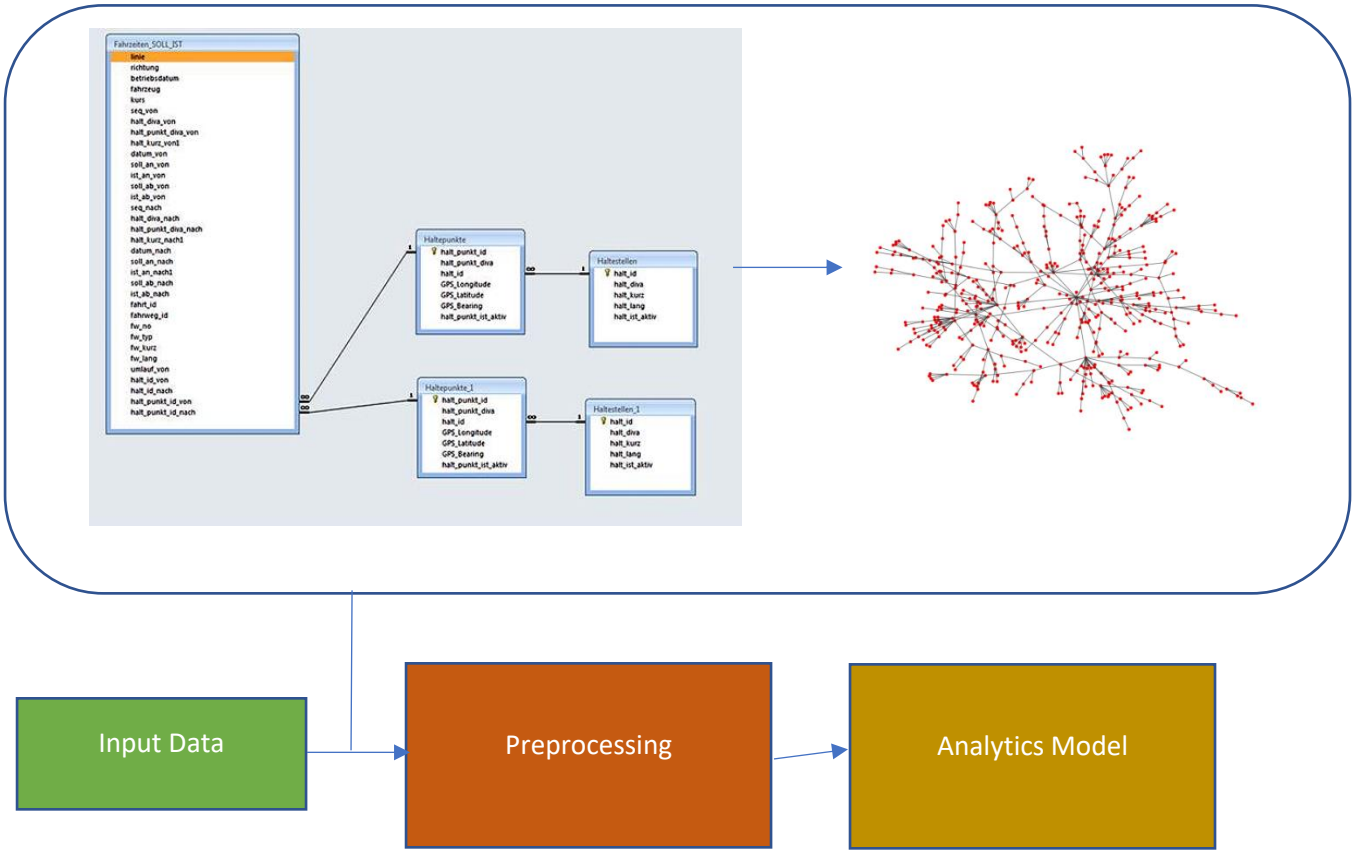


Figure 1. Block Diagram of System

The input data is provided on the website in the form of data tables. There is a total of 52 datasets for each week of years 2016, 2017, and 2018; only 12 datasets are available from 2015, as it is the year that the data collection was initiated. The data is retained on a server which has the capability of accepting SQL commands for data retrieval; and the datasets are also available for download in a comma-separated values format. Each of these files consist about 1.5 million datapoints, and each datapoint is 26 dimensional; each datapoint is mapped to two separate tables via a key, which indicate the bus stop information.

Using this information, the graph required for applying the GSP model can be constructed. To achieve this, the data must be examined and analyzed carefully; construction of the network and defining the signals on the graph can be done in multiple ways. However, once the graph construction methodology is set, the network is constructed as shown in the figure above. The preprocessing subsystem will be a set of functions (in python) that use the matching tables to process the datapoints into a graph.

The analytics model is based on the GSP method for analyzing the time varying signals defined on a graph. It includes a series of steps to partition the graph into smaller pieces, which could be disjoint, and performing Graph Fourier Transform (GFT) on the graph and using Autoregressive–Moving-Average model (ARMA) for training the model. Once the model is trained, the analytics engine can be used to predict the subsequent signal.

3.2. Characteristics

3.2.1. Functional / Performance Requirements

3.2.1.1. File Handling

The dataset is published on a weekly schedule to prevent the file sizes to become too large; each file is 250 MB each and consists over 1.5 million datapoints. The data preprocessing script will use python to process the data and construct the graph. Specifically, the “pandas” module will be used for parsing the files and extracting the information from them.

> *Rationale: The python language is an industry standard for data science projects. The performance and ease of use offered by pandas and the underlying module, ‘numpy’, makes python the best choice for handling large files offline.*

3.2.1.2. Graph Construction

Graph construction will be done via the “NetworkX” module. The ‘NetworkX’ module is a widely used tool for constructing large graphs and specifying meta-data easily. The state-of-the-art big data frameworks like ‘Hadoop’ may be used in this project.

> *Rationale: NetworkX is a very popular graph library in python. It includes many of the methods required for data analysis with highly optimized performance.*

3.2.1.3. Handling Missing Datapoints

Datapoints are often formatted in a way that make working with big data problems more difficult because of incompatibilities between the source and client computer. These types of datapoints will be formatted via interpolation using previous datapoints.

> *Rationale: Consistency in time-series data is crucial and if a datapoint is missing a value or is not formatted in a way for the researcher to be able to make use of that entry, that point cannot be removed, and needs to be handled via approximation, regression, or interpolation. Handling missing data is vital to the GSP model.*

4. Support Requirements

4.1.1. Computer Specifications:

It is recommended that the computer used to run the preprocessing and train the model have a powerful CPU and fast storage. CPU power dictates the time required for training the model, and the storage needs to be fast to accommodate the transferring of very large files into memory. Here are some acceptable specifications:

Processor	Intel Core i7-8650U @1.9 GHz
Memory	16 GB
Operating System	64-Bit, x-64
Storage	Samsung nVME m.2 EVO SSD (at least 40 GB free)
Internet Access	

Table - 3

GPU is not required at this time; however, if their project manager wished to include comparison between RNN results to the scope of the project, a powerful GPU will be essential.

Appendix A: Acronyms and Abbreviations

CPU	Central Processing Unit
GB	Giga Byte
nVME	Non-Volatile Memory
SSD	Solid State Drive
GPU	Graphics processing unit
RNN	Recurrent neural network

Appendix B: Definition of Terms

Pandas	Pandas is a software library written for the Python programming language for data manipulation and analysis.
Numpy	NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.
GSP	Research in graph signal processing (GSP) aims to develop tools for processing data defined on irregular graph domains.

Appendix C: Interface Control Documents

Interface Control Documents is attached in a separated document.

Graph Signal Processing in the Data Analytics of Road Congestion

INTERFACE CONTROL DOCUMENT

REVISION – 1

17 February 2019

INTERFACE CONTROL DOCUMENT

FOR

Graph Signal Processing in the Data Analytics of Road Congestion

Team 15 – GSP

Approved By:

_____

Project Leader Date

_____

John Lusher II, P.E. Date

_____

Minjeong Kim Date

Change Record

Rev.	Date	Originator	Approvals	Description
1	2/17/19	Arash Abdolahzade		Draft Release

Table of Contents

List of Figures

1. Overview

This document, the Interface Control Document (ICD), will outline the connectivity between the subsystems of the Graph Signal Processing (GSP) approach that was previously mentioned in the Concept of Operations document, and details of which were discussed in the Functional Systems Requests. Since this project is a research project, and it does not involve any hardware; as a result, interfaces described are going to be mostly software based and describing how theoretical concepts are adopted to make GSP work.

2. References and Definitions

2.1. References

[1] Shuman, David I, et al. “The Emerging Field of Signal Processing on Graphs.” Extending High-Dimensional Data Analysis to Networks and Other Irregular Domains, 1211.0053v2, 10 Mar. 2013

[2] Hasanzadeh, Arman, et al. “A GRAPH SIGNAL PROCESSING APPROACH FOR REAL-TIME TRAFFIC PREDICTION IN TRANSPORTATION NETWORKS.” Electrical Engineering and Systems Science > Signal Processing, Cornell University, 19 Nov. 2017, arxiv.org/abs/1711.06954.

2.2. Definitions

GSP	Graph Signal Processing
RNN	Recurrent neural network

3. Communications / Device Interface Protocols

3.1 Website to local storage

The files are downloaded from the VBZ website interface onto the local storage, and the names of the files are served as an input to the preprocessing and graph construction scripts. A high-speed internet is needed, as the size of files, when compounded, becomes very large.

4. Data Flow Interface

4.1 File System

Although the VBZ open dataset website is hosted by a server that accepts SQL queries, since the training can be done serverless (in the memory), the dataset will be cleaned and processed locally. So, the data files are going to be downloaded and after the graph construction, only the meta-data is needed.

4.2 Additional Information:

Every dataset is comprised of many datapoints, where to reduce redundancy the bus stop data and breakpoints in the bus network are stored in another file. Construction of the graph is going to benefit from these key-value relational dependencies. Figures below demonstrate an example of how two files are connected.

	soll_ab_nach	ist_ab_nach	fahrt_id	fahrweg_id	fw_no	fw_typ	fw_kurz	fw_lang	umlauf_von	halt_id_von	halt_id_nach	halt_punkt_id_von	halt_punkt_id_nach
0	17622	17693	316152	28607	15	2	15	DEP4 - KALK	122582	2251	1906	11165	10563
1	17958	17927	316062	29829	11	1	11	BEZI - BTIE für Ausfahrt	122863	1306	1502	10551	10622
2	18288	18265	316062	29829	11	1	11	BEZI - BTIE für Ausfahrt	122863	2228	2590	11156	10557
3	18192	18185	316062	29829	11	1	11	BEZI - BTIE für Ausfahrt	122863	1528	2228	11221	11156
4	18138	18116	316062	29829	11	1	11	BEZI - BTIE für Ausfahrt	122863	2657	1528	10574	11221
5	18048	18024	316062	29829	11	1	11	BEZI - BTIE für Ausfahrt	122863	1502	2657	10622	10574
6	18504	18537	316154	31058	1	1	1	FARB - BTIE	122582	2818	2535	10539	10537
7	18444	18488	316154	31058	1	1	1	FARB - BTIE	122582	1310	2818	10578	10539
8	18522	18553	315971	28607	15	2	15	DEP4 - KALK	123432	2251	1906	11165	10563
9	18546	18503	316062	29829	11	1	11	BEZI - BTIE für Ausfahrt	122863	2104	2612	10538	10577
10	18486	18441	316062	29829	11	1	11	BEZI - BTIE für Ausfahrt	122863	1565	2104	10536	10538
11	18396	18356	316062	29829	11	1	11	BEZI - BTIE für Ausfahrt	122863	2590	1565	10557	10536
12	18654	18648	316119	29829	11	1	11	BEZI - BTIE für Ausfahrt	123046	1507	2657	10622	10574
13	18564	18540	316119	29829	11	1	11	BEZI - BTIE für Ausfahrt	123046	1306	1502	10551	10622
14	19008	19012	316154	31058	1	1	1	FARB - BTIE	122582	2250	2248	10570	10559
15	18948	18957	316154	31058	1	1	1	FARB - BTIE	122582	1535	2250	10573	10570
16	18882	18894	316154	31058	1	1	1	FARB - BTIE	122582	2245	1535	10550	10573
17	18786	18802	316154	31058	1	1	1	FARB - BTIE	122582	2253	2245	5122	10550
18	18726	18745	316154	31058	1	1	1	FARB - BTIE	122582	2244	2253	10576	5122
19	18648	18691	316154	31058	1	1	1	FARB - BTIE	122582	2770	2244	10571	10576
20	18564	18595	316154	31058	1	1	1	FARB - BTIE	122582	2535	2770	10537	10571
21	18870	18903	316062	29829	11	1	11	BEZI - BTIE für Ausfahrt	122863	2651	2788	10566	10730
22	18798	18782	316062	29829	11	1	11	BEZI - BTIE für Ausfahrt	122863	2680	2651	10545	10566
23	18740	18720	316062	29829	11	1	11	BEZI - BTIE für Ausfahrt	122863	2615	2680	5064	10545

	halt_id	halt_diva	halt_kurz	halt_lang	halt_ist_aktiv
0	143	2570	BirmSte	Birmensdorf ZH, Sternen/WSL	True
1	309	3356	WalBiSt	Waldegg, Birmensdorferstrasse	True
2	373	6232	FRAF07	Zürich Flughafen, Fracht	True
3	588	3027	FLUG07	Zürich Flughafen, Bahnhof	True
4	623	2989	TZEN01	Thalwil, Zentrum	True
5	701	1012	GOLP	Zürich, Goldbrunnenplatz	True
6	729	687	UitiDrf	Uitikon, Dorf	True
7	751	2758	UNTR07	Glattbrugg, Unterriet	True
8	809	501	BirmBhf	Birmensdorf ZH, Bahnhof	True
9	832	1401	KILB	Kilchberg ZH, Kirche	True
10	833	3254	HALL	Kilchberg ZH, Hallenbad	True
11	891	1991	WalPost	Waldegg, Post	True
12	914	2412	FELD01	Thalwil, Feldstrasse	True
13	1005	2333	THSH01	Thalwil, Schützenhaus	True
14	1010	2160	SCHA	Zürich, Schaufelbergerstrasse	True
15	1067	992	GLAG07	Glattbrugg, Zentrum	True
16	1114	1627	MEHO	Dübendorf, Meiershofstrasse	True
17	1259	1391	BKIL	Kilchberg ZH, Bahnhof	True
18	1270	2680	TRIE	Zürich, Triemli	True
19	1300	1684	MUBA01	Thalwil, Mühlebachplatz	True
20	1305	1472	KUNS	Zürich, Kunsthaus	True
21	1306	478	BEZI	Zürich, Bezirksgebäude	True
22	1308	736	ETHZ	Zürich, ETH/Universitätsspital	True
23	1309	564	BUCH	Zürich, Bucheggplatz	True

