UNIVERSITY OF TARTU

Institute of Computer Science

Computer Science

Joonas Praks

Anomaly Detection and Imputation for Tartu Traffic Sensors

Master's Thesis (30 ECTS)

Supervisor: Pelle Jakovits, PhD

Tartu 2024

Anomaly Detection and Imputation for Tartu Traffic Sensors

Abstract:

The city of Tartu has 16 highway traffic sensors with many gaps of missing data. We analyzed the state of the sensors' data and evaluated different anomaly detection and imputation solutions to better its quality. The best anomaly detection approach was deemed to be daily clustering with local outlier factor (LOF) used as the clustering algorithm. For imputation we utilized linear interpolation with a combination of seasonal decomposition and seasonal splitting. The chosen solutions were integrated into a service that processes CSV files of traffic data and uploads the results to Cumulocity, an IoT data aggregation platform. We processed and uploaded the historical data of 2019-04-29 to 2023-06-01 of every highway sensor. Finally, we also tested our solution on light traffic data.

Keywords:

sensor, Tartu Smart City, traffic, anomaly, outlier, imputation, Cumulocity

CERCS:

P170 Computer science, numerical analysis, systems, control

Anomaaliate tuvastamine ning andmelünkade täitmine Tartu liiklussensorite näitel

Lühikokkuvõte:

Tartu linnal on 16 maantee liiklussensorit, mille andmetes esineb mitmeid auke. Me andsime ülevaate andmete olukorrast ning hindasime mitmeid anomaaliatuvastus- ning andmeparandus-lahendusi. Anomaaliaid leidsime kõige paremini päevase klasterdamise abil kasutades LOF algoritmi. Imputeerimislahenduseks valisime lineaarse interpoleerimise kombineerides ajaandmetes leitud hooajalisi mustreid. Me integreerisime valitud meetodid teenusesse, mis töötleb CSV andmeid ning laeb tulemid üles Cumulocitysse, IoT andmete agregeerimisplatvormile. Me töötlesime ning laadisime teenuse abil üles sensorite ajaloolised andmed vahemikus 2019-04-29 kuni 2023-06-01. Lõpetuseks katsetasime oma lahendust ka kergliiklusandmetel.

Võtmesõnad:

sensor, Tartu Smart City, liiklus, anomaalia, imputeerimine, Cumulocity

CERCS:

P170 Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria)

Contents

1. Introduction

In 2023 there are roughly 6 billion connected non-consumer IoT devices in the world (1). Sensors fitted to those IoT devices are a source of data that aids us in different decision-making scenarios. With sensor data we can establish historical patterns and make educated guesses about the future.

The work at hand was requested by the city of Tartu. In relation to the SmartEnCity project from 2016 until 2022 sensors of varying type were installed in Tartu. Among them were traffic sensors, namely sensors for highways and sensors for the inner city (2). The chief motivation for installing vehicle counting sensors is to be able to predict and evaluate the traffic load that any part of the city faces. This helps to plan for upcoming road constructions and maintenance but also for city-wide events to aid in alleviating possible bottlenecks and traffic jams.

These goals, however, are hindered by the quality of the data. Namely the data that the sensors have aggregated over the years 2019-2023 face several issues. Based on an initial overview the dataset for each of the observed sensors is missing several days' worth of data and also has a portion of the recorded events duplicated. A third breach in data quality is also likely since for some days there is a severe mismatch between the average seasonality and the recorded data for these days.

The city of Tartu has outlined three requirements that would help them to combat the poor quality of data. First, an imputation should be performed on the historical data sets of their sensors, that is the gaps in the data should be filled with some reasonable synthetic data. Second, they would like to continue these imputations on possible future data gaps on a daily basis. Finally, a mechanism should be in place to detect and annotate gaps and anomalies in the data. In this thesis we mainly look at Tartu's highway sensors, but ideally the solution for those requirements would provide good results for the data of other traffic sensors as well.

This thesis aims to find a solution to the aforementioned requirements and also offer some insight into the underlying data of the highway sensors. First, we explore the original dataset by assessing its quality and discovering patterns. Following that we try out and judge different approaches for data imputation and anomaly detection (from here on out AD). Then we create a service that encapsulates the chosen solutions for imputation and AD, and automates processing time series data and uploading the results to Cumulocity, a data aggregation platform.

1.1 Research objectives

- Analyze the sensor data, finding patterns that will aid in developing an approach for data imputation and AD
- Evaluate different solutions for imputation and AD
- Develop a service that is able to
  - detect anomalous data
  - replace missing gaps with synthetic data based on historical data
- Evaluate the quality of the created service

1.2 Research questions

- Is it possible to have a generic approach to automate data imputation and AD for various traffic sensors given the data quality of Tartu sensors?
- How well can we measure the results of our chosen imputation and AD approach given that our data is not labeled?

1.3 Thesis outline

In the Background chapter we will give a brief overview on the concepts that the thesis builds upon. Then in Input Data Analysis we look at the different features of the underlying data that we are working with. In Methodology we see the chosen solutions for AD and data imputation, but also some alternative approaches that were tested out. Following, in Implementation we go over some details of the created AD service, such as the selected tools and data flow. Then in Evaluation we assess the quality of the AD services output. In Extended Use Cases we give a comment on the different ways that the AD service could be used. Finally in Conclusion and Future Directions we look at the shortcomings of the AD service and what could be improved in the future.

2. Background

Here we briefly cover a set of concepts further utilized in the thesis. We also look at the different anomaly detection and imputation tools that we will bring under evaluation.

2.1 Smart City

A smart city is the concept of using information and communications technology (ICT) to improve and optimize the services of a city such as administration, education, logistics, utilities and healthcare (3). This is a broad definition as every city may focus on its own set of verticals. The city of Tartu can also be considered a smart city. It has a bike rental service, a program for installing smart home solutions and employs a variety of sensors for metrics such as traffic, air pollution and noise among other things. These IoT solutions were obtained in the course of the SmartEnCity project during the years 2016-2022. Some of the directions that this European Union funded project took were the renovation of several *khrushchevka* type apartment buildings, creating a smart bike share system and reducing energy consumption in streetlighting (2). Considering the project's objectives, the type of Smart City that Tartu represents would be better described by another definition. We would say that a smart city will take advantage of communications and sensor capabilities sewn into the cities' infrastructures to optimize electrical, transportation, and other logistical operations supporting daily life, thereby improving the quality of life for everyone (4).

2.2 Cumulocity

Cumulocity is an IoT platform for data aggregation, role management and dashboard creation. It is actively developed and maintained by Software AG, a German software corporation. (5) Cumulocity was chosen as the supporting platform for the SmartEnCity project by Telia for its maturity, reasonable latency and good documentation. Cumulocity also has the added benefit of enabling to create devices by combining small logical fragments. (6)

The main aspects of Cumulocity's domain model also important for this thesis are the Inventory, Events and Measurements. Inventory stores all the identity and hierarchical data of devices. Events are real time records stored in response to some trigger e.g. a light switch being turned on. Measurements are regularly spaced collections of numerical data e.g. temperature measurements every one hour. (7) We communicate with Cumulocity through their REST API layer.

2.3 Anomalies in time series

According to Cook et al. an anomaly is an abrupt change in a system's state that does not match the usual norms either in a local or global sense (8).

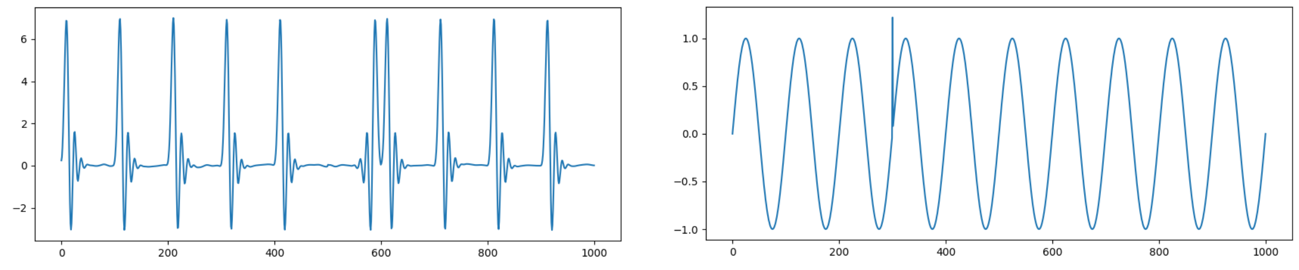


Figure 1: An example of a local and global anomaly respectively

Anomalies can stand out from regular data in three ways. A data point can be considered a point anomaly, a data pattern can represent a collective anomaly and both data points and patterns can form a contextual anomaly:

- In the work of Al-Amri et al. a point anomaly is a data point that falls out of the expected range for its domain. It can be labeled an outlier even when observed in isolation. Such anomalies often return to their normal ranges after a short time. (9)
- Fahim et al. explain that data considered contextually and conditionally anomalous appears ordinary if secluded. It raises suspicions only when viewed in relation to its neighboring data. (10) For example, it may be normal for the temperature in Estonia to be cold during winter but not in summer.
- Cook et al. describe collective outliers as data whose individual observations are not anomalous but, in a sequence, become implausible (8). Such anomalies can be easily seen in data series with clear patterns such as heartbeat measurements.

It is important to note that this definition makes no assumptions about the source of the anomaly. Some anomalies are caused by the drastic differences in the observed metrics: environmental or behavioral changes. Others are technical: faults in the sensor or data pipeline, or human interference.

2.4 Anomaly detection in time series

Anomaly detection is a well-researched topic with many proposed solutions. We delve into different AD solutions through TimeEval, an automated tool for testing anomaly detection algorithms for uni- and multivariate time series. It ships with its own vast library of detection methods and datasets. The algorithms they provide span the categories of statistics, data mining and deep learning among others. (11)

Out of the entire 71 algorithm roster that TimeEval provides we only look at the unsupervised methods since we have no labels and thus no training data to consider supervised approaches. We further filter available solutions by relying on the statistics Wenig et al. provide like scores, amount of errors and runtime, measured both on real and synthetic data (11,12). Reflecting these statistics, the anomaly detection methods we consider in this thesis are Subsequence LOF, GrammarViz and VALMOD.

Subsequence LOF applies the local outlier factor on sequences of a time series. Sequences are created by applying a sliding window of a given length to the original time series. LOF is a density clustering algorithm with the intuition that the degree of being an outlier should be calculated differently in different parts of the data. The final AD score for each datapoint depends on the local density of a datapoint compared to the local density of its neighbors. LOF has a single parameter MinPts which is the number of nearest neighbors that the local density is calculated by. (13)

GrammarViz discretizes subsequences into symbols and infers grammar rules from the result. Non-anomalous sequences can be encoded compactly by the grammar rules. Subsequences with a low coverage of derived rules, however, are considered to be anomalous. (14) Variable Length Motif Discovery algorithm (VALMOD) helps to efficiently find motifs and discords in a provided time series without having to upfront define the length of a motif or discord (15).

2.5 Time series imputation

Imputing a time series is a popular topic in the academic literature with proposed ideas and systemic reviews spanning several decades. A big section of that body of work has been dedicated specifically for traffic data imputation, be it for series of speed measurements, vehicle counts etc. In Li et al imputation approaches are grouped into prediction, interpolation and statistical learning methods (16). We concentrate on a small set of these methods to find the best fit for imputing the data gaps in Tartu traffic data.

Prediction methods such as ARIMA and exponential smoothing all use preceding data to make estimations regarding a future time period (17). Since the city of Tartu has voiced that their requested solution is not a real-time solution then we can take the data that follows a data gap into consideration when making an imputation. Thus, we are not interested in prediction solutions in this thesis.

Interpolation is further broken down by Li et al. into temporal- and spatial-neighboring solutions and pattern-similar methods. Temporal-neighboring methods fill gaps by interpolating over historical and future data of a single sensor. Spatial-neighboring, however, interpolates the data by comparing spatially close sensors at the same time point. This approach is usually performed on sensors on the same road that are relatively close to one another (16). This is not the case with the city of Tartu where the 16 highway sensors observe separate roads.

Pattern-similar imputations like k-Nearest neighbors according to Li et al. group data points into days and then try to estimate missing values of an incomplete day by finding similar days by the values that do exist (16). Those solutions don't work well in cases where the entire day is missing. A lack of any data points means that it is not possible to look for similar days for estimation. Since missing days is the most common missing data pattern in Tartu highway sensors pattern-similar imputations are not a good fit.

Implementations for the statistical learning methods that Li et al. mention were unfortunately not able to be found. This holds true for many academic works proposing novel time series imputation solutions — no open software is provided to compliment the claims made in the paper. We identified two time series-oriented bodies of work where implementations are provided: a framework called PyPots and the research and Python notebooks created by Chen et al. PyPots concentrates on imputing, predicting and clustering on multivariate time series using mainly neural net-based solutions for imputation (18). We failed to achieve any meaningful results with PyPots whether it was for our inexperience with parameter tuning or for the quality of the data.

The research of Chen et al. tackles both multi- and univariate time series imputing and prediction. We look into the low-rank autoregressive tensor completion (LATC) and Laplacian convolutional representation (LCR) algorithms. LATC uses low-rank tensor completion to model the global aspects of time series, such as seasonality. To take temporal and spatial dependencies into account a temporal variation is introduced with the help of an autoregressive model (19). LCR is an alternative low-rank completion model that uses a Laplacian kernel and the properties of circulant matrices to make the imputation solvable with a fast Fourier transform (20). We also take into consideration a slight variation of LCR that instead of a univariate time series accepts a multivariate time series. We call that approach LCR-2d.