df_2016_stop Format: %

Figure shows the Stop points in the transportation system

	halt_punkt_id	halt_punkt_diva	halt_id	GPS_Latitude	GPS_Longitude	GPS_Bearing	hal
0	303	51	143	47,360017	8,456337	85.00000	
1	304	50	143	47,360153	8,456180	270.00000	
2	686	50	309	47,368125	8,463072	212.00000	
3	687	51	309	47,368433	8,463819	19.00000	
4	823	51	373	47,452401	8,571871	208.00000	
5	824	50	373	47,452586	8,572158	29.00000	
6	825	1	373	47,452018	8,571423	92.00000	
7	826	0	373	47,452160	8,570723	275.00000	
8	827	3	373	47,451956	8,571329	96.00000	
9	1290	56	588	47,450402	8,563724	297.00000	
10	1291	53	588	47,450591	8,563741	262.00000	
11	1292	63	588	47,449842	8,563787	279.00000	
12	1293	55	588	47,450465	8,563712	260.00000	
13	1294	57	588	47,450367	8,563684	298.00000	
14	1295	58	588	47,450276	8,563721	285.00000	
15	1296	66	588	47,449718	8,563829	209.00000	
16	1297	65	588	47,449772	8,563804	220.00000	
17	1298	61	588	47,449998	8,563716	240.00000	
18	1299	1	588	47,450074	8,564155	350.00000	
19	1300	67	588	47,449645	8,563881	270.00000	
20	1301	50	588	47,450423	8,564057	279.00000	
21	1302	0	588	47,450273	8,564053	188.00000	
22	1303	68	588	47,449591	8,563893	270.00000	

df_2016_breakpoint Format: %

Figure shows the Breakpoints in the transportation network

Figure 1 – Snapshot from python “dataframe” of the master and two children datasets

Graph Signal Processing in the Data Analytics of Road Congestion

SCHEDULE

REVISION – 2

2 May 2019

	01/28/19 to 02/04/19	02/04/19 to 02/18/19	02/18/19 to 03/04/19	03/04/19 to 03/18/19	03/18/19 to 04/01/19	04/01/19 to 04/15/19	04/15/19 to 04/29/19	04/29/19 to 05/02/19
Define the project								
Write ConOps								
Examine and gain intuition about the dataset								
Write FSR								
Write ICD								
Learn about GSP								
Learn about previous work on GSP applications in traffic data analysis								
Develop graph construction plan								

Legend

- Behind Schedule
- Not Started, on Schedule
- In progress
- Completed

Graph Signal Processing in the Data Analytics of Road Congestion

Validation

REVISION – 2

2 May 2019

Validation plan for Graph Signal Processing in Data Analytics

Status Indicators	
Completed	
On Schedule/In Progress	
Behind Schedule	

Task	Deadline	Current State	Status
Data Preprocessing	04/29/2019	Completed	
Ensure missing datapoints are handled	03/10/2019	Completed	
Confirm file reading methodology	03/10/2019	Completed	
Ensure correct usage of the entire dataset components	04/14/2019	Completed	
Graph partitioning time	04/14/2019	Completed	
Graph constructed can be used to validate the GSP model	04/29/2019	Completed	

Graph Signal Processing in the Data Analytics of Road Congestion

SUBSYSTEM REPORTS

REVISION – 1

1 May 2019

SUBSYSTEM REPORTS

FOR

Graph Signal Processing in the Data Analytics of Road Congestion

Team 15 – GSP

Approved By:

_____

Project Leader Date

_____

John Lusher II, P.E. Date

_____

Minjeong Kim Date

Change Record

Rev.	Date	Originator	Approvals	Description
1	5/1/19	Arash Abdolahzade		Final Release

Table of Contents

List of Figures

List of Tables

1. Introduction

The graph signal processing (GSP) is a general term for transformation of complex data structures into simpler independent components for further analysis. The novel approach *to traffic congestion analysis, developed and published by Arman Hasanzadeh et al. in the paper "A Graph Signal Processing Approach For Real-Time Traffic Prediction In Transportation Networks*" seeks to utilize the spatio-temporal dependencies between the datapoints to enhance the prediction accuracy, and use GSP to reduce the computational and space complexity of the process. For further study in GSP application, this data mining and cleaning subsystem examines the quality issues of the Zurich public transportation dataset, creates a graph and defines the traffic signal (normalized travel time) on each node (road segments). The design choices are supported figures and tables, and the process of data cleaning and mining is better explained by limiting the scope to a specific example.

2. Data preprocessing and Mining Subsystem Report

2.1 Subsystem Introduction

In traffic congestion analysis, the GSP based analytic algorithm requires the data to be in a form of a directed or undirected graph G (V, E, W), where the nodes (V) represent road segments, edges (E) represent connectivity (not necessarily physical, can be modeled), and weights (W) represent a weight function determining the strength of the connection between the nodes. This subsystem is designed to mine information from the publicly available VBZ bus transportation system target-actual dataset (Zurich's public transportation dataset) to build a data structure that comply with the requirements of the GSP algorithm. Ultimately, a graph of road segments with a signal defined on each node is mined from the massive VBZ dataset. The process of developing the final graph is explained in subsections below through an example (a subset of the entire dataset). Although for this report, only one month of the data (out of the year) is used, the code has been tested with all the other months and the results are very similar. The graph and the signals mined from this dataset will be used to evaluate the GSP algorithm against other algorithms later and serve as a platform for testing and improving the algorithm (ECEN 404).

2.2 Subsystem Details

The VBZ dataset is comprised of the target and actual times (rounded to the nearest seconds) for the buses in the public transportation system of Zurich. The dataset is distributed into datasheet files, where each file contains about 1.5 million recordings. These recordings can vary for each road segment, as some road segments are traversed more frequently than others. The following sections are dedicated to the process of understanding this dataset, developing the data mining strategy, and explain the results to support the design decisions made. The data recordings are available from September of 2015 and continuing (published every week). However, this report is generated by using the first month of 2016.

2.2.1 - Exploratory data analysis – Delays

First step of understanding the dataset was to explore all the attributes and trying to find relations between them; pandas and numpy, especially the pandas dataframe and Series were heavily used throughout this project.

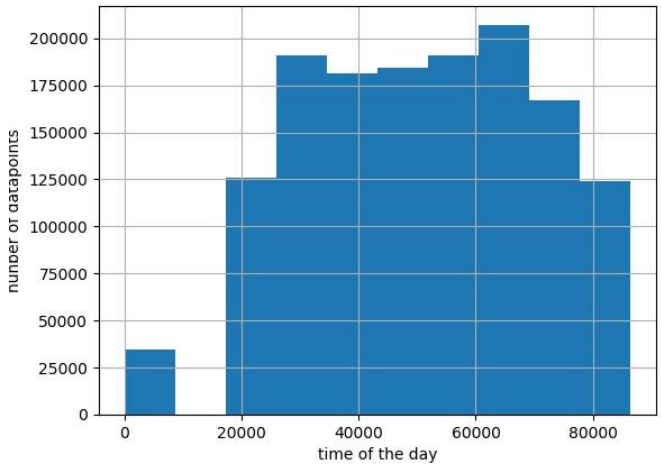
2.2.1.1 – Exploring the data

The attribute description page on the VBZ website is in German, and some of the terms are not translated very well (by Google translate); also, many of the crucial attributes are internally used proprietary IDs, where there is not enough information about how these attributes are connected to others. So, further study and many experimental analyses were necessary to uncover how they can be manipulated for organizing the data mining step.

Table 1: Some important attributes from the dataset

Attribute Name	Definition
seq_von	sequence of stops in a given route - from
seq_nach	sequence of stops in a given route - to
betriebsdatum	date of data recording
fahrt_id	unique ride id – may be overridden every week by another ride (is not unique to that ride in a successive week)
fahrweg_id	unique route id - useful for keeping track of a stop sequence
fw_typ	type of transportation
halt_id_von and nach, halt_punkt_id_von and nach	keys to Haltestelle and Haltepunkt

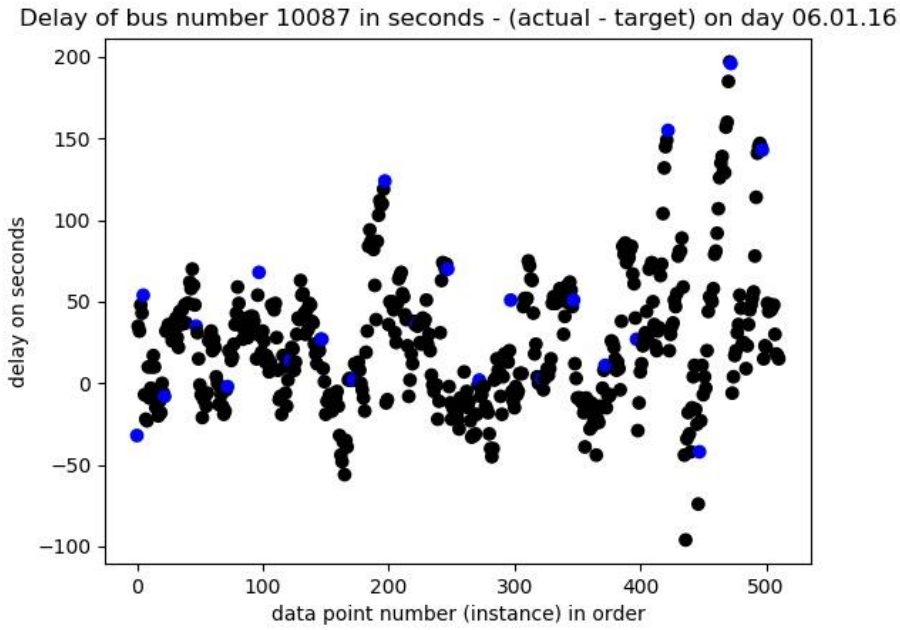
Figure 1: Number of datapoints distribution per day against the time of the day (in seconds after midnight)



Also, since each row consisted of a 'from' and a 'to' point, to validate that these two numbers always matched, all of the values of 'to' section were shifted up by a values, and then subtracted from the 'from' values; the results were checked to see if there are any anomalies, since the rest of the analysis is heavily dependent on using these two points that belong to one row at a time.

The plot below shows the distribution of the delays per day, where the black points indicate a route change.

Figure 2: Delays in a given day



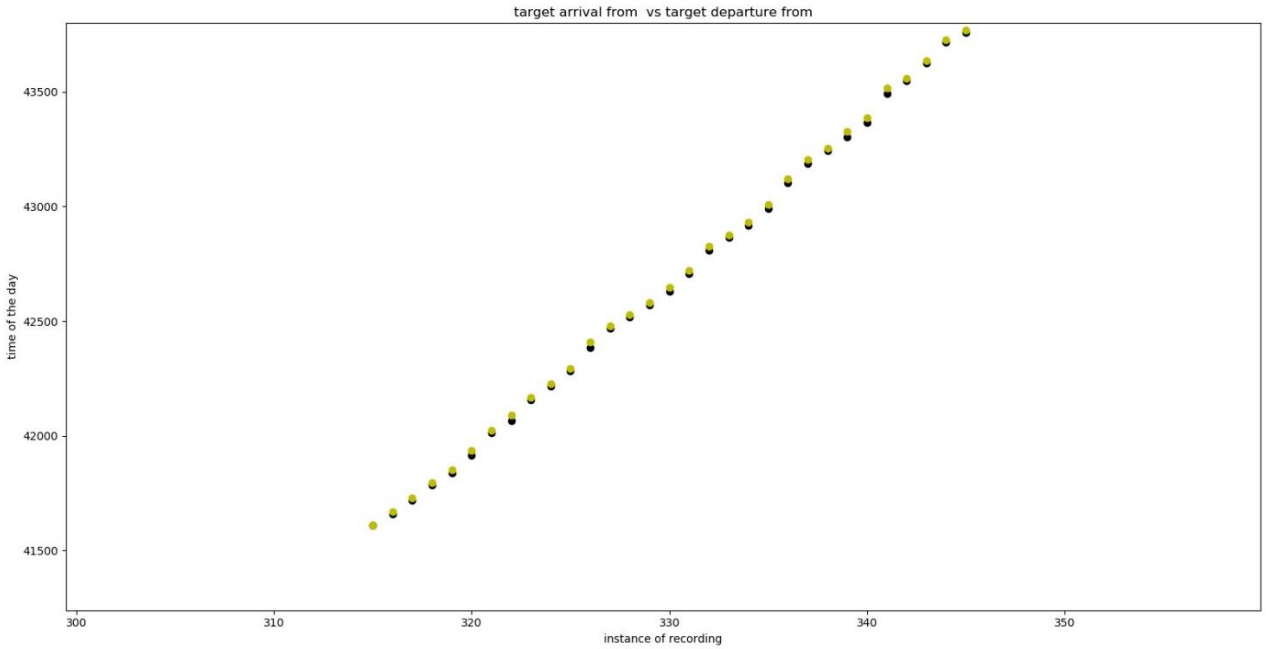
2.2.2 - Exploratory data analysis - Travel Times and Built in Considerations

The time travel index (TTI), which is a ratio of the measured travel time vs. free-flow travel time, is a key indicator of traffic congestion; however, in this dataset, the free flow travel times for a road segment was unavailable. So, it needed to be determined whether there is already a built-in consideration for a given route. Understanding this quality of the dataset is crucial to forming the traffic signal indicating a congestion, since the actual travel time needs to be normalized because of the differences in instances of buses from other routes traversing the same road segment. Since the TTI signal could not be formed from the dataset itself, the traffic signal is going to be modeled as the actual time travel divided by the anticipated time travel. The next two sections support the decision for this modeling choice.

2.2.2.1 – First Attempts

At first, I attempted to determine the expected delays by plotting the target departure times ‘from’, and target arrival times ‘from’ side by side, for every time recording in a given day and course number. The resulting plot is as shown.