Lastly, we look at MICE, Multiple Implementation by Chained Equations. In iterations each column of a dataset is predicted by a fitted model trained on the other columns (21). This solution is not time series specific and does not take temporal or spatial correlation into account. However, it is a popular general data imputer and we use it as a benchmark against the more novel approaches.

3. Input data analysis

In this chapter we discuss the processing and shape of the underlying data of Tartu traffic counters. We assess its quality by analyzing the patterns of the missing data. Observations made here help us later during anomaly detection and imputation.

3.1 Data description

The data originates from 16 traffic counting sensors called AVC controllers that are placed around the city of Tartu Figure 2).

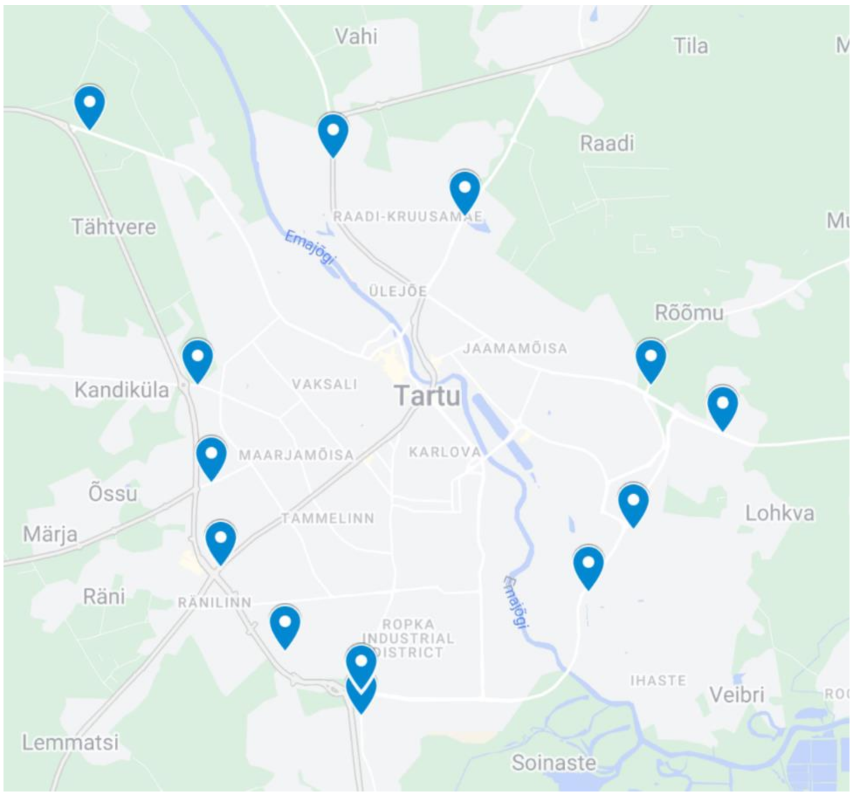
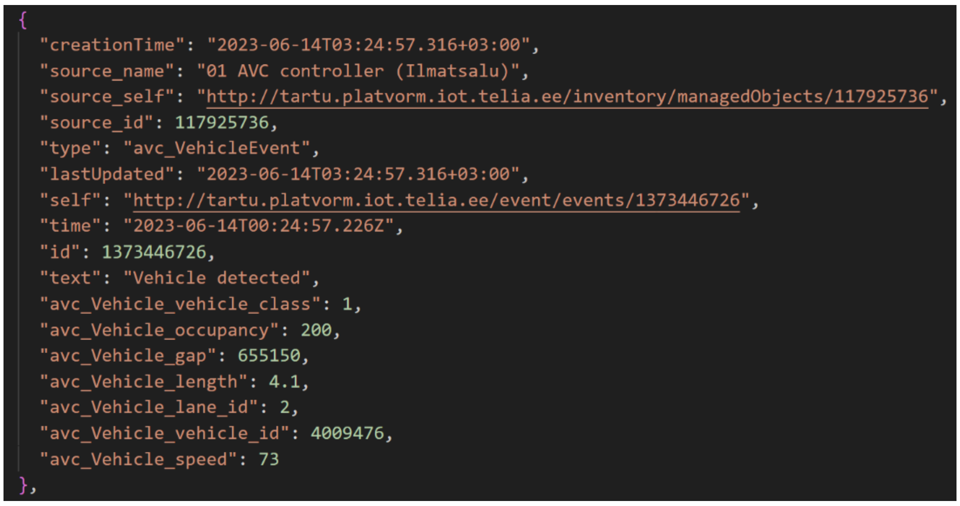


Figure 2: Locations of AVC controllers. The map is from Google Maps. Sensors' locations were taken from Cumulocity.

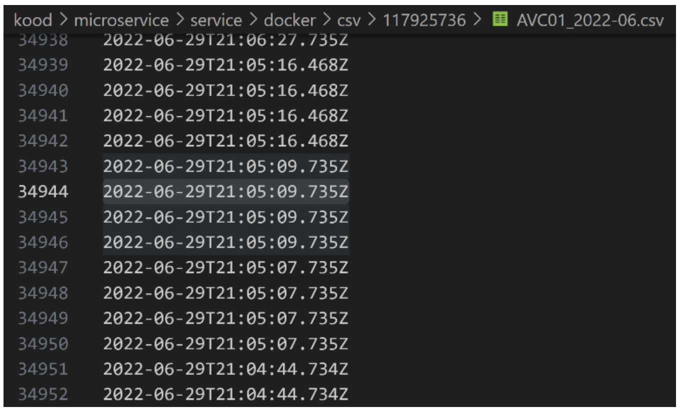
In addition to counting traffic the sensors are also able to for example determine the speed and size of the vehicles and the particular lane that a vehicle occupies as demonstrated by Figure 3.



```
{
  "creationTime": "2023-06-14T03:24:57.316+03:00",
  "source_name": "01 AVC controller (Ilmatsalu)",
  "source_self": "http://tartu.platvorm.iot.telia.ee/inventory/managedObjects/117925736",
  "source_id": 117925736,
  "type": "avc_VehicleEvent",
  "lastUpdated": "2023-06-14T03:24:57.316+03:00",
  "self": "http://tartu.platvorm.iot.telia.ee/event/events/1373446726",
  "time": "2023-06-14T00:24:57.226Z",
  "id": 1373446726,
  "text": "Vehicle detected",
  "avc_Vehicle_vehicle_class": 1,
  "avc_Vehicle_occupancy": 200,
  "avc_Vehicle_gap": 655150,
  "avc_Vehicle_length": 4.1,
  "avc_Vehicle_lane_id": 2,
  "avc_Vehicle_vehicle_id": 4009476,
  "avc_Vehicle_speed": 73
},
```

Figure 3: An example Cumulocity event from AVC 01

The observation of vehicles creates an irregular time series. We downloaded this data for each sensor in the form of CSV files using a Cumulocity integration solution (22). The time range that we are working with is from 2019-04-29 up to 2023-06-01. These series of observations have a lot of repeating records since the second half of 2021 as seen from Figure 4 and Figure 6.


kood > microservice > service > docker > csv > 117925736 > AVC01_2022-06.csv

```
2022-06-29T21:06:27.735Z
2022-06-29T21:05:16.468Z
2022-06-29T21:05:16.468Z
2022-06-29T21:05:16.468Z
2022-06-29T21:05:16.468Z
2022-06-29T21:05:09.735Z
2022-06-29T21:05:09.735Z
2022-06-29T21:05:09.735Z
2022-06-29T21:05:09.735Z
2022-06-29T21:05:07.735Z
2022-06-29T21:05:07.735Z
2022-06-29T21:05:07.735Z
2022-06-29T21:05:07.735Z
2022-06-29T21:04:44.734Z
2022-06-29T21:04:44.734Z
```

Figure 4: Example of duplicates in traffic sensor data.

Considering the time precision we are dealing with and the strong tendency for the repeating records to come in groups of four, we assume that those are duplicates and not multiple vehicles being detected at the exact same time. We deduplicate the data by only collecting unique timestamps (Table 1).

Table 1: Events and duplicates in the date range of 2019-04-29 to 2023-06-01

Sensor	Nr. of events	Nr. of unique events	Percent of duplicates
AVC01	7079933	6091924	13.96
AVC02	19727450	9318225	52.77
AVC03	6831859	5998398	12.20
AVC04	10272380	7359636	28.36
AVC05	7224224	3618080	49.92
AVC06	27263221	13747548	49.57
AVC07	16793166	8640523	48.55
AVC08	11969475	5813543	51.43
AVC09	15398101	8316529	45.99
AVC10	4708191	2208482	53.09
AVC11	11626986	5460864	53.03
AVC12	23483590	11929854	49.20
AVC13	4778183	2259833	52.71
AVC14	27399051	13056700	52.35
AVC15	19750268	9584468	51.47
AVC16	6403190	3189108	50.20

We aggregate the data to 15-minute bins to help visualize and analyze the data while keeping the loss of data granularity to a minimum (Figure 5). We also localize the UTC times to the Estonian time of UTC +2/+3 since the observed traffic operates under that time zone.

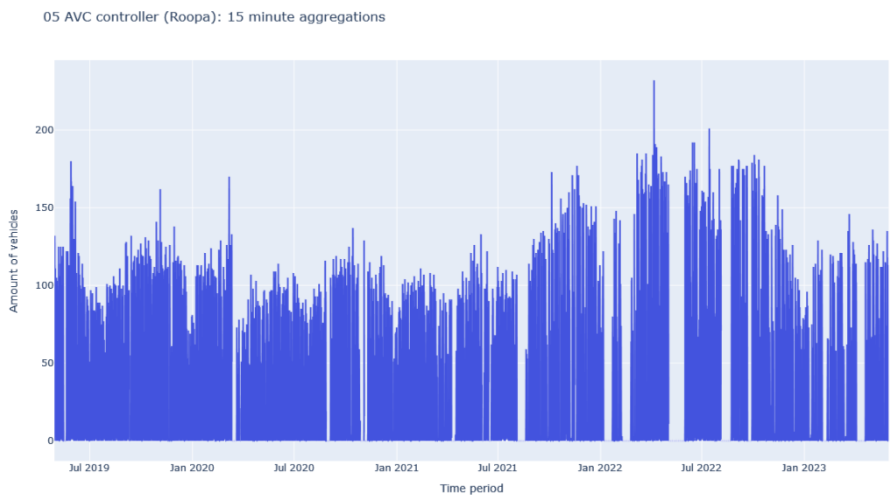


Figure 5: 15-minute aggregations for the 05 AVC controller

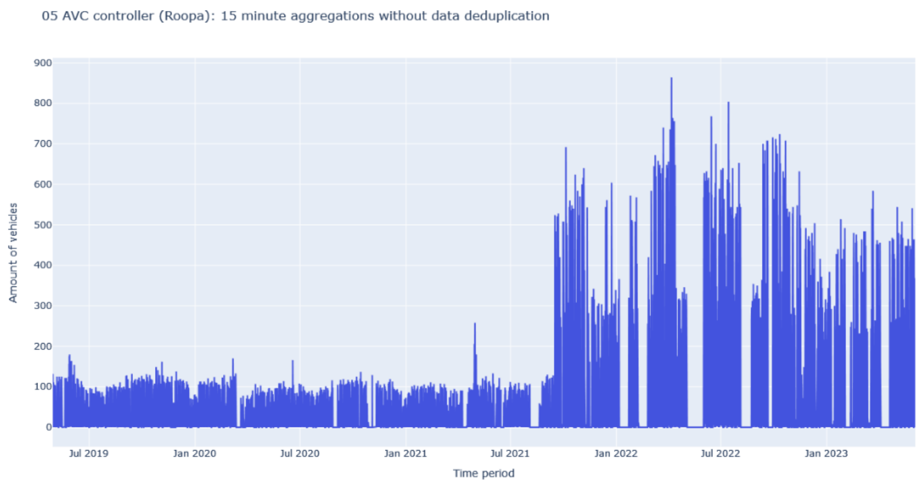


Figure 6: 15-minute aggregations for the 05 AVC controller without data deduplication

Averaging all of the workdays of any sensor shows a couple of things (Figure 7). First, the traffic volume is much higher during the daytime. Second, there are two rush hours: one roughly at 8:00 and the second at 17:00. These maximums are related to the start and end of workdays.

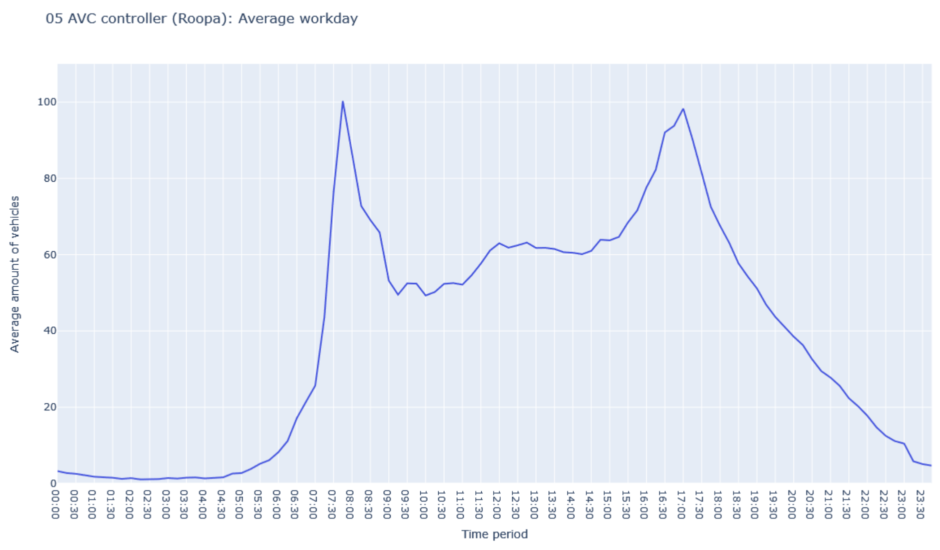


Figure 7: Average workday of the 05 AVC controller

For an average weekend there are no noticeable rush hours (Figure 8). The data volume is considerably smaller than on the workdays. There is a bit more nightly traffic during weekends. There is no sharp growth or decline in data rate at any time period.

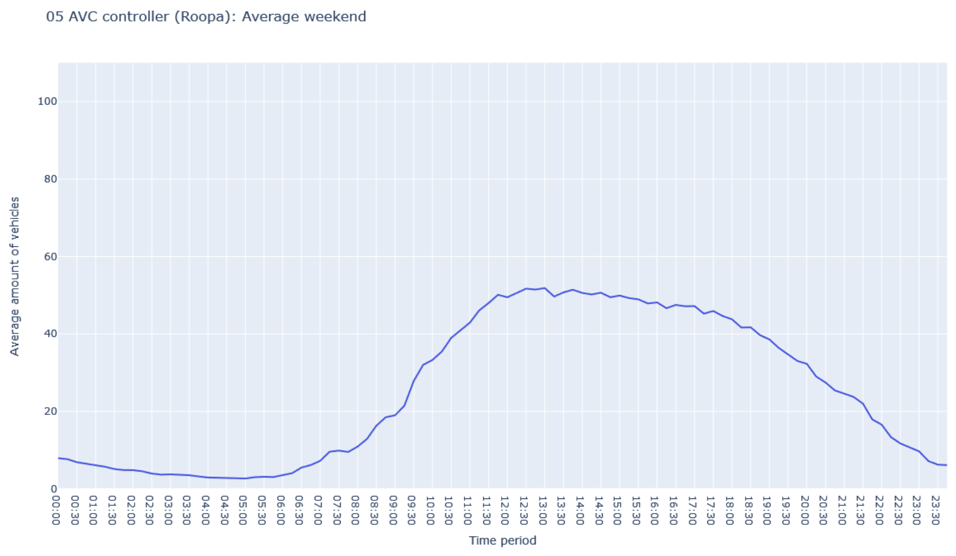


Figure 8: Average weekend of the 05 AVC controller

3.2 Data gaps

The data exhibits many periods during which no observations were registered. Those periods can range from 15 minutes to entire weeks. The amount, width and irregularity of those missing gaps makes it improbable that for every one of those gaps there were no vehicles to observe. Rather, the missing gaps are a combination of both missing vehicles and technical faults or human interference.

3.2.1 Nighttime gaps

The distribution in Figure 9 shows that there are more data gaps during the nighttime. Since there is much less traffic during the night (Figure 7), it is plausible that in some 15-minute time spans during the night no vehicle has passed the traffic counter. To get a closer look at the gaps whose origin is anomalous we remove these nighttime data gaps.

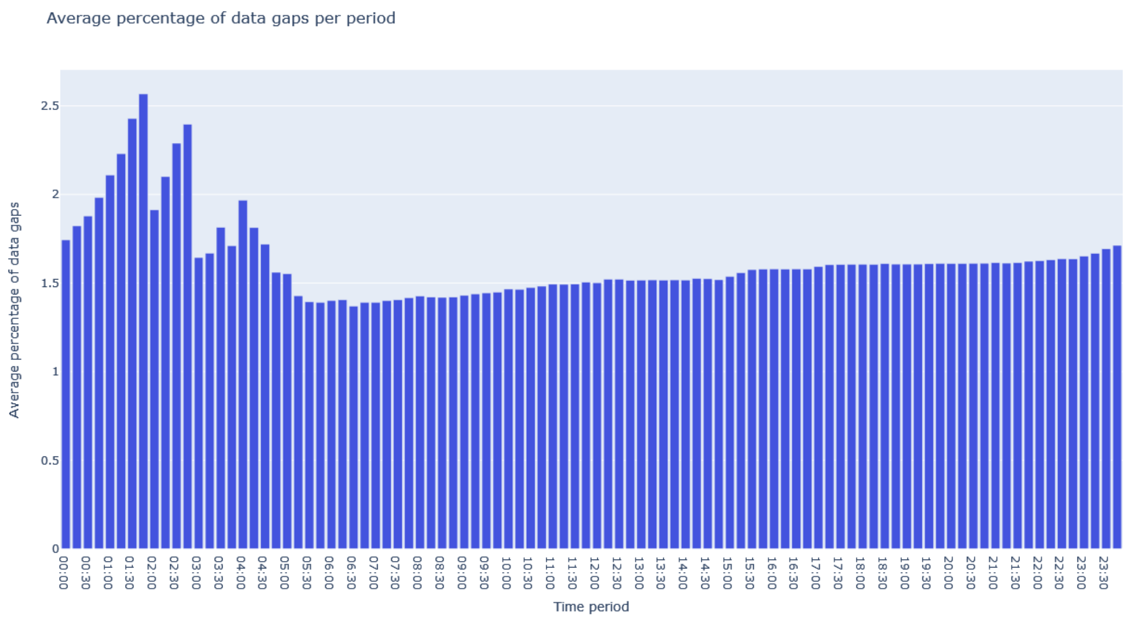


Figure 9: The average percentage of data gaps per period. E.g around 2% of 04:15 to 04:30 periods are missing on all days of all sensors. Days with zero observations were not accounted for.

To eliminate the nighttime gaps, we establish a static period between 23:00 and 6:00. The selection was done on the basis of Figure 9. The choice could have been more nuanced, each sensor could have had their own slightly different night span. However, the sensors' nighttime boundaries were similar enough that the additional complexity was deemed unnecessary. A nighttime gap could be in theory actually caused by something other than the normal traffic behavior. We don't unfortunately have any evidence on that matter so we will label all gaps that happened during the night as nighttime gaps.

Between 71 to 93% of data gaps were located between our nighttime boundaries. The rest of those gaps are missing data that we later impute (Figure 10).

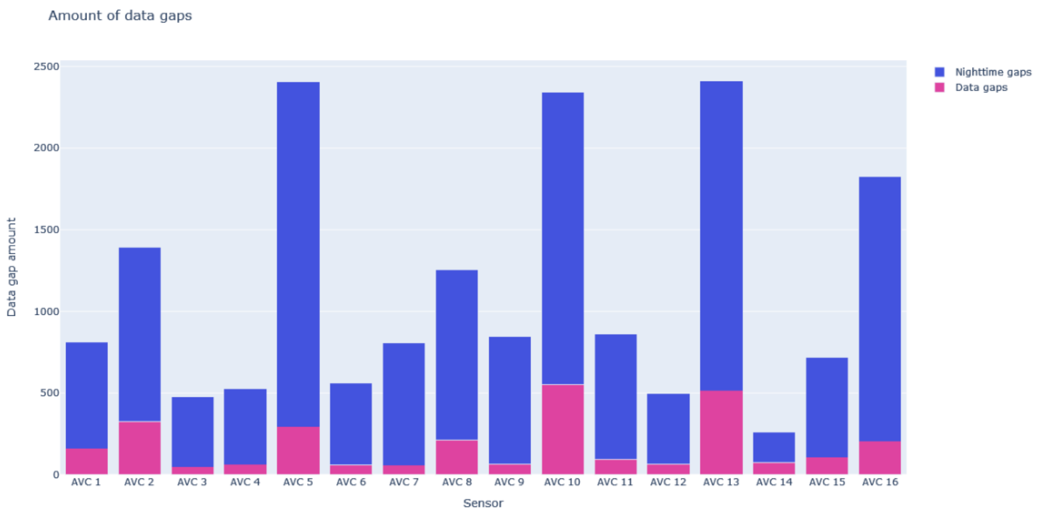


Figure 10 The amount of gaps in the data for each sensor. Nighttime gaps are defined by starting and ending between 23:00 and 6:00 within the same day or neighboring days.

3.3.2 Patterns in data gaps

After refining the data gaps, we can make some observations regarding the regularity of the gaps. The full spread diagram can be seen in Appendix II: Data gaps by duration. There is a clear indicator that the span of many of the data gaps is more or less one day (Figure 11). Looking at the distribution of the starting and end times of data gaps in Figure 13 we can see that most of the gaps both start and end during the nighttime. Paired with the previous remark this points to the fact that the gaps often start during the night and end at the next night, implying that they are not completely random.

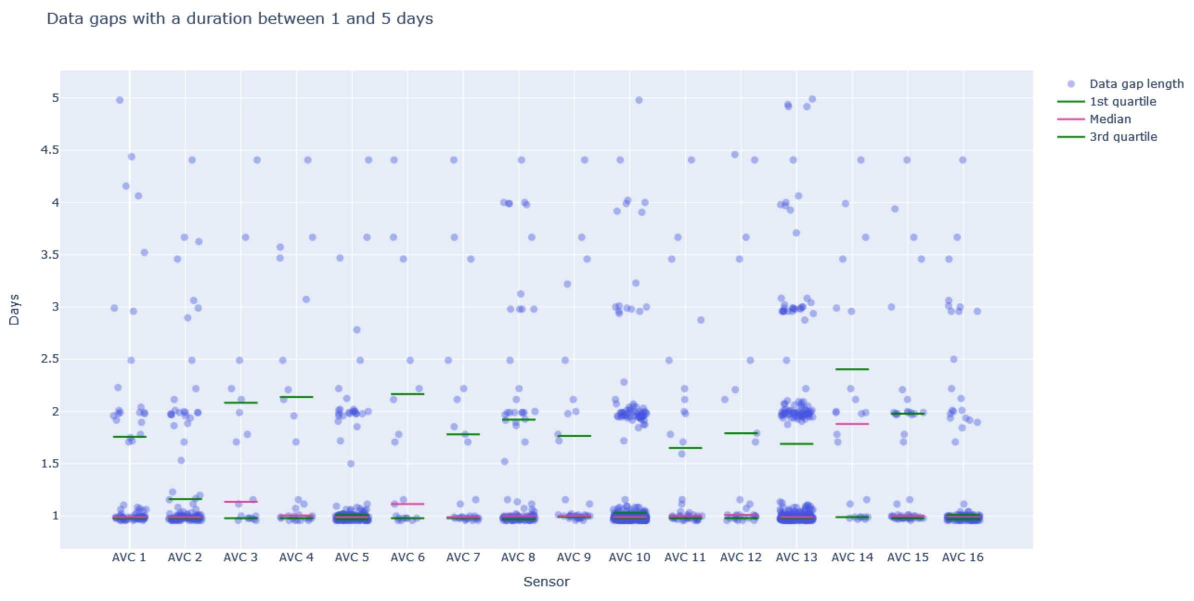


Figure 11: Data gaps with a duration between 1 and 5 days. Note the clear pattern of 1, 2 and for some sensors even 3 day long data gaps. See the full data gap distribution in the Appendix.

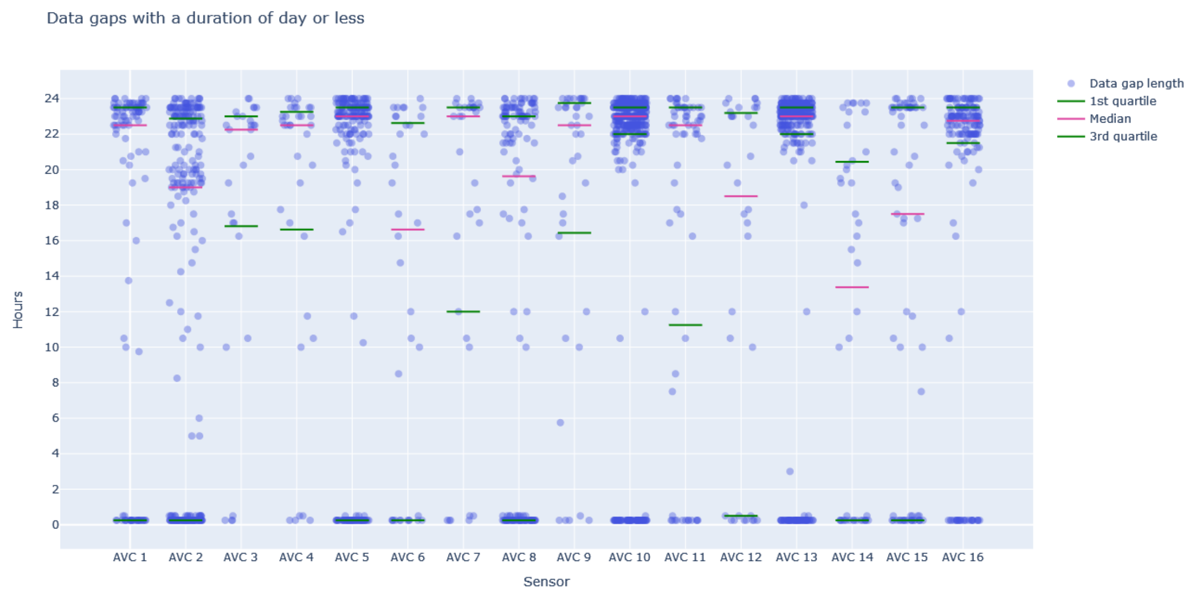


Figure 12: Data gaps with a duration of day or less. In the scope of a single day, data gaps are mainly only 15 to 30 minutes long, or take up the entire day.