Figure 3: Target arrivals aginst target departures



Throughout the day, it was apparent that there is a repeating pattern, however, it was difficult to detect how varying these built-in allowances were. So, travel time and stationary time signals were visualized for many instances of different combination of routes, bus numbers, and days. The following is an example, showing travel time and stationary time on the 5th of January 2016.

Figure 4: Travel times in seconds for a given day – showing the distribution in a given day

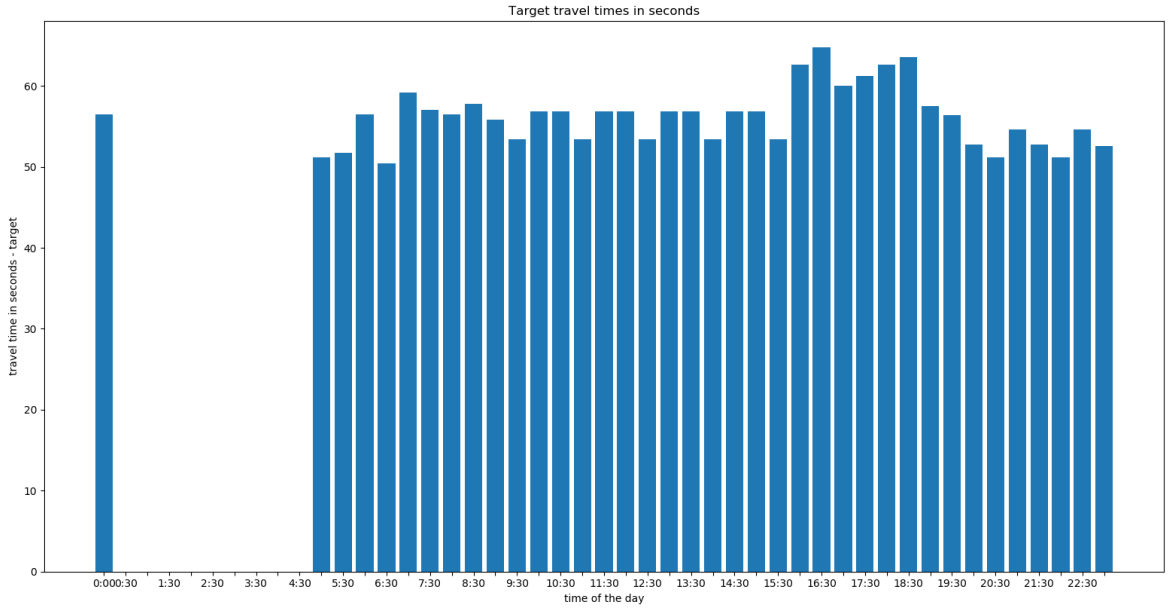
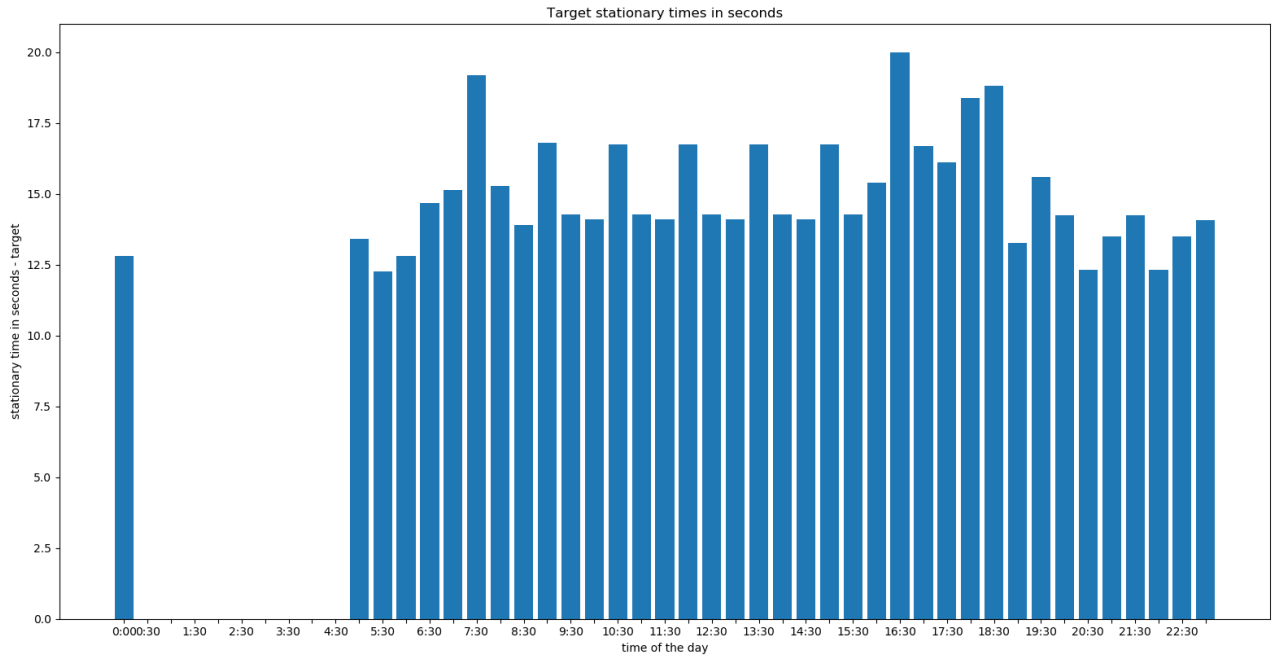


Figure 5: Stationary times in seconds for a given day – showing the distribution in a given day



Target stationary time is defined as the time anticipated for a bus to stay at a given stop (target departure from - target arrival from). And the target travel time is defined as the time determined by VBZ for bus to spend traversing a given route.

2.2.2.2 – Using Heatmaps

The results above show some variation in travel time as a function of the time of the day, however, a heatmap of time intervals vs. the road node was made to understand how these variations were spread, and their connection with the actual travel time. The intensity of the colors indicates higher travel time. The example below, shows the top three traversed routes in order (sequence of unique stops “haltpunkt”s). The dataset is limited to January 5th, 2016, and the results are aggregated by taking the input majority.

Figure 6: Heatmap of the target (left set) and actual (right set) travel time a given day per day for sequence (route) number 18305 – most commonly traversed route

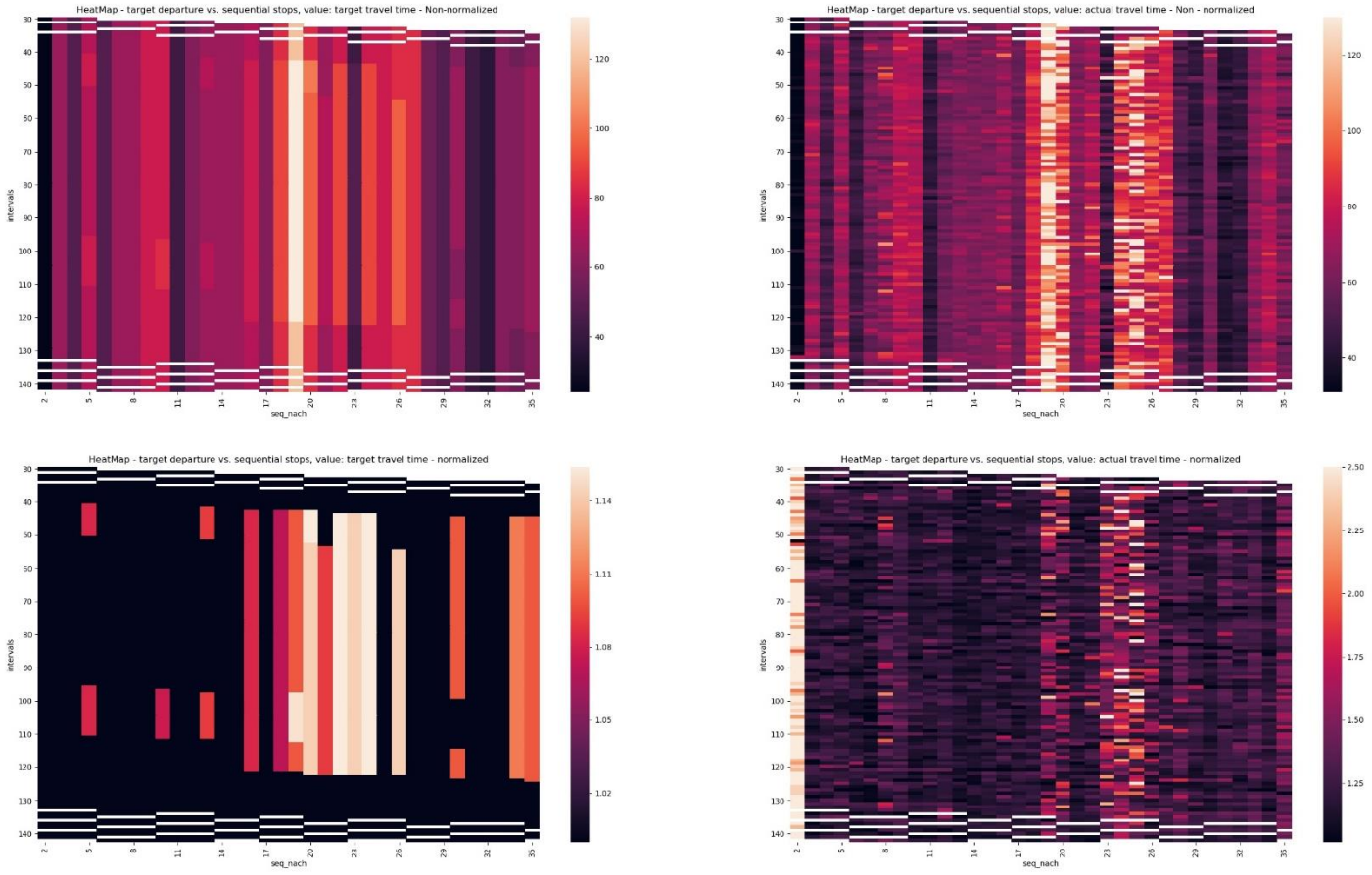


Figure 7: Heatmap of the target (left set) and actual (right set) travel time a given day per day for sequence (route) number 18306 – second most commonly traversed route

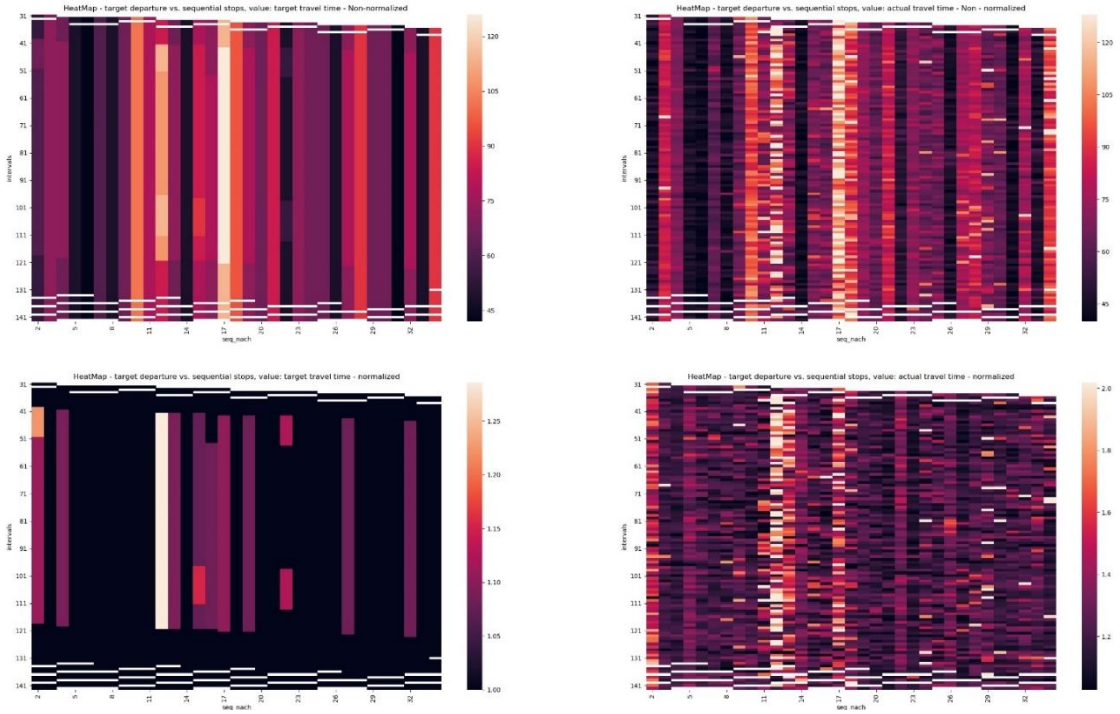
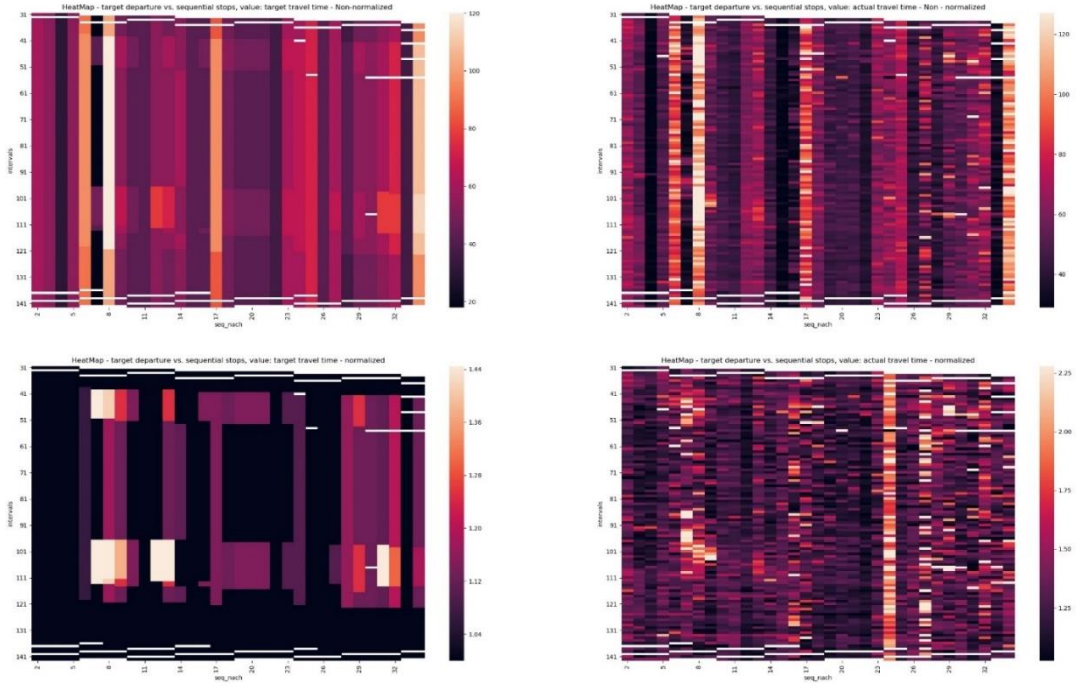


Figure 8: Heatmap of the target (left set) and actual (right set) travel time a given day per day for sequence (route) number 21424 – third most commonly traversed route



The plots on the left represent the target time travels (aggregated by majority), and on the right, actual travel time. The time intervals are set at 10 minutes. The top plot is not normalized; however, the bottom plots are normalized by the minimum of the travel time for that route (X-axis). As can be seen, depending on the route, there are built in consideration for the travel times, where they vary according to the time of the day. For instance, in the plots belonging to Route 21424, brighter spots can be seen in some routes in two different segments, where those two segments belong to interval range (40 to 55), and (95 to 115), which are 6:45 AM to 9 AM and 4 PM to 7:30 PM. So, the target travel time, which is defined as the result of subtracting the “target departure from“ from the “target arrival to”, is going to serve as a normalizing factor much like the free-flow traffic in TTI.