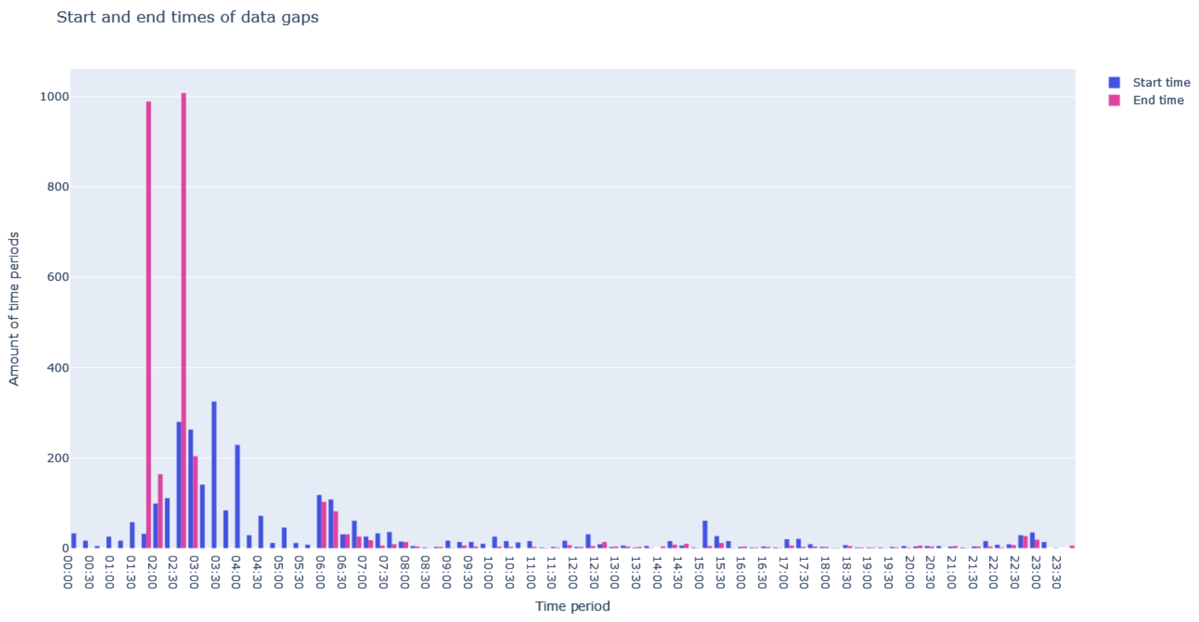


Figure 13: Start and end times of data gaps. A summation over all sensors showing an overwhelming majority of data gaps start and end at night. Note that in UTC time most of the end times would be during midnight.

There are 39 data gaps that exist in every sensor's dataset as seen in Figure 14. This insinuates that at times there have been overarching problems with the sensor management solution instead of the individual sensors.

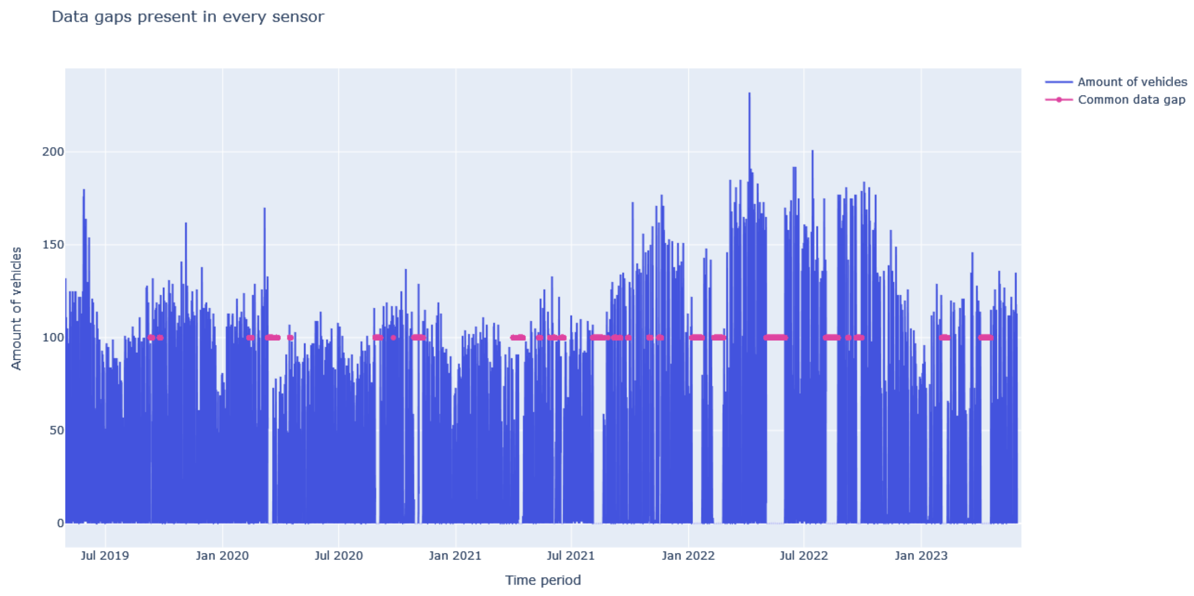


Figure 14: Data gaps present in every sensor. The underlying time series used for presentation is from the AVC 05 controller.

4. Methodology

We determine what we consider an anomaly in the Tartu sensors' data. In addition, we showcase the approaches we use for anomaly detection and imputation.

4.1 Definition of anomalies in the context of Tartu traffic data

Before proceeding with anomaly detection there should be some broad limits on what constitutes an anomaly in the framework of Tartu traffic data. There is no accompanying data or labels provided with the sensor data that would help us in making that distinction. Thus, we need some patterns to look out for when validating the output of our AD solution.

Detecting point anomalies is straight-forward in all domains given that the approximate range of normal data is known. Finding contextual anomalies, however, needs some understanding of the underlying data. We expect workdays to have a common seasonality. We expect the same from weekends. Also, in the scope of a week or a month, days should display similar traffic totals. In short, we expect the AD solution to consider a day to be anomalous if it deviates too much from the common seasonality or if its total traffic count is very different to its neighbors (Figure 15, 16).

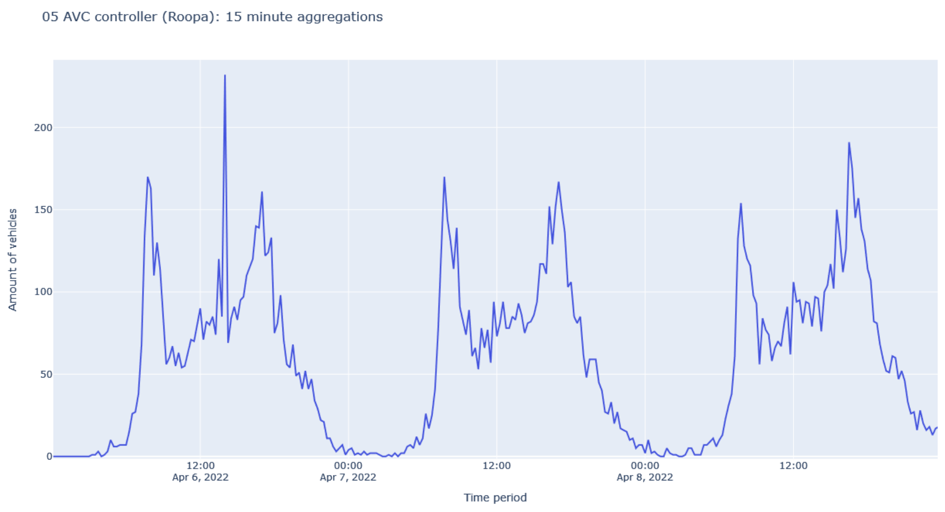


Figure 15: Example of both a point anomaly and a mismatch in seasonality

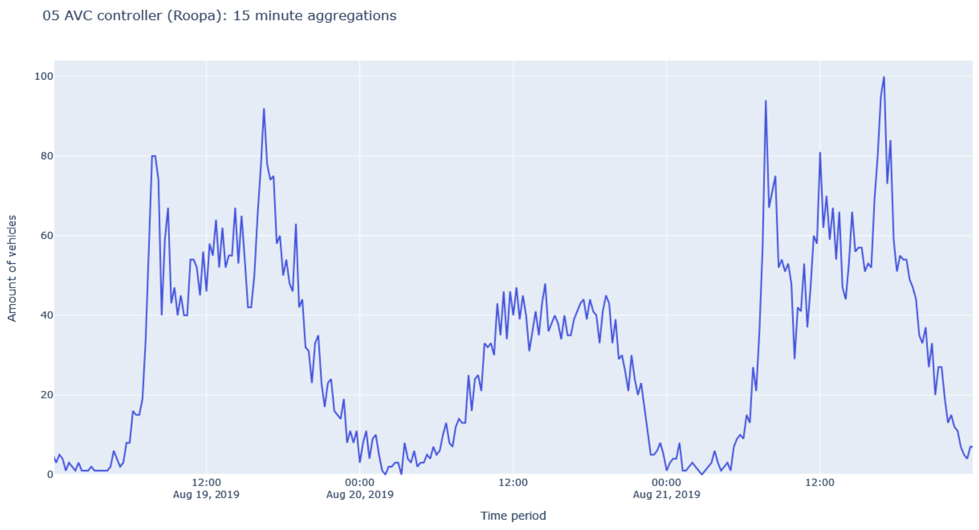


Figure 16: Example of a difference in traffic totals as well as a mismatch in seasonality

In addition, we can validate the AD solutions output with holidays. If there is a holiday during a workday, the data shape of that day always resembles more of a weekend. Since weekend and workdays have very different features the AD solution should be able to identify that holidays are anomalous.

4.2 Anomaly detection solution

In addition to testing out the TimeEval algorithms, we decided to also include a variation of the Sub-LOF algorithm into the evaluation. While Sub-LOF works with sliding window sequences we figured that given the strong daily patterns in the traffic data, we could extract days from a sensor's data as datapoints and cluster them together (Figure 17, 18). The outliers of clustering will be labeled as anomalies. Seeing that we chose 15-minutes as sampling size during data aggregation, each day is comprised of 96 features, one for each time period.

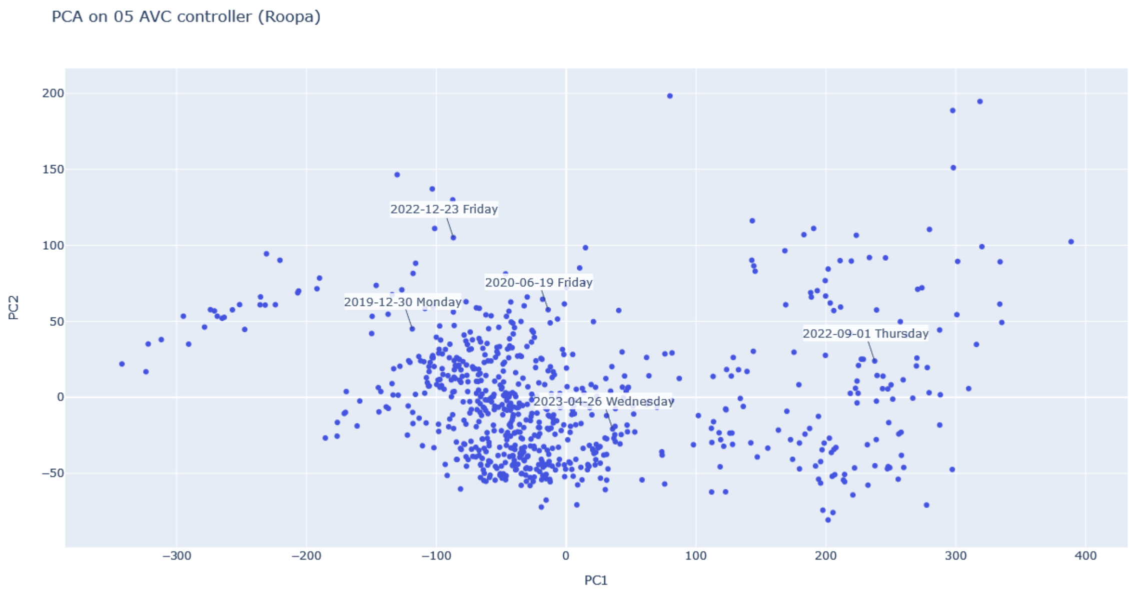


Figure 17: Principal Analysis Component on workdays of 05 AVC controller demonstrating that clusters are present in the data

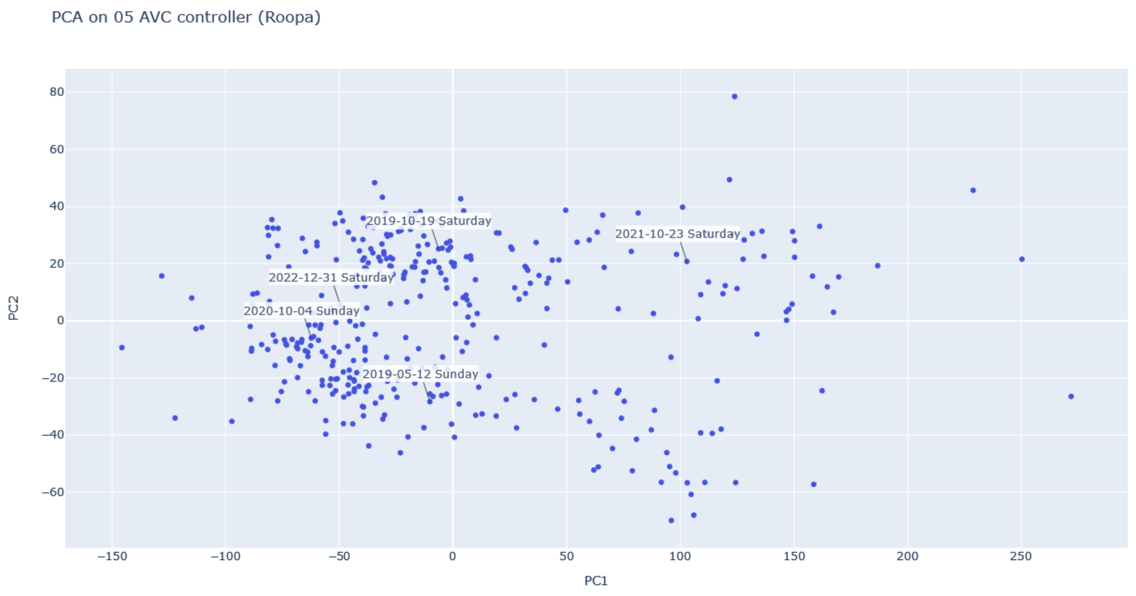


Figure 18: Principal Analysis Component on weekends of 05 AVC controller demonstrating that clusters are present in the data

We decided to cluster workdays and weekends separately as the patterns between the two are very different. This also ensures that holidays that take place during workdays are not clustered together with weekends, since the data shapes between the two are so similar. The best results were achieved by setting LOF's MinPts attribute as 30% of the number of inputted days.

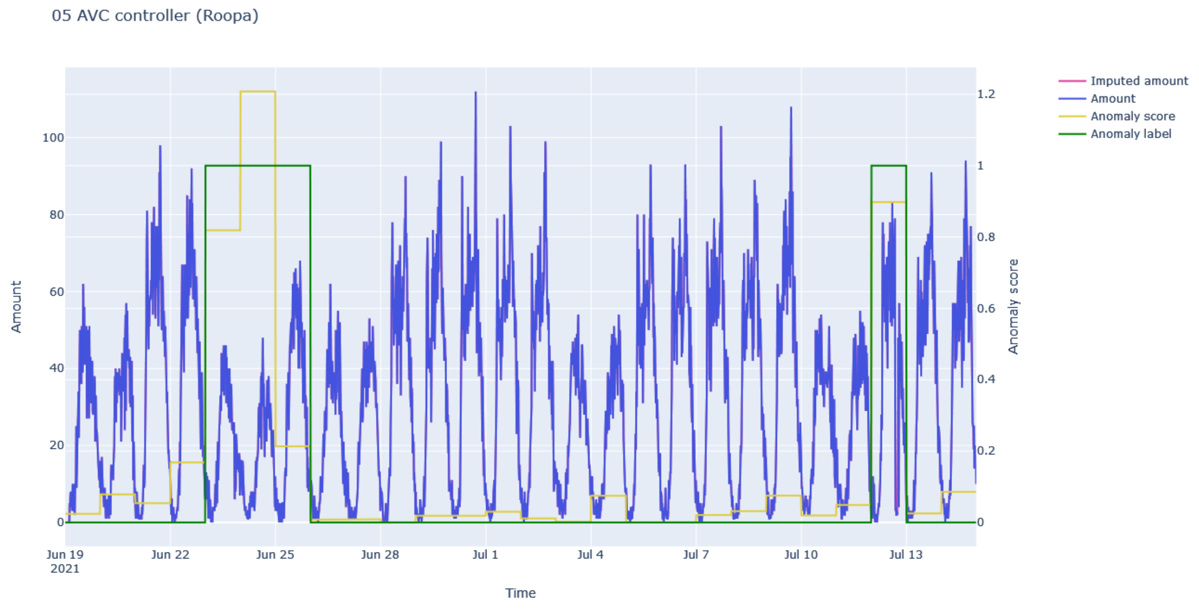


Figure 19: Excerpt from the output of the AD solution

LOF does not output a binary label regarding the anomaly status of data. Rather, it gives a score of outlier-ness to each datapoint. To convert each score to a label we use an upper bound from the IQR method of outlier detection. That is, we compare each LOF degree to the sum of the 3rd quartile and the interquartile range. (Figure 19)

We chose to perform AD only on days that have sufficient amount of data. Including days with big data gaps skews the AD: we would start to heavily identify days with data gaps as anomalies.

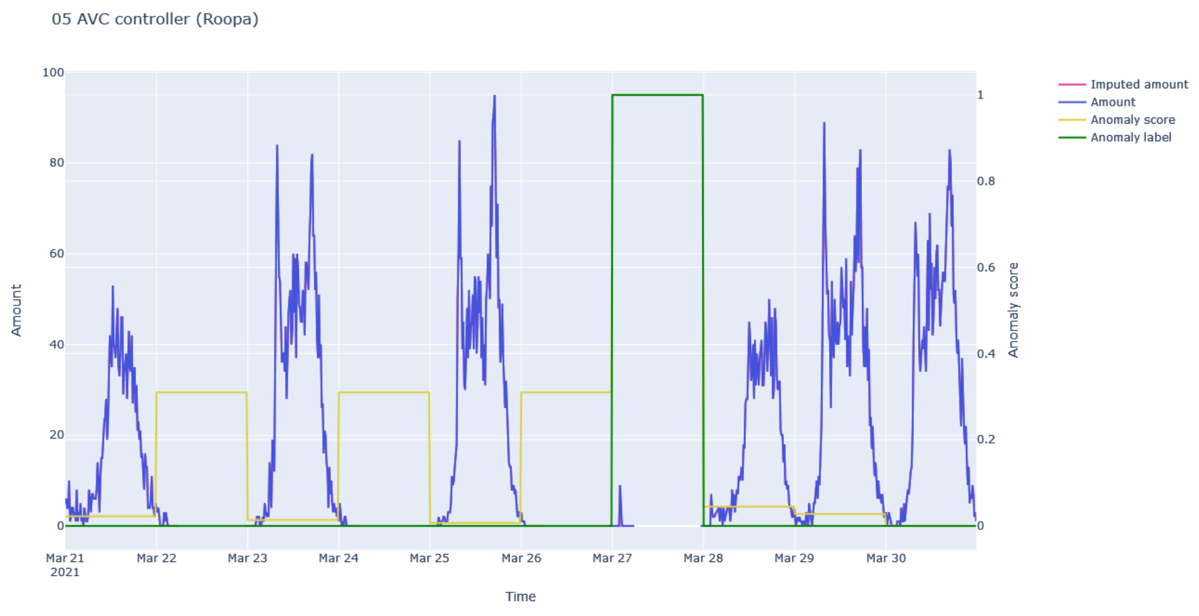


Figure 20: AD if we would not ignore days with data gaps. Note how the 22nd, 24th and 26th of March with almost the entire data set missing do not score high enough for the anomaly label to be raised since there is an abundance of days with similar properties.

While it is true that days with missing data should be deemed anomalies it is not very interesting since identifying them is trivial and does not require a sophisticated AD solution. Furthermore, as the number of days that have most or all of their data missing is very high the AD solution would start considering them to be a normal part of the data set (Figure 20). To annotate days with data gaps we have opted on using a separate flag (Figure 21).

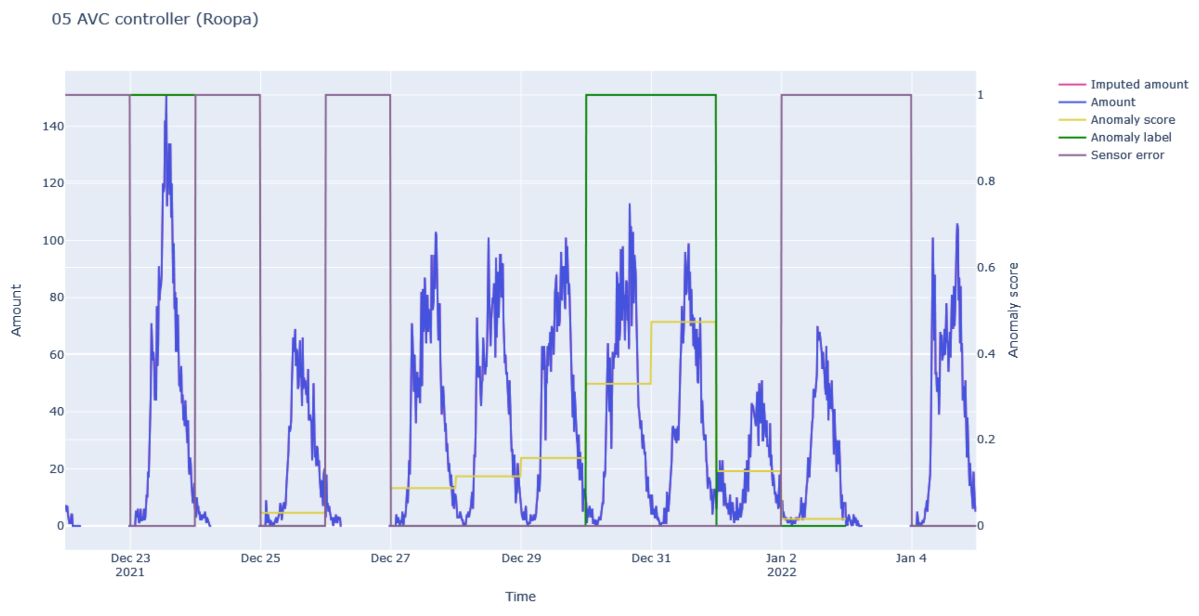


Figure 21: Excerpt from the output of the AD solution. Sensor error flag identifies days that have data gaps.

A trade-off in using clustering for anomaly detection is that the relationship between different days cannot be utilized. Only the shape of the datapoints is taken into consideration, but temporality is lost. Anomalies are detected in a global scope, not in smaller time frames like a week or a month. That means that a day that is clearly anomalous in the scope of one week may not be detected by the AD solution if that day is normal in respect to the entire data set.

4.3 Candidate anomaly detection solutions

In addition to our proposed solution, we tried to find anomalies from the Tartu traffic data with 3 additional methods: Subsequence LOF, GrammarViz and VALMOD. Sub-LOF and GrammarViz start with a preprocessing step of splitting the input time series into subsequences of fixed length with a step size of 1. This process is also known as applying a sliding window. We tried the candidate solutions with both window size 96 and 672, that is daily and weekly frequencies.

4.4 Imputation solution

We created a simple imputation method that takes into consideration historical and future datapoints. The solution employs a combination of interpolations based on seasonal decomposition and seasonal splitting. These ideas were borrowed from the R language library ImputeTS (23) as well as from the descriptions of imputation models in practice from Zhong et al. (24). Imputation with seasonal decomposition is used to only close small data gaps and is performed before AD. The imputation using seasonal splitting covers all the remaining data gaps.

4.4.1 Small gap imputation

Anomaly detection uses only days with no data gaps, nighttime gaps excluded. As seen in Figure 12 there are numerous data gaps that are up to one hour long. We fix these smaller gaps before AD to maximize the number of days available.

For imputing small gaps, a combination of seasonal decomposition and linear interpolation is used. Seasonal decomposition is a technique of subtracting seasonality from a time series leaving only a combination of trend and residuals (Figure 22). By applying linear interpolation to the deseasonalized data and then introducing the seasonality back to the data we are able to better represent the imputed data points as part of the underlying time series. Regular interpolation would not be able to replicate the correct data shape should the data gap span over a rush hour (Figure 23).

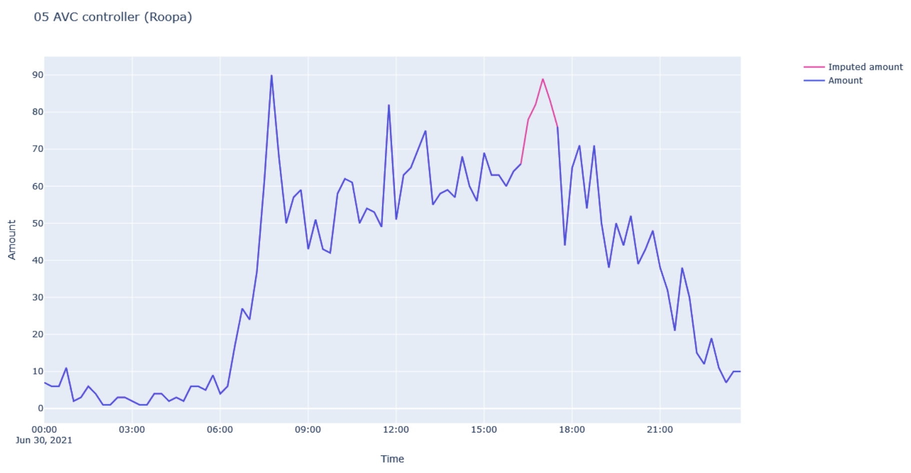


Figure 22: Small gap imputation with seasonal decomposition

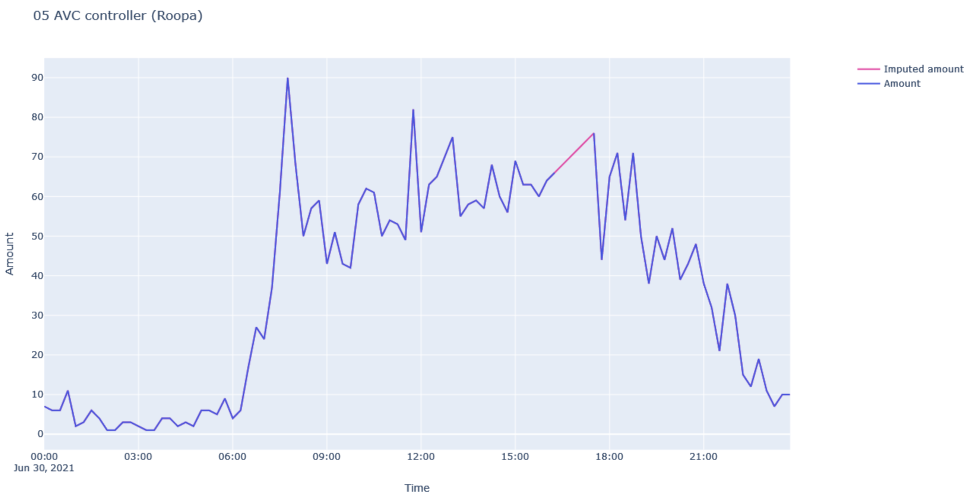


Figure 23: Small gap imputation without seasonal decomposition

4.4.2 Big gap imputation

Data gaps that are longer than one hour are filled with linear interpolation conducted on seasonally split data. Splitting helps to leverage values from neighboring data when calculating an interpolation. The autocorrelation in Figure 24 proves that a strong daily correlation exists in the vehicle count data. In other words, we have evidence that a data gap at 12:00 in any day could be imputed based on the values that the neighboring days have at 12:00. Thus, we could split our data by daily seasonality.