2.2.3 - Exploratory data analysis – Map Visualization

After finding a suitable normalizing factor, I seeked to visualize the bus stops to understand the distribution of them, as well as validate the connection between the Stops (breakpoints and in German: haltpunkt) and stations (German: haltestelle). The following shows a selection of the maps visualized using GPS data from the haltpunkt and haltestelle data tables. The GPS location of stations (haltestelle) is calculated by aggregating (by averaging latitude and longitude) of all the stops (haltpunkts) belonging to that station. In some instances, the GPS data is missing for the some of the stops, however, these stops never appear in the recordings table.

Figure 9: All the stations of the network

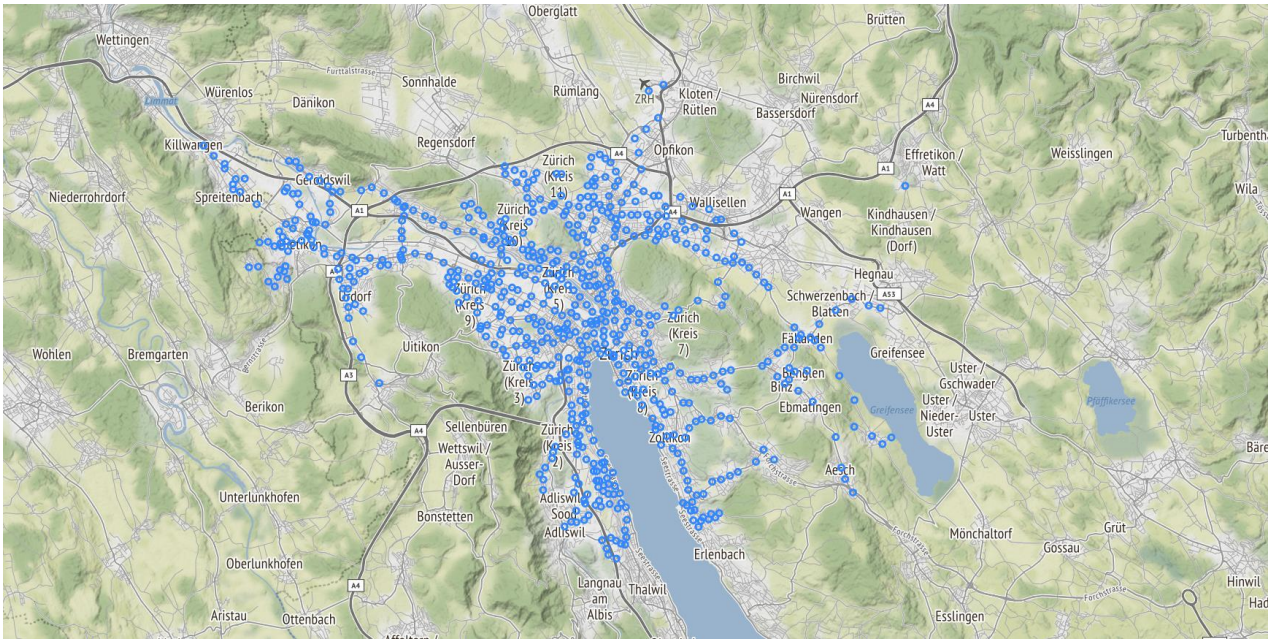


Figure 10: Stops distributions – Similar colors indicate that the stops belonging to a given station

To see how well the network is connected, I picked the busiest station (station id: 1565) and found each of the routes (set of stations) traversing through this station. The map can be seen below; the physical connections between the stations are crucial for the construction of the graph of road segments. The color coding in this map is not the same as the map above; same color stops (haltpunkt) belong to a specific route (set of stations).

Figure 11: Every route passing the most traversed station

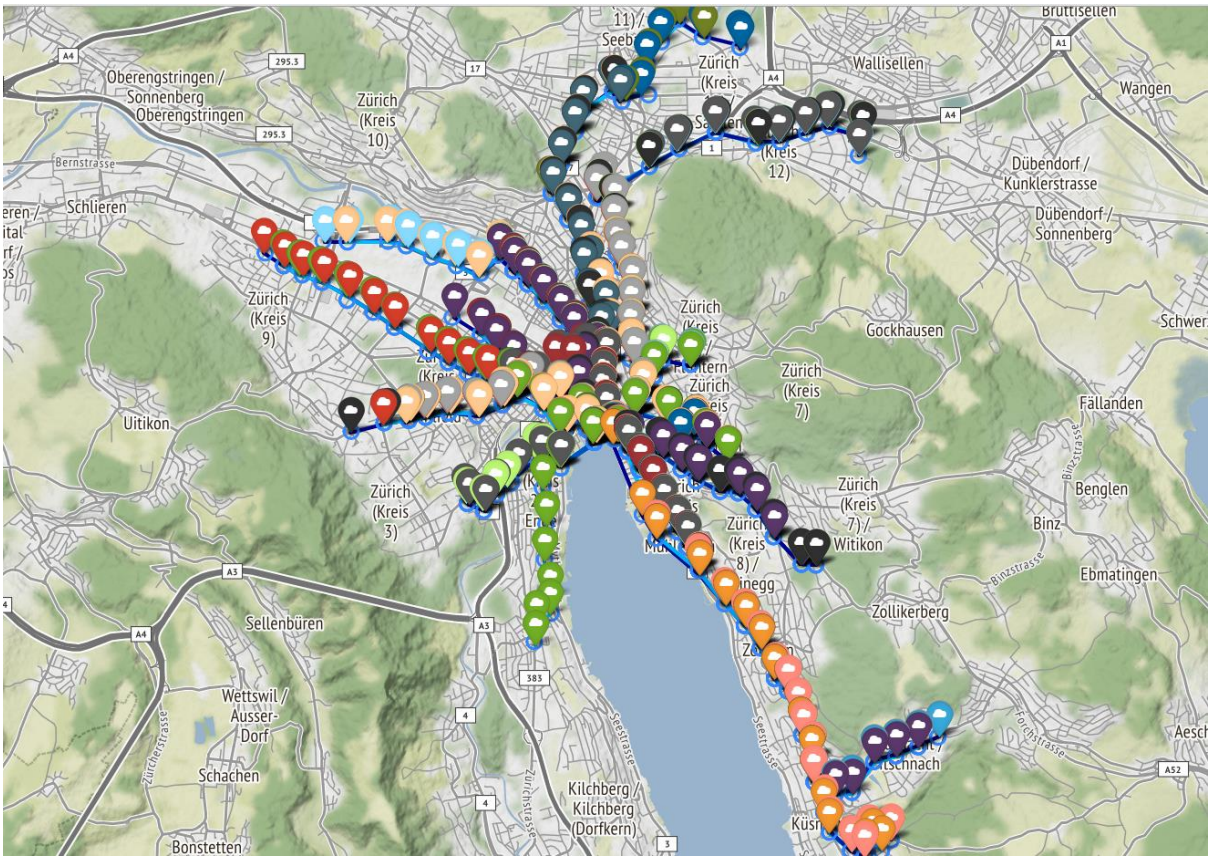
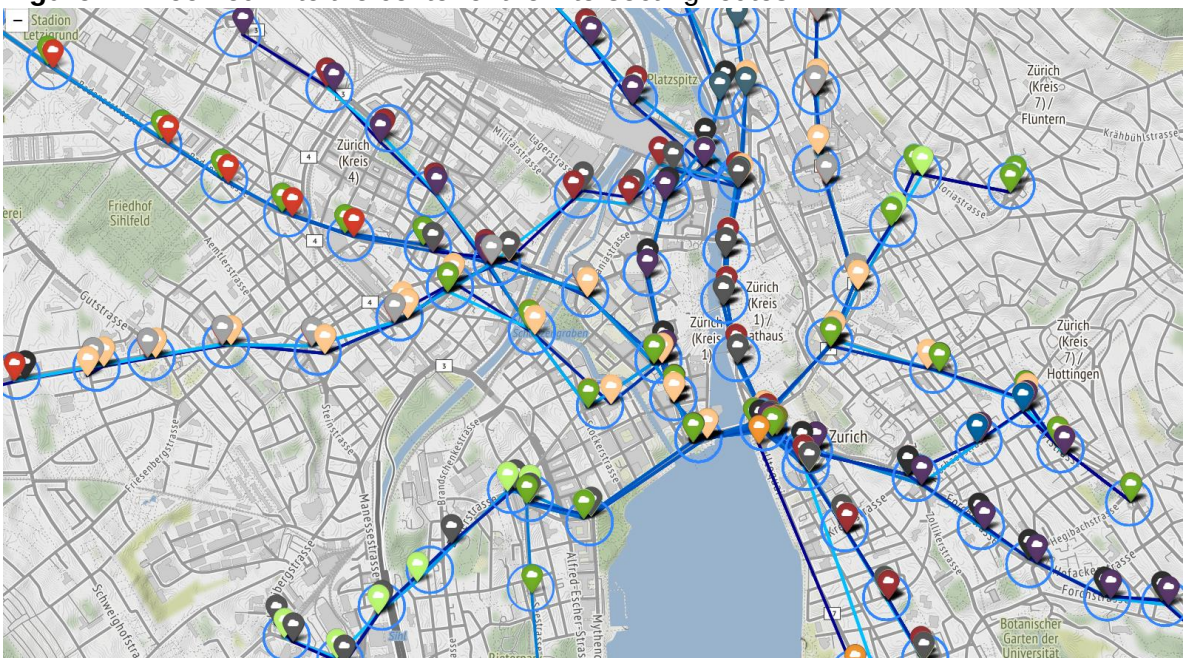


Figure 12: Zoomed in to the center of the intersecting routes



The blue line darkness increases as the bus traverses the path, determines the direction of travel.

2.2.4 - Exploratory data analysis - Data Granularity, Defined Signal, and Graph Construction

One of the major issues with using the VBZ dataset to mine traffic congestion signals, based on normalized travel times, is the fact that signals are defined in asynchronous interval. That is, the variation between how often a bus is scheduled to arrive at some stop is too high. The road segments in the central part of the city can have buses traversing them as high as 40 times per hour, where on the contrary, roads at the outskirts of the city are going to be traversed once every hour; and in some extreme cases, some of the road segments are only traversed during the rush hours (6:30 AM to 9 AM, and 4 PM to 7:30 PM).

2.2.4.1 – Time intervals

Using the example dataset described before, there are 1560 road segments, and 648 stations. So, picking a reasonable time interval length (i.e. how long the working day would be binned into) is crucial to the integrity of the resulting graph and the signal; if the interval lengths are too large (30 minutes), more nodes will be kept due to the availability of data for that road segment, however, if there is a traffic congestions, it will likely clear out within 30 mins. On the other hand, if the interval length is reduced, the number of nodes needed to be eliminated because of lack of available datapoints will render the resulting graph too small. Many different combinations were examined, and a subset of the resulting granularities are described below. During the construction of the table below, the start of the day is assumed to be at 6 AM, and end of the day at 8 PM. The issue of start and end time will be addressed in the next section.