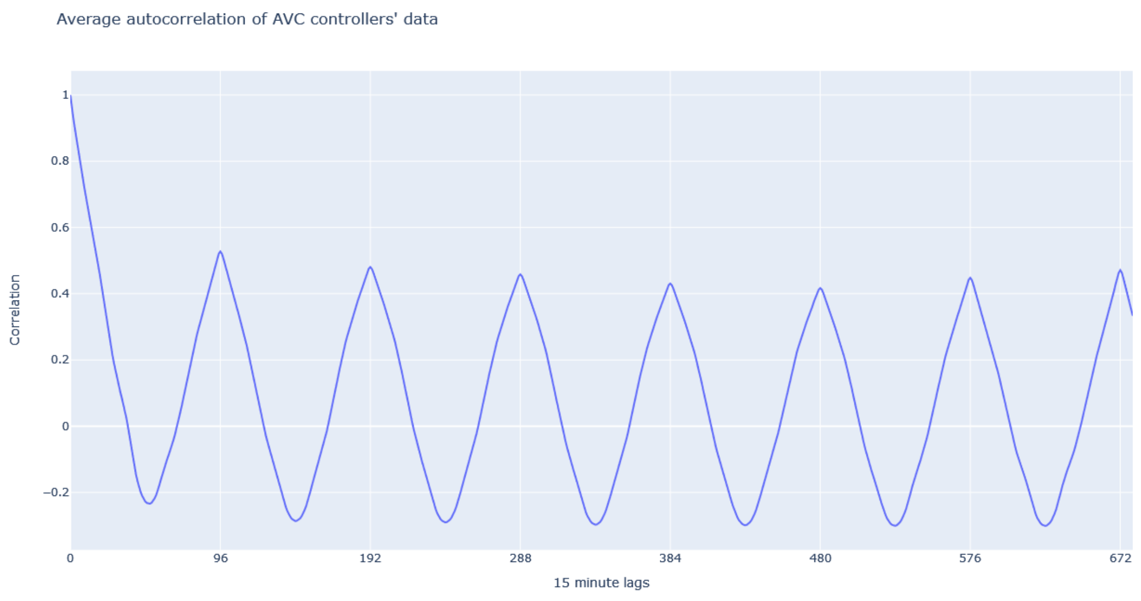


Figure 24: Average autocorrelation of AVC controllers. Data has been adjusted for daylight saving times. Note that a lag of 96 means 1 day.

In addition to the daily seasonality, we also decided to employ weekly seasonality. That is, we interpolate over a series of weekdays at a specific time period (Figure 25). A combination of 0.4 daily interpolation and 0.6 weekly interpolation gives the best results.

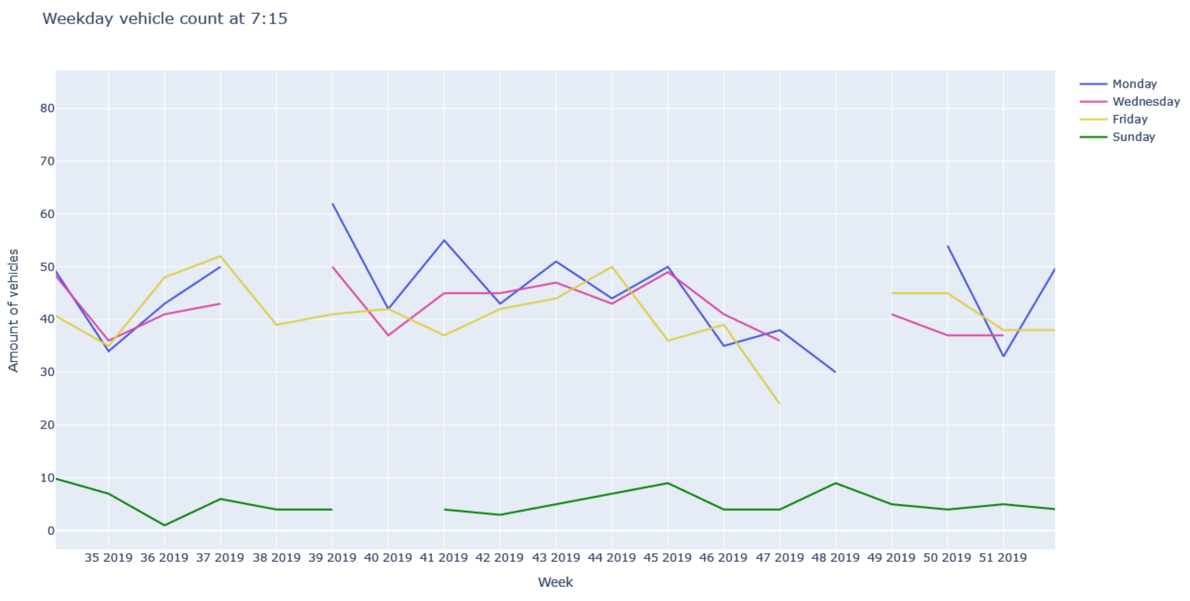


Figure 25: Excerpt from seasonally split vehicle count. Each figure is a series of a weekday at a fixed time. The underlying dataset is from 05 AVC controller

Seasonal splitting has better results than seasonal decomposition in scenarios where the entire day is missing. Instead of showing an average synthetic day we formulate the imputed day on the average of the next and previous days.

Thanks to the big gap imputation being performed after AD we are able to take note of the days that are identified as anomalies. To avoid propagating anomalous data to our synthetic data we forbid these days to participate in the imputation process. (Figure 26, 27)

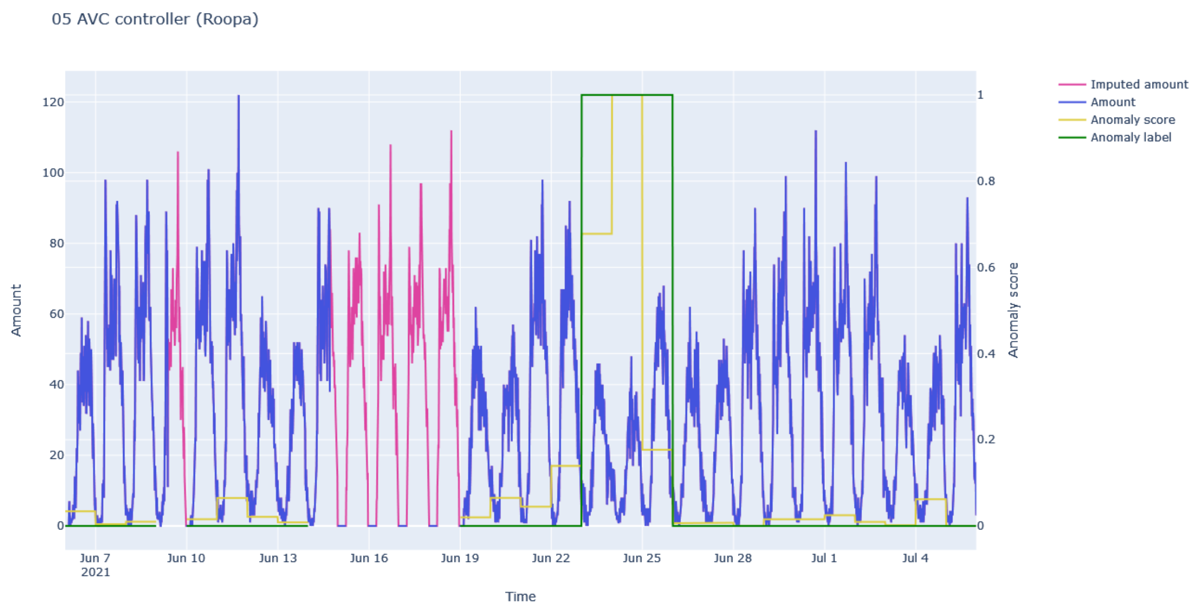


Figure 26: Big gap imputation. Notice how the imputations on the 16th, 17th and 18th of June are not affected by the anomalies of the 23rd, 24th and 25th of June.

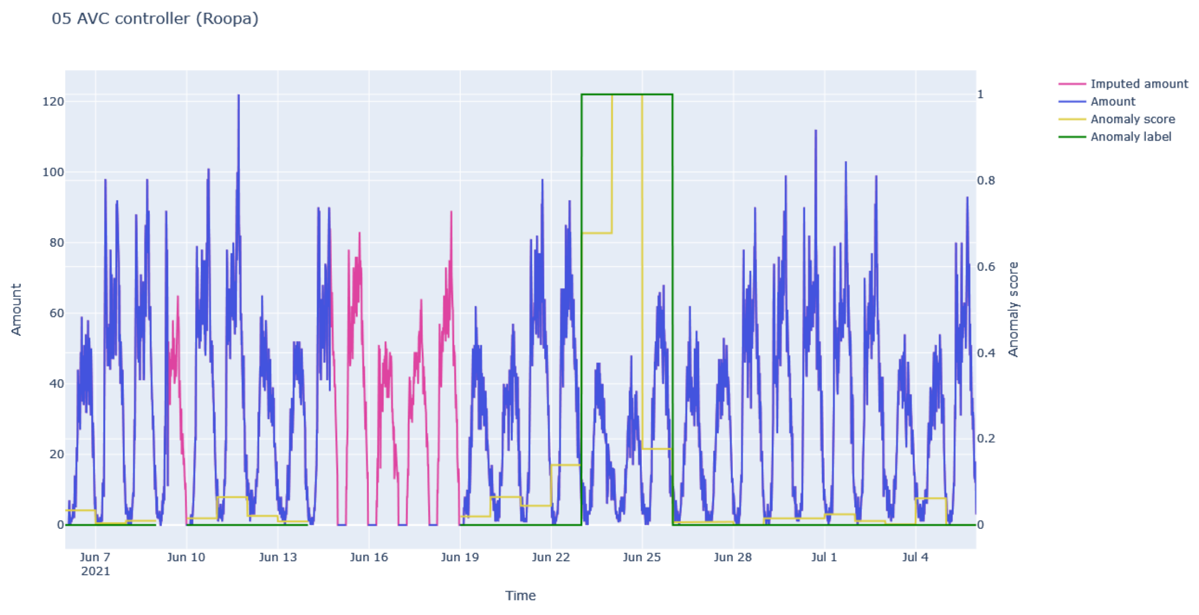


Figure 27: Big gap imputation if we would not exclude anomalous days.

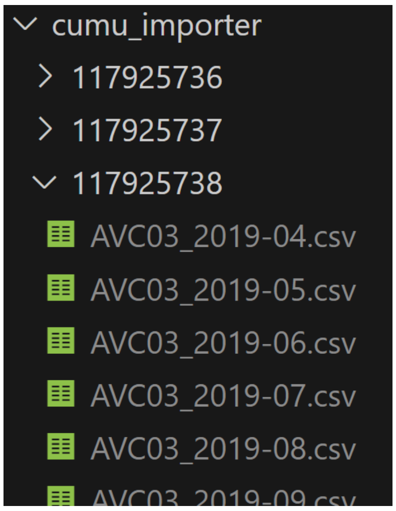
4.5 Candidate imputation solutions

We evaluated the results of our imputation solution against four other solutions from existing frameworks or literature: LATC, LCR and LCR-2d from Chen et al. (19,20) and MICE through the IterativeImputer class from the scikit-learn library (26). The methods under test accept data in different layouts. LCR expects a one-dimensional time series array like our seasonal interpolation solution. For LCR-2d and MICE we fold the time series into an array of 96 data points, forming an array of days. LATC was created to impute a set of spatially neighboring sensors observing different sections of the same road. It thus accepts time series in the form of sensor x day x time of day (19). The sensors we work with do not have this relationship. We think, however, that because of weekly patterns existing in our traffic data you could consider each week as a separate time series. Thus, we fold our time series for LATC by week in addition to by day creating a tensor of week x day x time of day.

5. Implementation

The created AD and imputation service is a Python script (Figure 30). We use PyOD, a Python outlier detection library for the LOF implementation in our AD solution (27). For imputation we are using the Statsmodels' time series analysis package (28). Data processing is done with NumPy (29) and Pandas (30), the number and tabular data processing libraries that we had the most previous experience in. We are communicating with Cumulocity via the c8y_api package, the official API wrapper for Python provided by Software AG (31). The source code can be accessed from Appendix I: Code repository. The full output of the service can be seen in Appendix III: AD and imputation service's output.