Table 2: Interval length and number of routes to be kept

Length of interval (in seconds)	**Number of nodes to keep**
120	0
180	1
240	43
300	101
450	568
600	722
900	1029
1200	1094

Figure 13: Distribution of non-empty vs empty bins – 30 min intervals

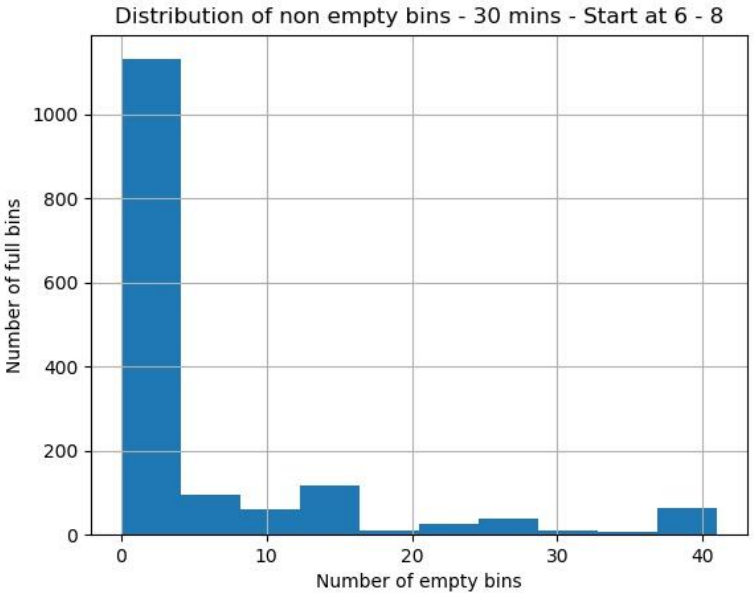


Figure 14: Distribution of non-empty vs empty bins – 10 min intervals

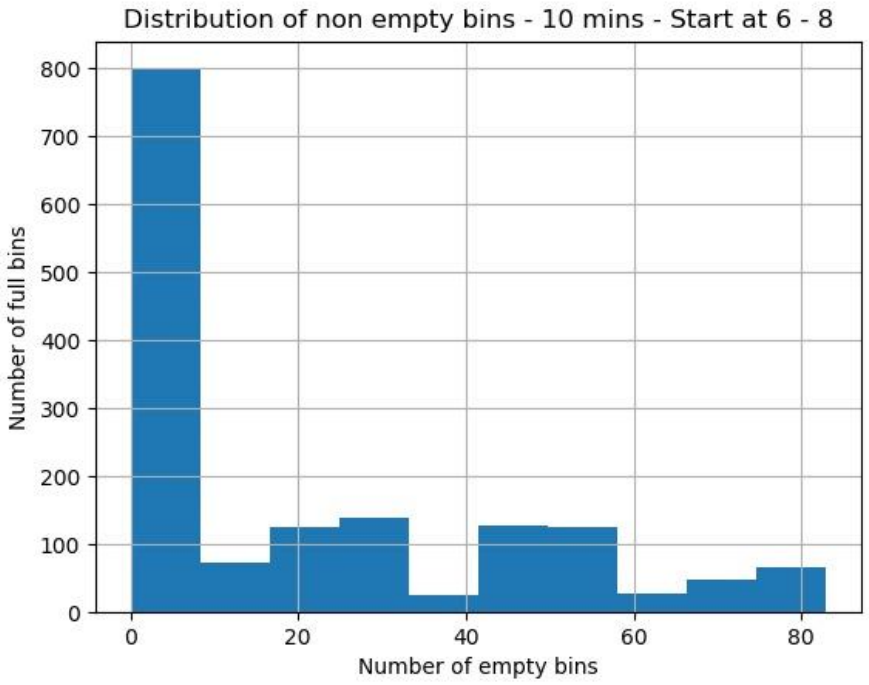
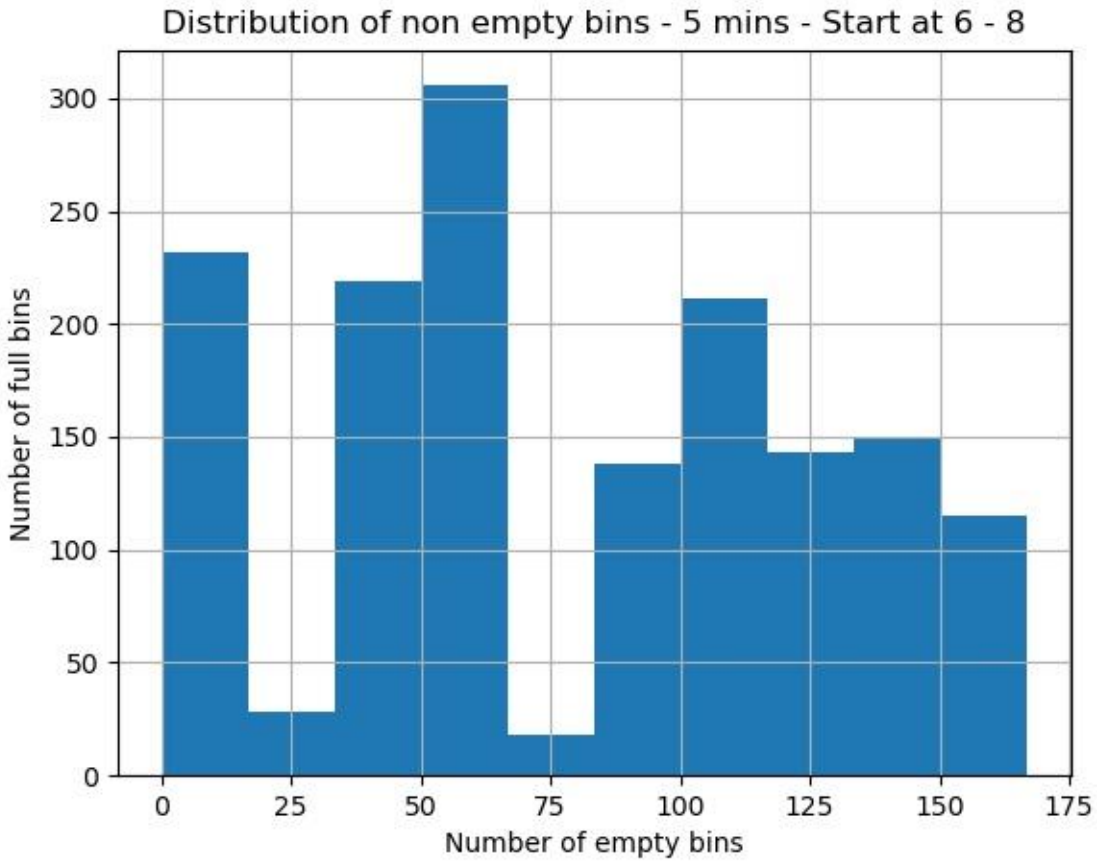


Figure 15: Distribution of non-empty vs empty bins – 5 min intervals



2.2.4.2 – Starting time

Starting times are also important to the road segment vs. signal granularity tradeoff. There is a noticeable imbalance in the distribution of data points per interval in the start and end of the day. Table below shows a small subset of examined combinations.

Table 3: Starting times and nodes to keep

Interval (seconds)	Start and end time	Number of road segments to keep
300	5 AM to 12 AM	0
300	6 AM to 8 PM	101
600	5 AM to 12 AM	0
600	6 AM to 8 PM	722

The following figures show the distribution of the empty bins against non-empty ones.

Figure 16: Distribution of non-empty vs empty bins – 5 min intervals 5 AM to 12 AM

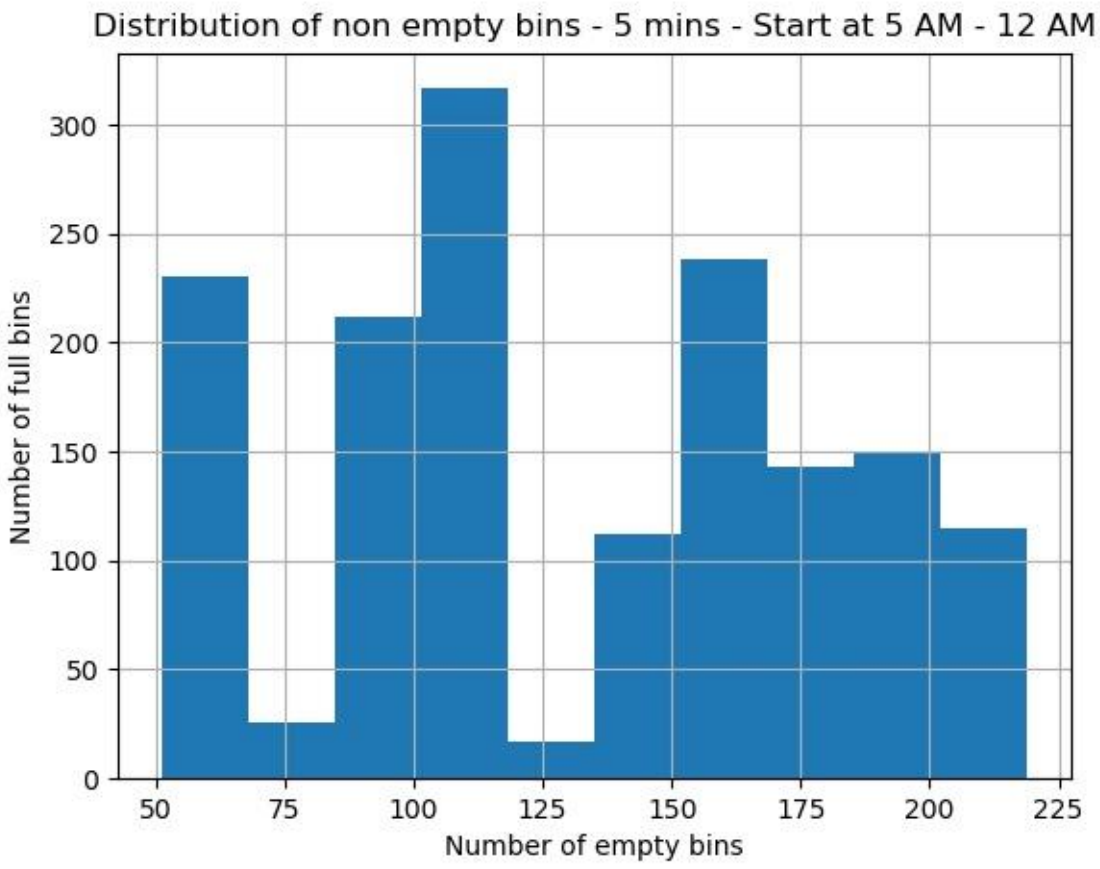
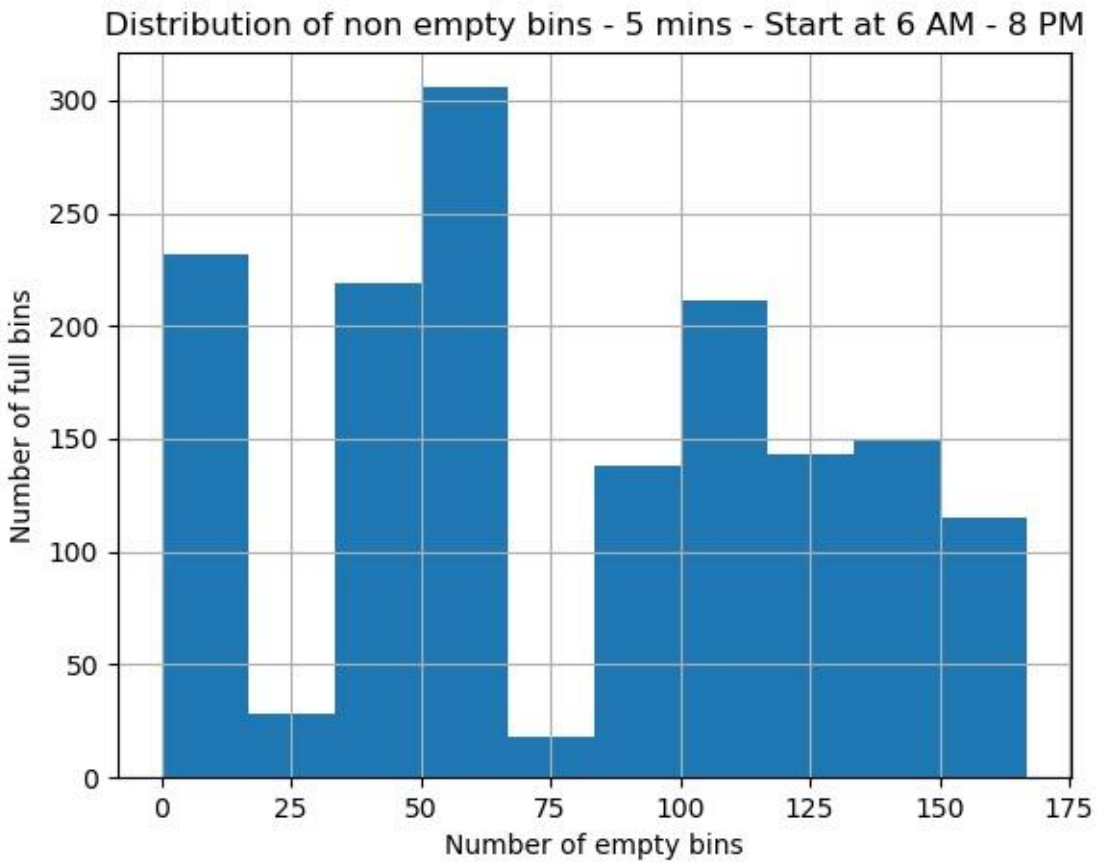
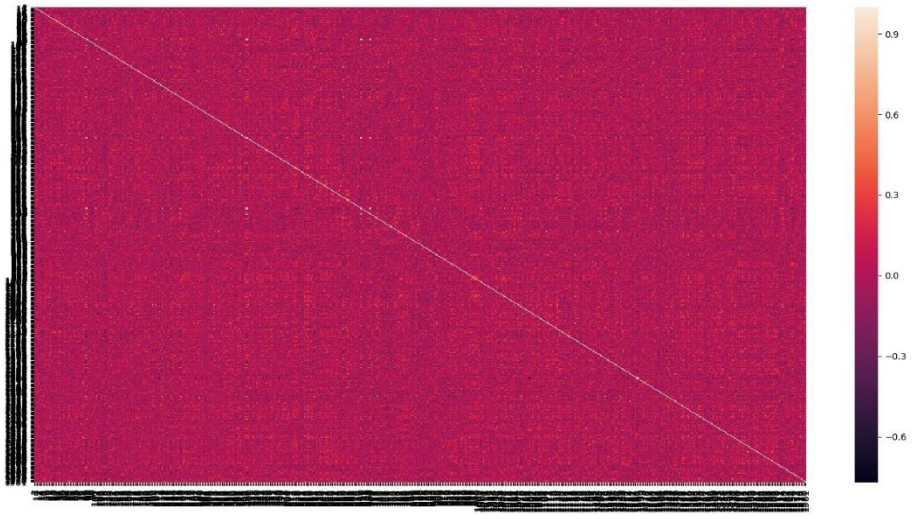


Figure 17: Distribution of non-empty vs empty bins – 5 min intervals 6 AM to 8 PM



Although the official start and end time of the day for the Zurich public transportation is 5 AM to 1:30 AM (next day), through many iterations, the final interval length is set at 10 minutes, and start and end times are set as 6 AM to 8 PM. All the other data points are removed from the dataframe. Nodes with 15 or less missing values (empty bins) are kept, and the final dataframe is formed. The rest of the datapoints are interpolated using the scipy library (linear method). The final dataframe's correlation matrix is as follows.

Figrue 18: Heatmap showing the cross-correlation between all the traffic signals (normalized time travel)



2.2.4.3 – Graph Construction

With the signal and nodes (the road segments) defined, the graphs can now be constructed. Two graphs were formed with the nodes defined as the road segments, where they are only different by how the edges are defined. The first version uses the proximity of the mid-points of the road segments to create an edge between two points. An edge exists between two nodes (v1 and v2) (road segments) if the midpoint of that node (v1) is less than 1000 meters from the other node (v2). The second version uses the physical connections between the nodes; that is, if a node (road segment) is connected via some station to another node, an edge exists between them. The weights are defined as inverse of the physical distance between the two adjacent nodes.