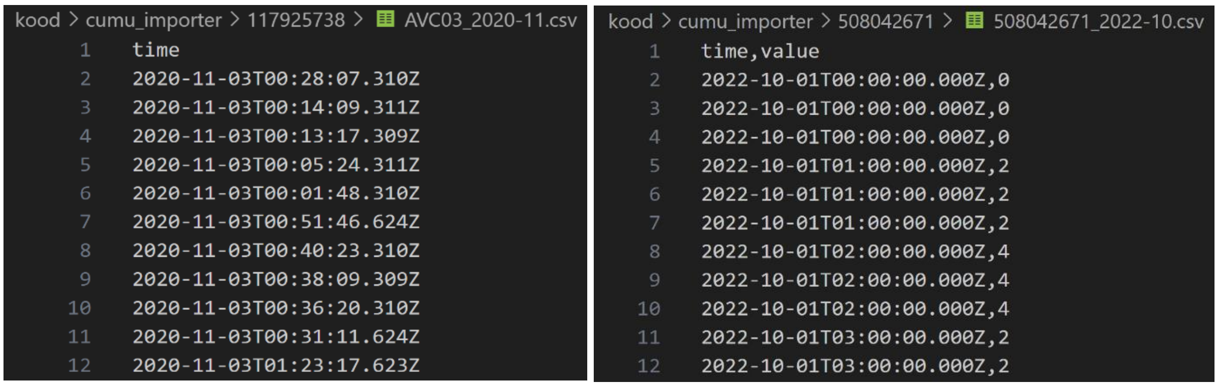
The service expects a directory of folders of CSV files containing time series or events. Each CSV folder should be named after the Cumulocity source device's id. The names of the files must end with a .csv suffix. (Figure 28)



```
cumu_importer
  117925736
  117925737
  117925738
    AVC03_2019-04.csv
    AVC03_2019-05.csv
    AVC03_2019-06.csv
    AVC03_2019-07.csv
    AVC03_2019-08.csv
    AVC03_2019-09.csv
```

Figure 28: An example directory layout for sensor data CSVs

For files containing event data a single column named "time" is expected. For measurement data columns "time" and "value" must be included (Figure 29).



```
kood > cumu_importer > 117925738 > AVC03_2020-11.csv
time
2020-11-03T00:28:07.310Z
2020-11-03T00:14:09.311Z
2020-11-03T00:13:17.309Z
2020-11-03T00:05:24.311Z
2020-11-03T00:01:48.310Z
2020-11-03T00:51:46.624Z
2020-11-03T00:40:23.310Z
2020-11-03T00:38:09.309Z
2020-11-03T00:36:20.310Z
2020-11-03T00:31:11.624Z
2020-11-03T01:23:17.623Z
```

```
kood > cumu_importer > 508042671 > 508042671_2022-10.csv
time,value
2022-10-01T00:00:00.000Z,0
2022-10-01T00:00:00.000Z,0
2022-10-01T00:00:00.000Z,0
2022-10-01T01:00:00.000Z,2
2022-10-01T01:00:00.000Z,2
2022-10-01T01:00:00.000Z,2
2022-10-01T02:00:00.000Z,4
2022-10-01T02:00:00.000Z,4
2022-10-01T02:00:00.000Z,4
2022-10-01T03:00:00.000Z,2
2022-10-01T03:00:00.000Z,2
```

Figure 29: Example files for event and measurement data files

With a sufficiently large time range data download can take a lot of time. Every download for 6 months' worth of a sensor's data ranged from 5 to 25 minutes depending on the popularity of the observed road and possible data gaps. By that we estimate that the entire data set download for 16 sensors in the date range of 2019-04-29 to 2023-06-01 took around 32 hours. The Python library (22) we used to download the AVC controllers' data served as a straightforward solution, but it failed to obtain the entire roughly 2GB dataset without any runtime errors. The extra step of proper error handling is why we decided to leave file downloading to the user.

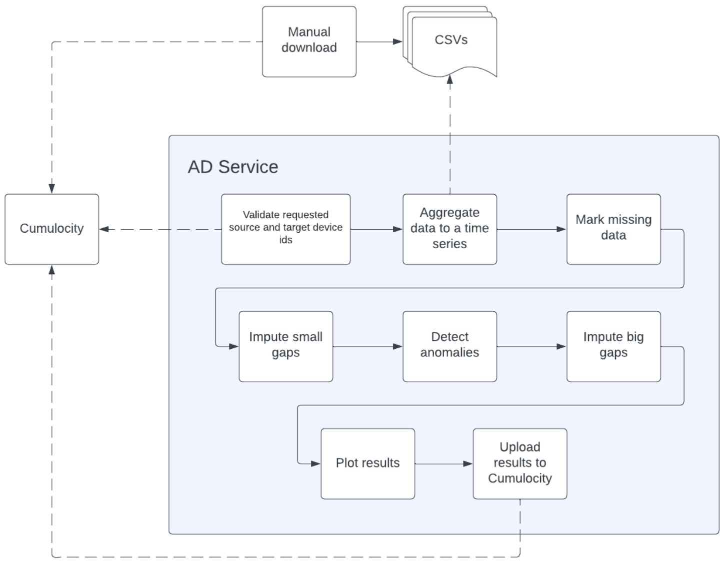
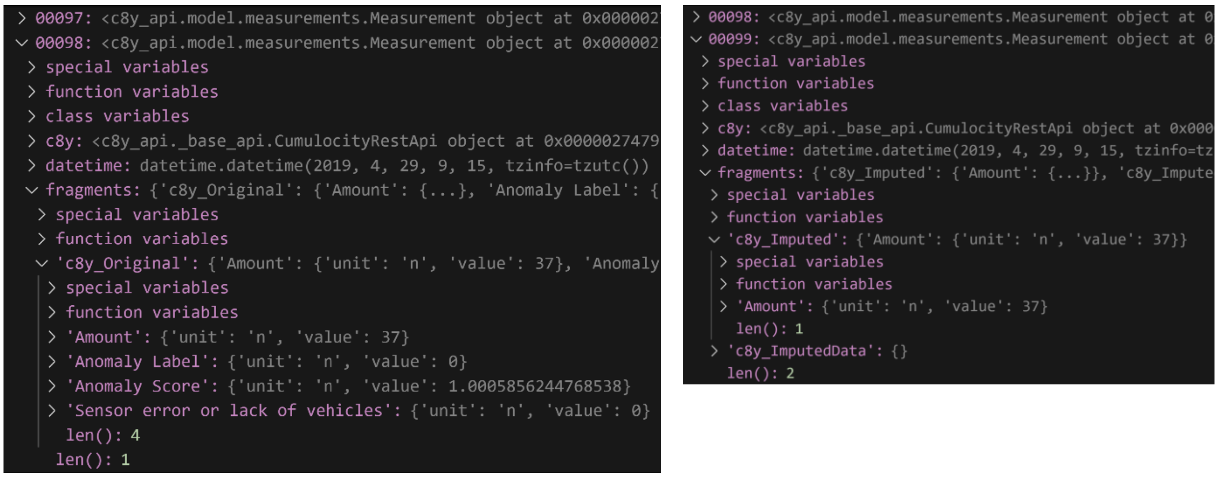


Figure 30: Processes behind the AD and imputation service

As an additional input the service also expects pairs of source and target Cumulocity device identifiers. Source device id is used to read in the correct CSV files, target id is used as the result upload destination.

If the provided CSV files contain events, then they are aggregated into time series with a frequency of 15 minutes. Next any days with missing data gaps are labeled missing, both for displaying data missingness to the end user but also for finding a healthy data set for anomaly detection since it only accepts days without missing values. We impute 15 minutes to 1 hour data gaps with seasonal decomposition to leverage the neighboring data points and to makes more days accessible to be used in anomaly detection. Then we detect and flag anomalies in the data set with the LOF implementation from PyOD. Next, we cover all the remaining data gaps with a mixture of daily and weekly linear interpolation.

Finally, data is uploaded to Cumulocity in the shape of Measurements (Figure 31). For each datapoint we create two Measurement objects. The first carries four datapoints: the original data unchanged, the anomaly score, the anomaly label and the data gap flag. The other Measurement contains the original data with the gaps filled with imputations and also a marker property.



```
> 00097: <c8y_api.model.measurements.Measurement object at 0x000002
v 00098: <c8y_api.model.measurements.Measurement object at 0x000002
  > special variables
  > function variables
  > class variables
  > c8y: <c8y_api._base_api.CumulocityRestApi object at 0x0000027479
  > datetime: datetime.datetime(2019, 4, 29, 9, 15, tzinfo=tzutc())
  v fragments: {'c8y_Original': {'Amount': {...}, 'Anomaly Label': {
    > special variables
    > function variables
    v 'c8y_Original': {'Amount': {'unit': 'n', 'value': 37}, 'Anomaly
      > special variables
      > function variables
      > 'Amount': {'unit': 'n', 'value': 37}
      > 'Anomaly Label': {'unit': 'n', 'value': 0}
      > 'Anomaly Score': {'unit': 'n', 'value': 1.0005856244768538}
      > 'Sensor error or lack of vehicles': {'unit': 'n', 'value': 0}
        len(): 4
      len(): 1
```

```
> 00098: <c8y_api.model.measurements.Measurement object at 0
v 00099: <c8y_api.model.measurements.Measurement object at 0
  > special variables
  > function variables
  > class variables
  > c8y: <c8y_api._base_api.CumulocityRestApi object at 0x000
  > datetime: datetime.datetime(2019, 4, 29, 9, 15, tzinfo=tz
  v fragments: {'c8y_Imputed': {'Amount': {...}}, 'c8y_Impute
    > special variables
    > function variables
    v 'c8y_Imputed': {'Amount': {'unit': 'n', 'value': 37}}
      > special variables
      > function variables
      > 'Amount': {'unit': 'n', 'value': 37}
        len(): 1
    > 'c8y_ImputedData': {}
      len(): 2
```

Figure 31: Examples of data uploaded to Cumulocity.

In total our solution communicates with Cumulocity on two fronts. We first validate via the Inventory API that the provided source and target devices exist in Cumulocity. Later we use the Measurement API to create a collection of real and imputed data points.

6. Evaluation of results

Here we are estimating the quality of the AD and imputation algorithms' output. The full overview of the results of the chosen solution can be seen in the Appendix III: AD and imputation service's output.

6.1 Anomaly detection results

We cannot say for a fact when a day of traffic data should be considered normal, but for a few select days we do know when it should be seen as anomalous. The data on holidays during a workday is very similar to the data on weekends. By relying on that notion, we can measure whether the AD solution was able to correctly label each holiday on a workday as an anomaly. We will also expect shortened workdays to be classified as anomalies. In four days in the Estonian calendar a workday is shortened by three hours. This means that the second rush hour should also happen three hours before it usually does and thus the day should be visible to our AD solution.

6.1.1 Candidate anomaly detection solutions

The results of the candidate algorithms applied via TimeEval did not bare any meaningful results. The best results were given by GrammarViz with a weekly window as seen from Figure 32. This is not, however, good enough since the results are spread over multiple days and thus difficult to interpret because of the large window size.

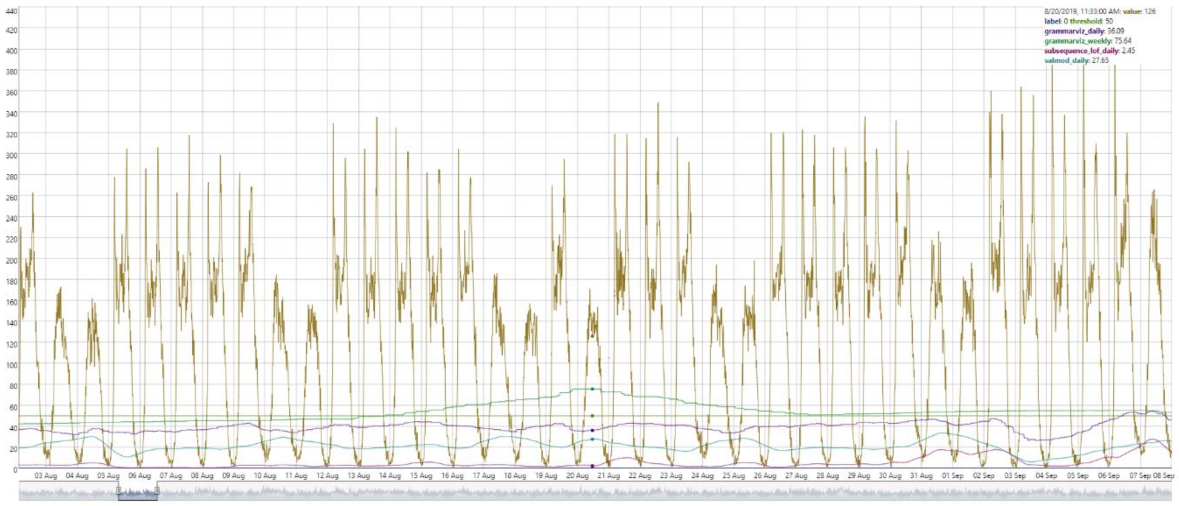


Figure 32: Excerpt from applying several different TimeEval algorithms to the traffic data. Notice how weekly GrammarViz is able to detect that the 20th of August is anomalous, but because of the large window size the result is spread over multiple days.

Sub-LOF, although marked as a top performer by Wenig et al. (11) gave no indication on being able to discern regular days from anomalous ones. This is supported by Keogh et al. who say that clusters received by clustering a set of time series extracted by a sliding window are essentially random (33).