The validity of these two versions of the graph is one of the research aspects of this project. However, it can be argued that graph 1, G1, is superior to the second version, G2, as more connections are made between the nodes with high signal correlation coefficient. Pearson correlation coefficient is used to capture the correlation between the two signals. The following figures show the connections made between the nodes, and their correlation. The results match my intuition, that two roads may be close, which in turn, the traffic signals can be very related, but since they are not physically connected, there would be no edge defined between them in G2.

Figure 19: Cross correlation coefficients (Pearson) for all the nodes (road segments) connected to node number 659, where the connections are defined by G1

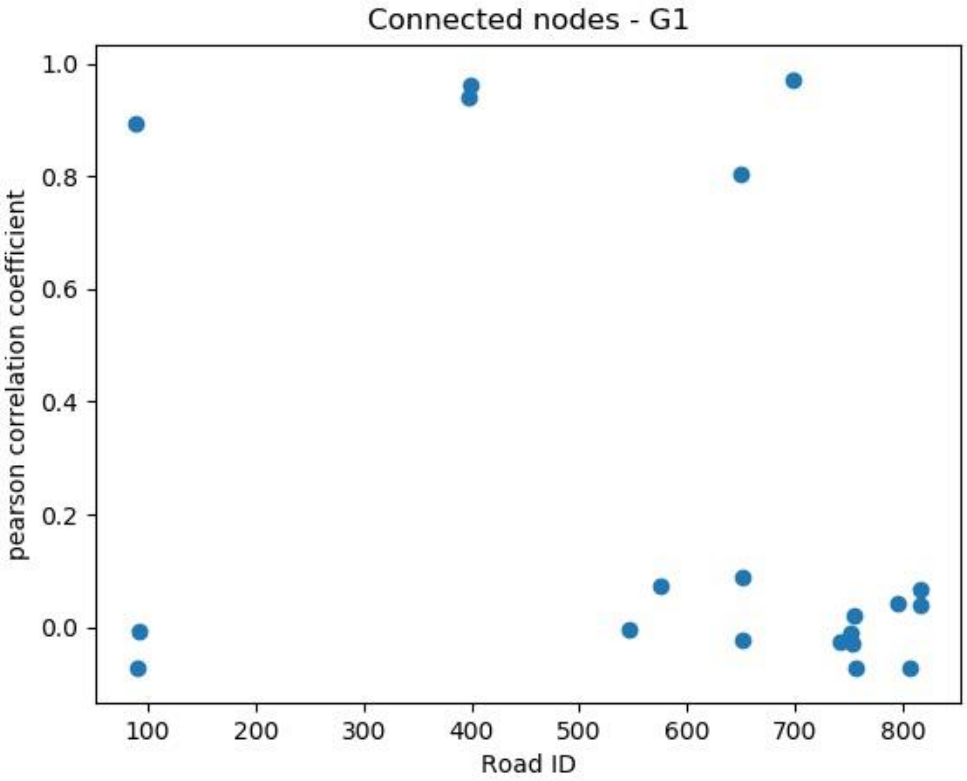


Figure 20: Cross correlation coefficients (Pearson) for all the nodes (road segments) connected to node number 659, where the connections are defined by G2

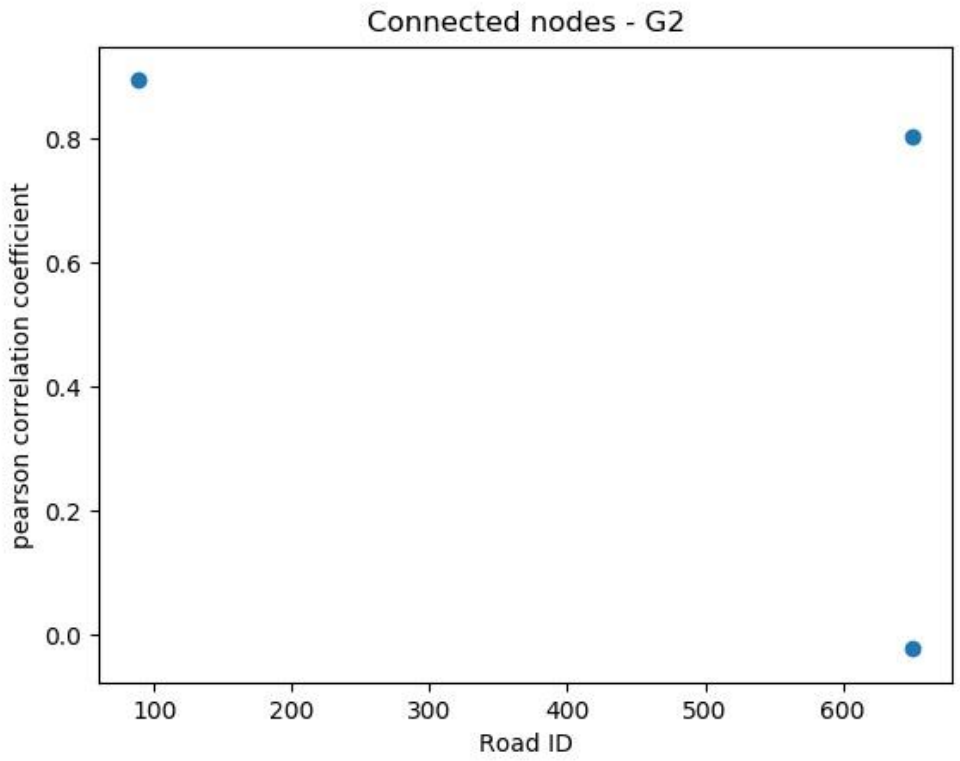


Figure 21: Graph of G1 topology

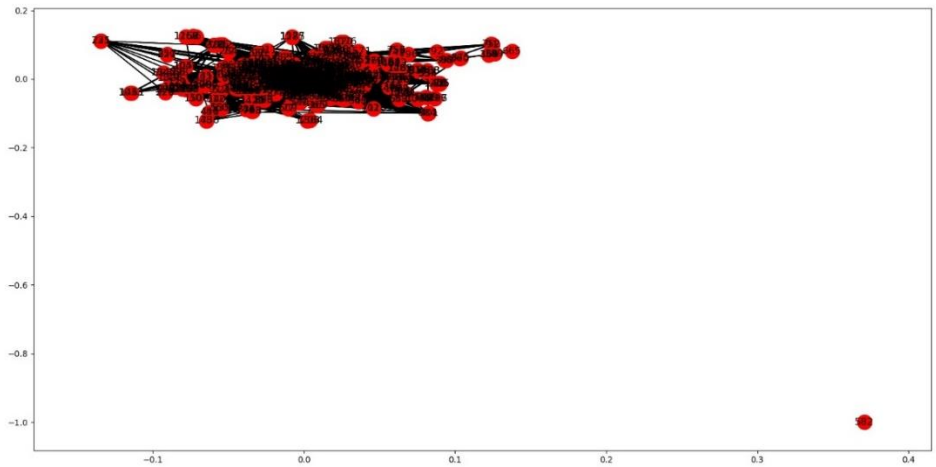


Figure 22: Graph of G1 topology – zoomed in

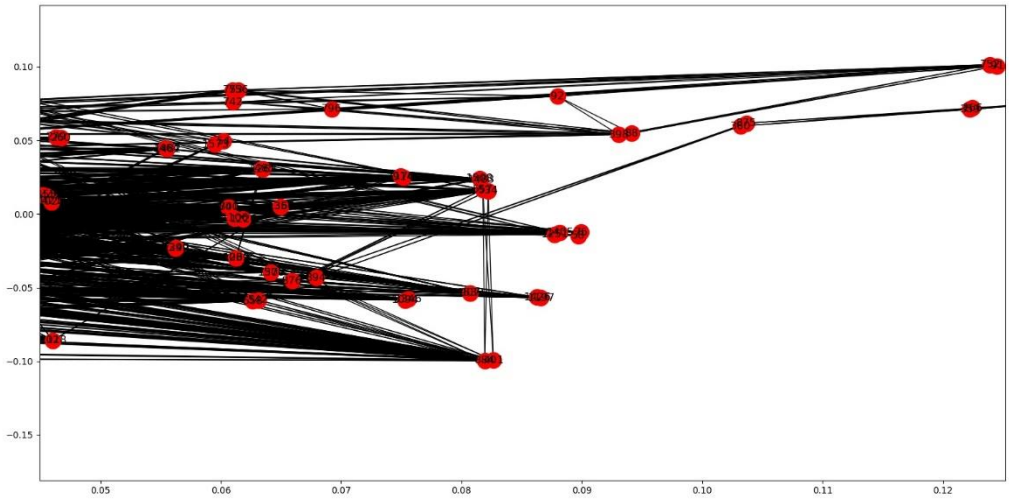


Figure 23: Graph of G2 topology

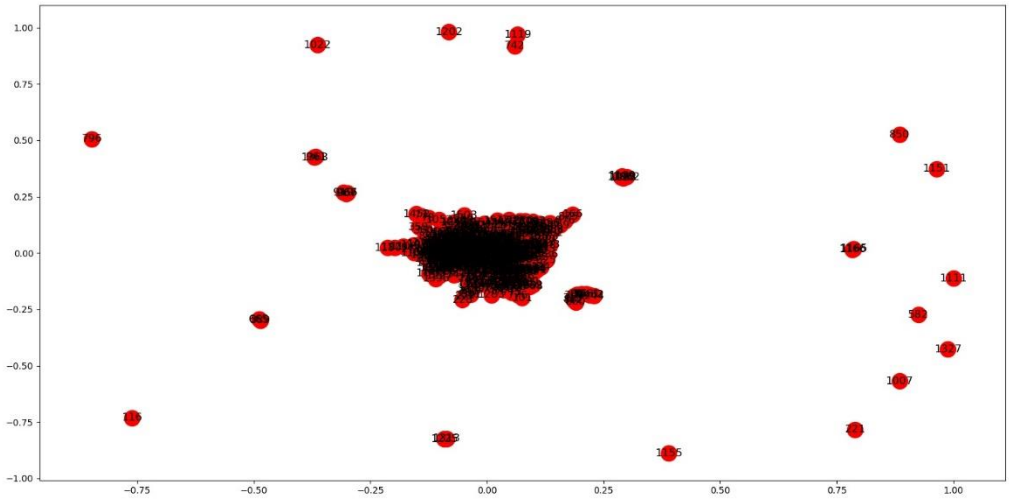
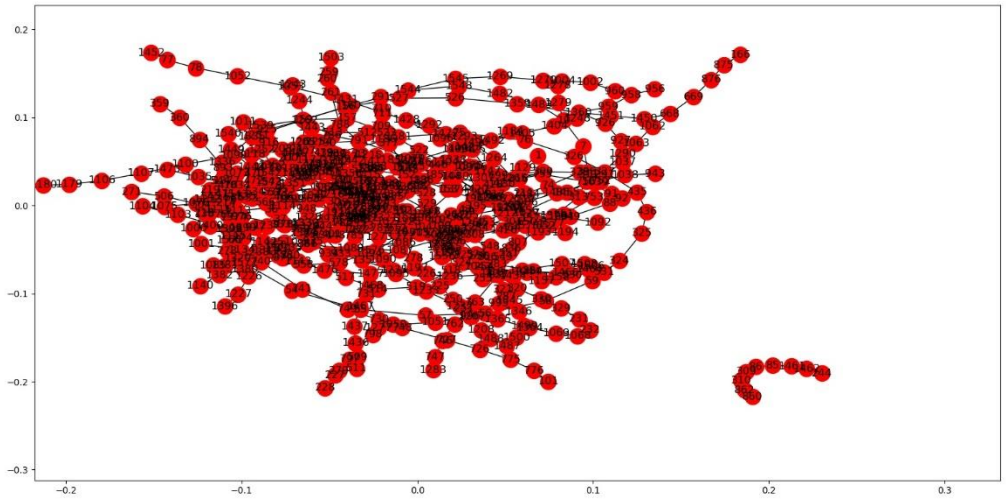


Figure 24: Graph of G2 topology – zoomed in



Due to the proximity, and the granularity issues addressed above, the G1 graph is very strongly connected, where only one node is not connected to the strongly connected subgraph. However, G2 has some sequences that are not connected to the majorly connected subgraph shown above.

Finally, I developed a script to animate the traffic signal via a varying intensity heatmap.

Figure 25: Traffic patterns at 6:40 AM Jan 4th, 2016

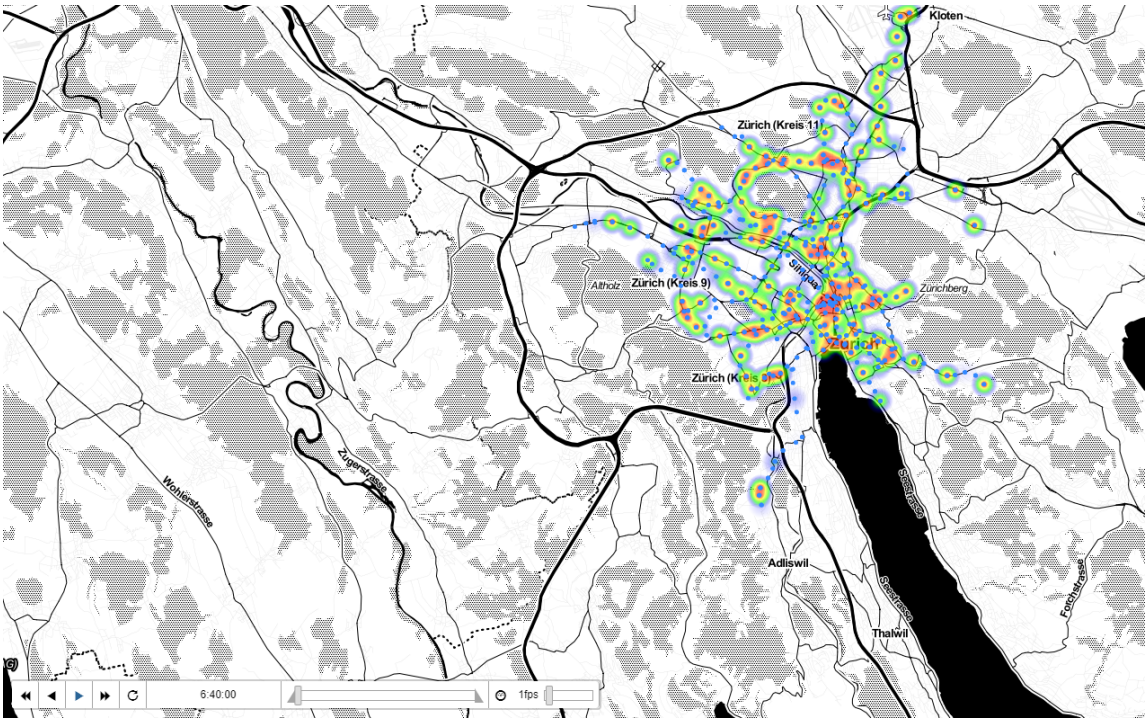


Figure 26: Traffic patterns at 11:10 AM Jan 4th 2016

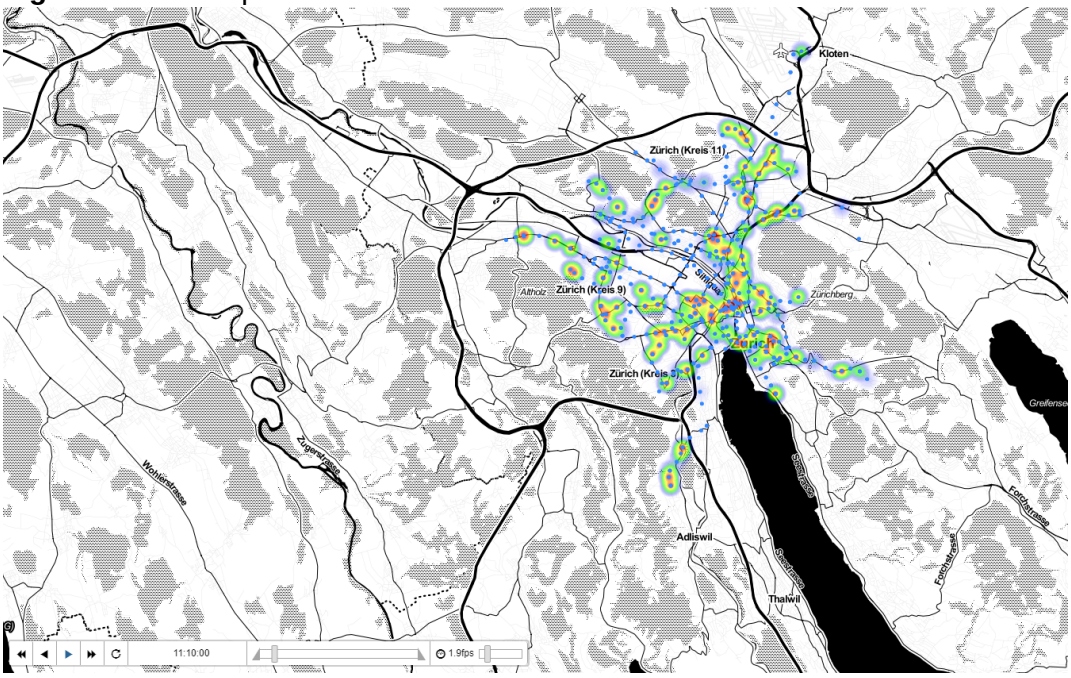
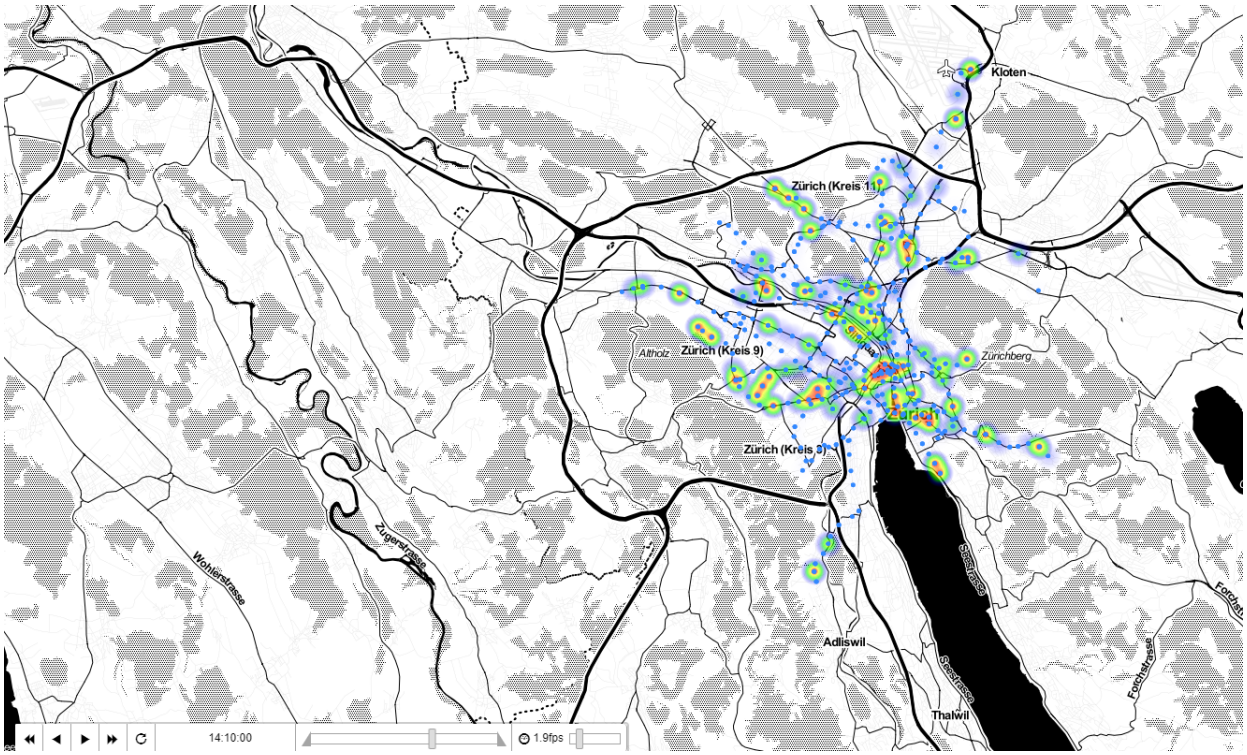


Figure 27: Traffic patterns at 14:10 AM Jan 4th 2016



2.3 Subsystem Conclusion

This subsystem consisted of mining necessary information from the massive dataset of the public transportation system of Zurich. The deep study into this dataset revealed some quality issues that are going to be challenging to resolve. The major issue with this dataset is the balance of number of nodes vs. the granularity of the dataset. As mentioned above, the dataset does not possess the necessary information to model a signal at a reasonable sample space. For example, one of the state-of-the-art research papers in Spatio-Temporal data analysis, “Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting” has tested their algorithms on very well-defined signals (average speed measured by the sensors) at 5 minutes intervals, where they have datapoints throughout the day (24 hours). This dataset is limited to datapoints from 5 AM to 1:30 AM (next day), which can introduce an issue with CNN based models; because the passing ConvNet blocks assume dependence between the datapoints in the block, but, since there is a very large discontinuity between the datapoints (from one day to another), training these models on this dataset would be ineffective.

3. Analytics Subsystem

3.1. Subsystem Introduction

Since this is a research project, this section would mark the end of this report. After the data mining and cleaning phase, the next step is to use the constructed graph for testing and validation of an analytics algorithm of choice. At first, we intended to use the resulting graph to examine the effectiveness of graph signal processing methods in time prediction; the graph signal processing in short is the process of transforming a graph with a signal defined on each node (at each time-step) to a new space where the eigen-vectors of the Laplacian matrix form the basis of, then use a simple linear model for prediction, and finally use inverse-transform to reconstruct the actual graph with the time-series on the nodes. However, our attention was later focused on matrix completion using encoders, which is a more exciting and challenging sub-field of machine learning. Usage of deep learning methods for matrix competition has been historically limited to link prediction and node inference, so, we are focusing on using the complete construct of the underlying graph to help with the imputation. Matrix competition and data imputation is a crucial part of many machine learning problems. A graph could be constructed for many of the complex real-world problems today, and using our method, variational graph auto-encoder, can take on the imputation task by naturally incorporating the relations between the nodes. Some of the previous usage of imputing graph links and data on the nodes are in recommender systems and sensor data imputation.

3.2. Subsystem Details

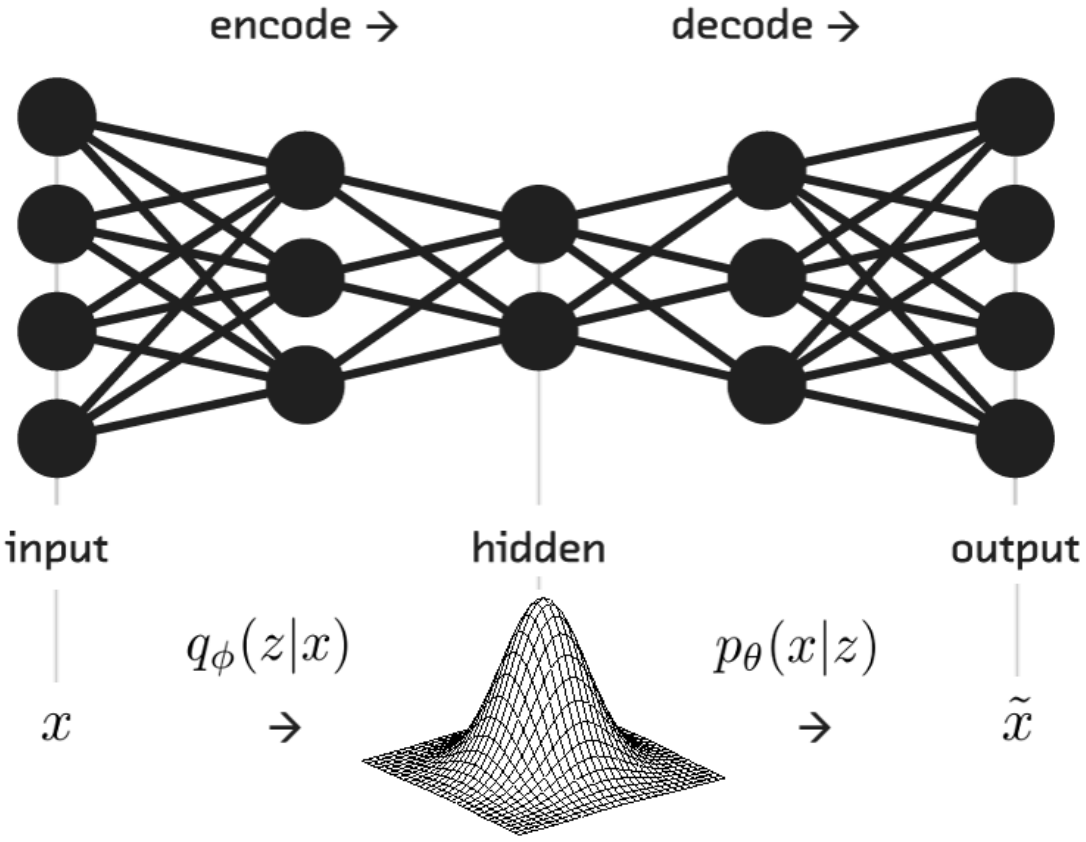
There are many algorithms that are capable of imputation like Non-negative matrix factorization, Singular value decomposition, KNN based methods, and Kriging. The issue with many of the matrix completion/data imputation is that none consider the relations that data sources might have with one another. This work is heavily based on “Variational Graph Auto-Encoders” (by Thomas N. Kipf and Max Welling from University of Amsterdam). The mentioned paper introduces the VGAE and shows the results on link prediction. This method uses the graph convolutional networks, which is extending the idea of convolutional neural networks to arbitrary graphs, and variational auto-encoders (deep learning method for data reconstruction and feature learning). The result is that a masked version of the input data, which in this case is a traffic signal on a graph constructed from the data mining subsystem, is fed to this neural network and the model is evaluated based on how well it can reconstruct the masked elements.

3.2.1. Variational Auto-Encoders

Auto-encoders are used for reconstruction of an input. Their general architecture includes dense or CNN layers that map the input data to a smaller space, and then another few layers to reconstruct the input. The loss function is usually the distance between the input and the prediction. However, the more promising model that is an extension of the auto-encoders, variational auto-encoders, adds more flexibility to the model to help with the reconstruction. The latent space (the space that the input data is mapped to) is consistent of a mean and standard deviation vector which then is used to construct a gaussian model. There are samples drawn from

this distribution which then are propagated through the network to output where they reconstruct the input signal.

Figure 28: Architecture of variational auto-encoders



The reparameterization trick is used to propagate gradients (backpropagation) through the mean and standard deviation. The weights of these two layers are learnt as a normal procedure, however, they share the same input from the last layer. To form the latent space, as shown in the figure above, a unit normal distribution is used via the leant mean and standard deviation.

3.2.2. Variational Graph Auto-Encoders

The notation of convolution can be extended to graph data. The Laplacian matrix, which is defined as the subtraction of the adjacency matrix from the degree matrix, is the most important part of a graph convolution operation. The cost function of this model has the divergence between a normal distribution and the distribution that is being learnt as an extra term. Combined with the notation of graph convolution and variational auto-encoders, this model can leverage the underlying structure of the data and reconstruct the traffic signals defined on the road graph.

Figure 29: The loss function of this model; the beta term is called entanglement hyper-parameter, which is used as a constraint on the divergence term to control how close the latent distribution should be to a normal one.

$$L = \frac{1}{N}\sum(X - \hat{X})^2 + \beta D[q(z|x)\|p(z)]$$

Figure 30: The convolution operation is defined as follows:

$$f_{\mathrm{GCN}}(\boldsymbol{X}) = \sigma\left(\tilde{\boldsymbol{D}}^{-\frac{1}{2}}\tilde{\boldsymbol{A}}\tilde{\boldsymbol{D}}^{-\frac{1}{2}}\boldsymbol{X}\boldsymbol{W}\right)$$

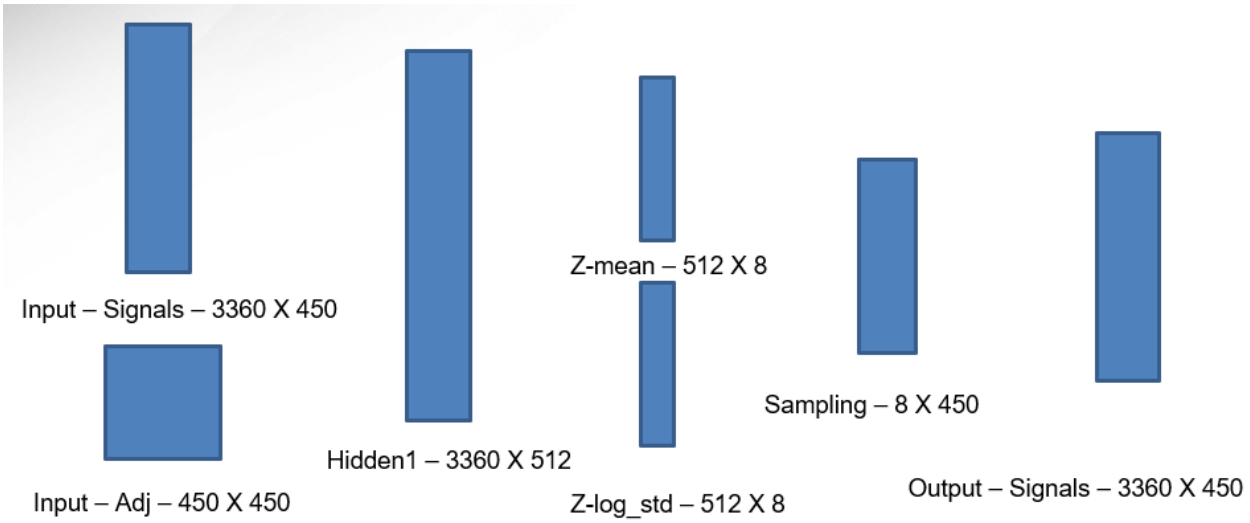
$$\tilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}_N,\ \tilde{\boldsymbol{D}}_{ij} = \sum_j \tilde{\boldsymbol{A}}_{ij},\ \boldsymbol{X} \in \mathbb{R}^{N\times T}$$

In the figure above, A is the adjacency matrix and D is the degree matrix.

3.2.3. Variational Graph Auto-Encoders – Numerical Results

The input data for this experiment was a graph that was constructed using the data gathered from the Zurich public transportation system. For each timestep (10-minute intervals), a traffic signal which consist of a normalized travel time for a road segment was constructed as described in the data pre-processing subsystem. The resulting data is a graph with 450 nodes, where each node represents a road segment in Zurich where the public transportation (busses and trams) travel. The Euclidean distance was used for defining the edges; if the middle of the road segment (GPS data) was within the 1500 radius of any other road segment, those two roads would be connected. All of the validation and reasoning for this modeling decision was described in the previous subsystem sections. A signal matrix which has the dimension of 450 X 3360 represent the traffic values described above. The adjacency matrix is the only meta-data needed from the graph itself which has the dimension of 450 X 450. After fine tuning, the final architecture for this model is shown in figure 31.

Figure 31: Architecture assumed for this problem.



The model is trained for 50 epochs; the split between the train and test portion is 30/70, 50/50, and 70/30. The learning is done on the matrix where the entire matrix is masked according to the training set, and the rest is used to evaluate the model. The beta is chosen to be 0.1 and initial learning rate 0.001.

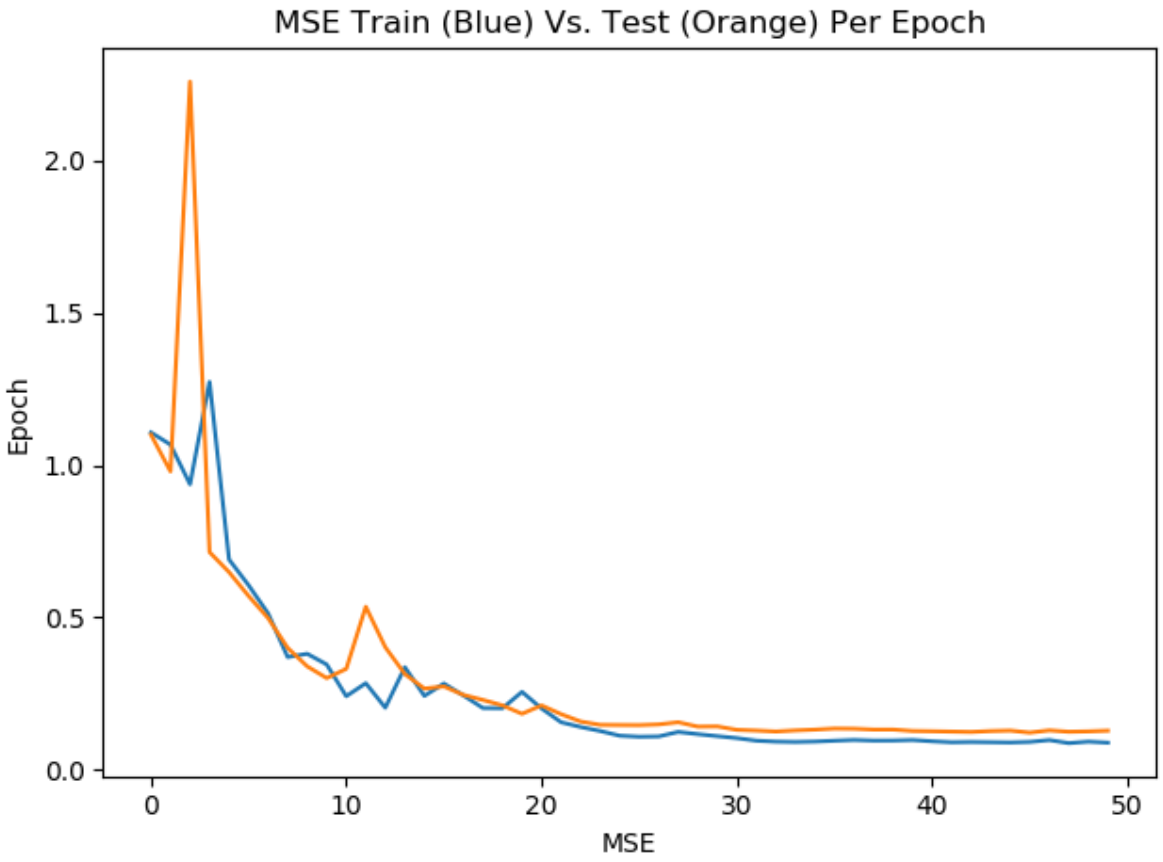
To validate the results, a common matrix completion algorithm, non-negative matrix factorization is used on the same dataset (with the same masks). The method for evaluation is mean squared error and mean absolute error.

Table 4: The error metrics for the VGAE (our model) and NMF

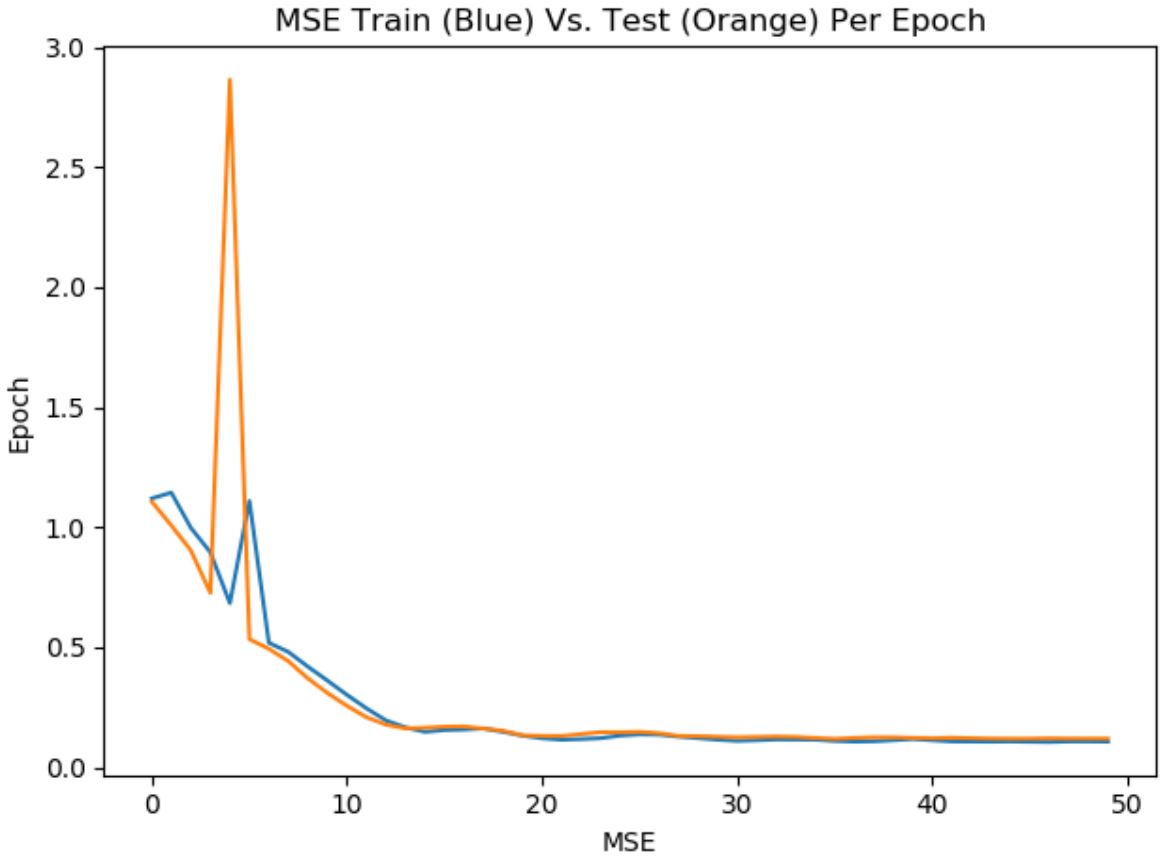
	Train - MSE	Train MAE	Test MSE	Test MAE
VGAE – 30 / 70	0.08584	0.20565	0.12389	0.20893
NMF – 30 / 70	0.58341	0.71182	0.15195	0.30882
VGAE – 50 / 50	0.10556	0.19977	0.12298	0.20151
NMF – 50 / 50	0.318279	0.508246	0.32541	0.51136
VGAE – 70 / 30	0.10327	0.20458	0.14726	0.20645
NMF – 70 / 30	0.16413	0.30715	0.54688	0.71582

As shown on Table 4, the VGAE method is performing better than NMF. The VGAE is naturally incorporating the underlying structure of the graph and leveraging the relations between these signal values. It is important to note that with the decrease in the number of samples known to the model, NMF's performance drops rapidly, but, VGAE sees a lesser impact due to this. The VGAE is implemented in TensorFlow (1.3 with GPU support for training), and for NMF, the sk-learn library was used.

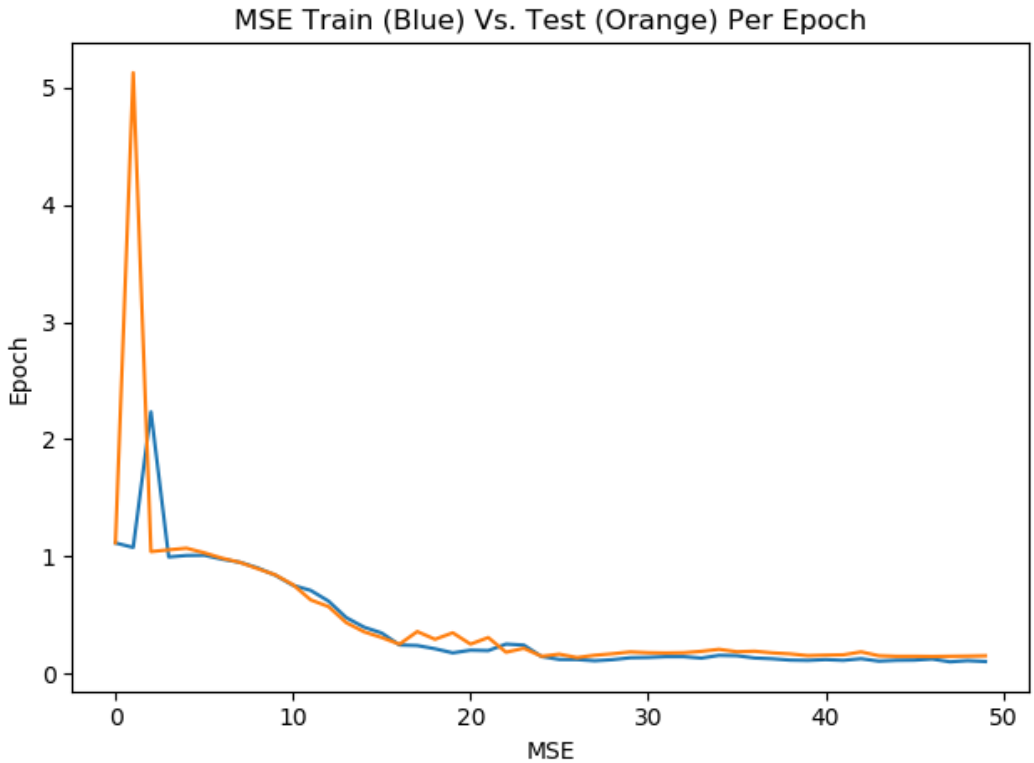
Figure 32: The train-test mean squared error with each epoch for all splits



Train 30 / Test 70



Train 50 / Test 50



Train 70 / Test 30

3.3 Subsystem Conclusion

Algorithms and models that are being designed to use graphs as inputs are on the rise. Traffic is the classic example of the usage of a graph, but there are many other problems that prove to benefit from significantly from incorporating the already known relations between the entities present. This model shows that the imputing of missing data points can be done more accurately when some underlying assumption is present and known.

Many of the modeling decisions were made using trial and error. The choices for the normalizing factor, intervals, and architecture were all supported by data presented in the exploratory data analysis subsystem, although, it was crucial to keep the signals as pure as possible. So, the interpolation was kept to only 5 points in total as it is crucial to keep the integrity of the signals since this data was used to validate an experimental algorithm.