6.1.2 Chosen anomaly detection solution

In contrast to the poor performance of the tested TimeEval algorithms the tumbled LOF approach gave us much more significant results for many sensors' data. Nine out of sixteen sensors had 90% or more of their holidays and shortened workdays correctly labeled (Table 2).

Table 2: Mean of the anomaly labels provided by the AD solution

Sensor	Mean AD label on holidays and shortened workdays
06 AVC	1.00
07 AVC	1.00
09 AVC	1.00
12 AVC	1.00
13 AVC	1.00
10 AVC	0.97
02 AVC	0.96
11 AVC	0.95
16 AVC	0.94
04 AVC	0.89
03 AVC	0.81
05 AVC	0.72
15 AVC	0.18
08 AVC	0.14
01 AVC	0.07
14 AVC	0.05

The data of the last four sensors performs severely under expectations. For 08, 14 and 15 AVC controllers this could be attributed to their overall poor data quality. The gathered data often rapidly decreases and dies off during summer times and then returns to normal behavior in autumn (Figure 33). These periods also grab the attention of the AD solution, leaving the holidays unlabeled.

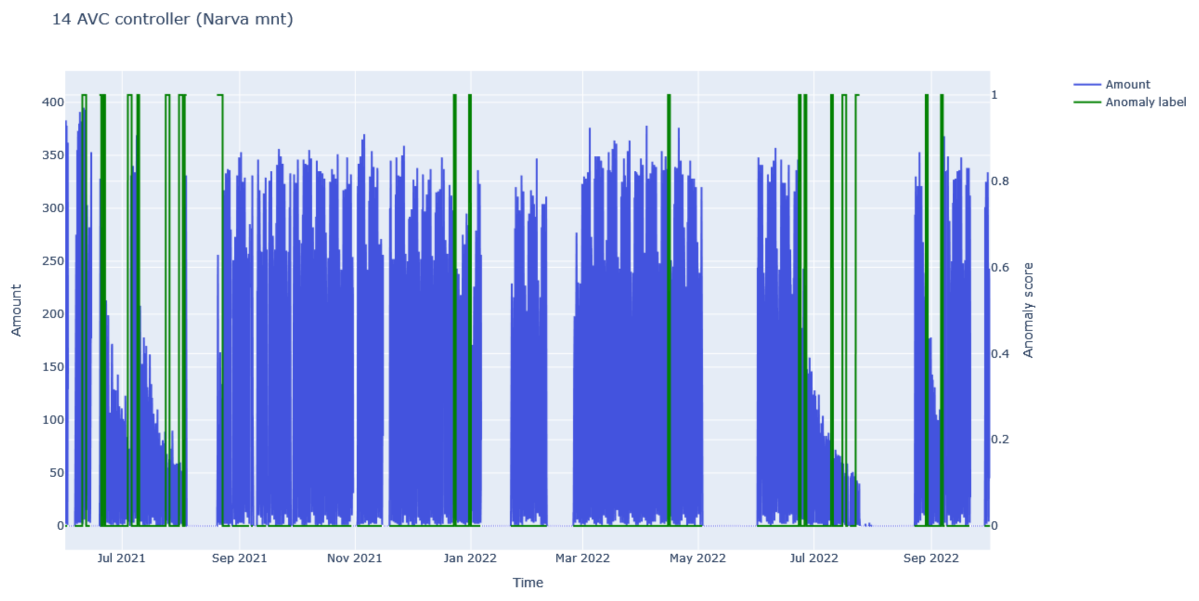


Figure 33: Excerpt from the data of the 14 AVC controller. Note how during summer the traffic count sharply diminishes.

In addition to holidays there were a couple of events in Tartu big enough that their impact could be seen from the traffic data. The Metallica concert took place in 2019-07-18 and is apparent in the nightly data of the next day in the 06, 07 and 09 AVC controllers (Figure 34).

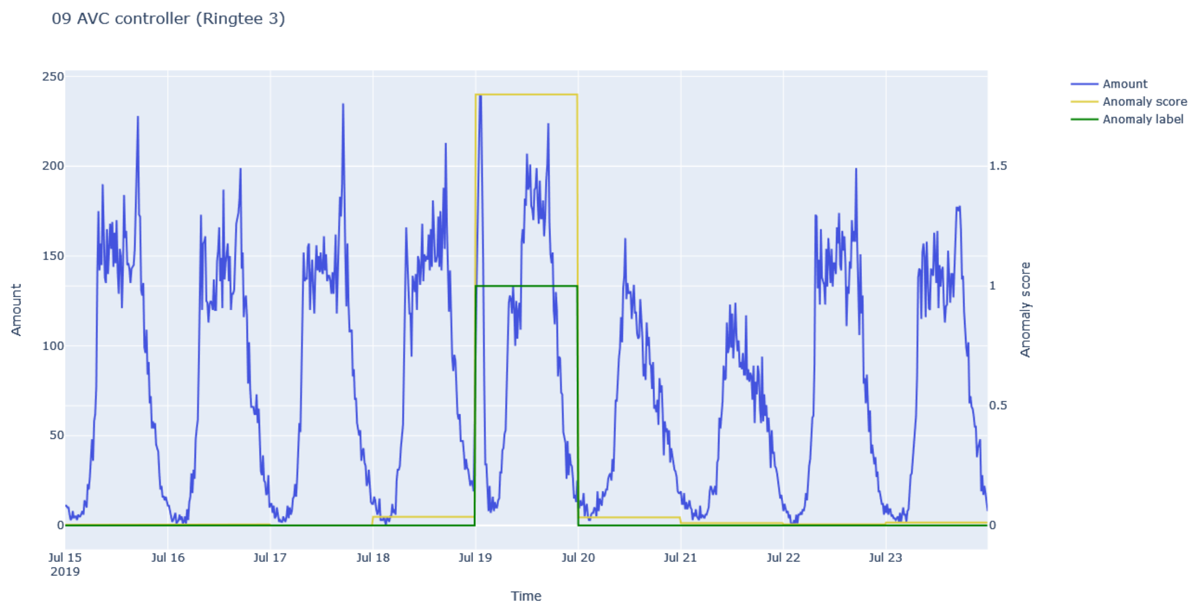


Figure 34: AD during the week when Metallica performed. The concert took place in the 18th of July, but data becomes anomalous on the night of the next day.

Tartu Rattaralli is a yearly bike competition whose influence can be seen in the 15 AVC controller data as the traffic was temporarily halted (Figure 35). The day of the competition got a high anomaly score compared to the neighboring days, but the temporality dimension between days is not taken into account and from a global standpoint the score was not big enough to be considered an anomaly.

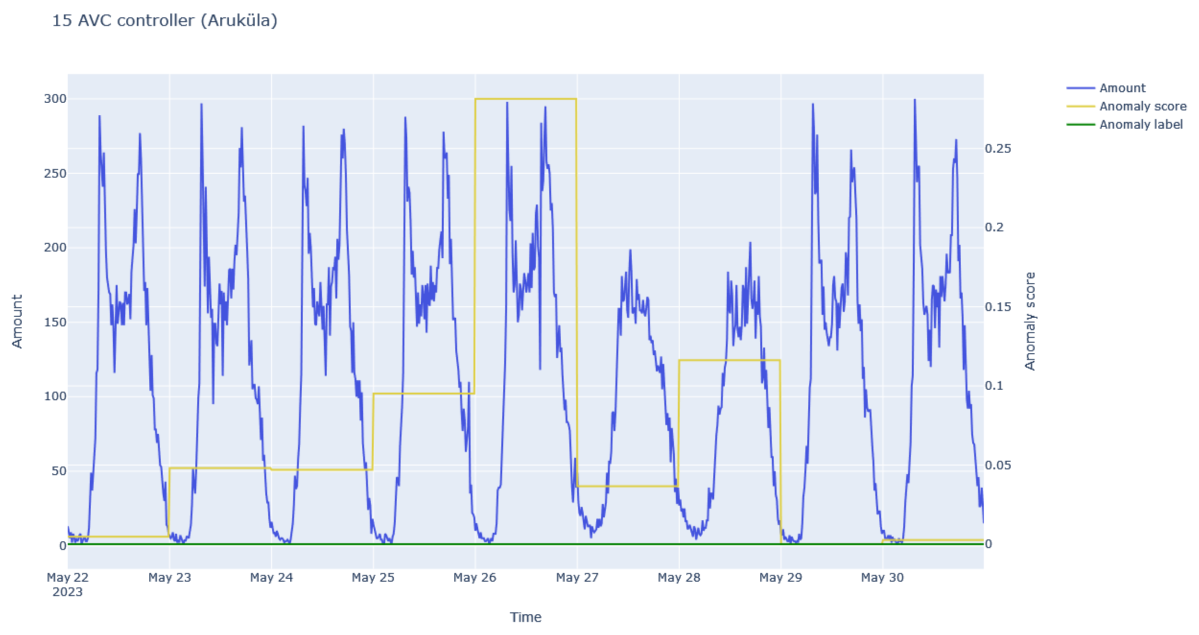


Figure 35: AD on the weekend of Tartu Rattaralli. Notice the rise in the anomaly score on the 26th of May.

6.2 Imputation results

The city of Tartu has two main use cases for imputation: filling the historical dataset and daily imputation. We mirrored these use cases in our evaluation by first measuring the imputation results on the entire and then on a small dataset. The underlying mechanic of this evaluation is to systematically remove a portion of the known data, allow an imputation method to fill the gap and then measure the error between the real and imputed datapoints. Each individual test is carried out 10 times with the same random seed to guarantee equal missingness of data between the tests. Root-mean-square error (RMSE) is used as our error metric to assess the effectiveness of an approach.

6.2.1 Historical dataset imputation

We evaluated each algorithm under 7, 30 and 90 randomly missing days. The time span we used is from 2019-04-19 to 2023-06-01. The best performers are interpolation, LATC and LCR-2d (Table 3).

Table 3 Average RMSE of historical data imputation solutions

	01AVC	**02AVC**	**03AVC**	**04AVC**	**05AVC**	**06AVC**	**07AVC**	**08AVC**
Interpolation	18.49	31.57	15.18	15.37	9.97	22.74	15.74	22.42
LATC	18.44	28.94	15.29	15.49	10.83	26.00	15.47	18.63
LCR	59.43	93.48	59.46	68.49	33.08	114.78	63.63	59.99
LCR-2d	21.30	31.57	16.79	16.40	9.61	24.67	16.16	18.57
MICE	38.92	40.83	29.67	29.70	17.52	46.43	30.08	38.89
	09AVC	**10AVC**	**11AVC**	**12AVC**	**13AVC**	**14AVC**	**15AVC**	**16AVC**
Interpolation	15.71	8.91	11.29	21.23	9.42	30.61	23.02	8.44
LATC	16.32	11.06	11.85	20.87	12.32	24.19	19.40	7.38
LCR	62.44	25.41	43.47	99.85	30.46	115.00	80.17	20.90
LCR-2d	15.88	11.22	12.09	20.92	9.72	26.52	20.83	8.05
MICE	28.51	11.49	24.48	36.81	14.12	53.04	41.37	10.60

There is a high amount of missingness in many sensors' data. Since some of the imputation methods we test may be more sensitive to the quality of the data than others we decided to also run the tests against a historical dataset where data gaps have been first filled with interpolated data before introducing systematic data gaps (Table 4).

Table 4 Average RMSE of historical data imputation solutions on imputed data

	01AVC	**02AVC**	**03AVC**	**04AVC**	**05AVC**	**06AVC**	**07AVC**	**08AVC**
Interpolation	10.22	23.14	7.35	9.52	8.30	16.93	11.84	9.54
LATC	17.51	31.90	13.20	17.26	14.42	29.53	21.05	16.29
LCR	19.98	45.88	18.57	15.91	12.46	36.45	22.58	22.07
LCR-2d	17.99	33.86	19.95	14.07	11.61	25.65	19.07	18.48
MICE	21.78	34.95	24.45	19.41	15.67	39.71	27.03	28.81
	09AVC	**10AVC**	**11AVC**	**12AVC**	**13AVC**	**14AVC**	**15AVC**	**16AVC**
Interpolation	11.43	5.08	9.11	15.00	4.21	16.29	14.86	6.72
LATC	19.14	9.72	13.64	24.78	10.46	29.91	29.65	10.36
LCR	18.93	10.27	13.90	27.94	10.59	32.55	27.31	7.82
LCR-2d	16.27	9.36	11.72	21.90	9.43	24.58	20.26	8.10
MICE	23.66	10.76	16.53	30.44	12.59	33.46	31.93	9.36

There is a severe improvement in quality for the LCR algorithm indicating that the quality of data that the highway sensors have collected may not be good enough for more sophisticated imputation techniques. The quality of the LATC and LCR-2d, however, did not change that much.

6.2.2 Daily imputation

Measuring the results under the daily imputation use case is split into two variations. We are interested in how the algorithms perform both with only historical data and a combination of historical and future data. This helps to later decide when and in which time intervals to run the selected imputation method for maximum accuracy.

For historical evaluation we look at random one-, two- and four-week sequences and use our imputation solution to predict the datapoints of the following day (Table 5).

Table 5 Average RMSE of daily imputation solutions on historic data

	01AVC	**02AVC**	**03AVC**	**04AVC**	**05AVC**	**06AVC**	**07AVC**	**08AVC**
Interpolation	9.46	29.19	9.17	10.95	6.82	18.99	13.07	11.72
LATC	13.93	54.36	10.77	16.90	10.60	24.00	18.29	18.89
LCR	24.56	54.59	26.39	21.48	17.18	44.38	29.12	29.39
LCR-2d	24.40	42.04	28.38	20.74	16.22	40.66	27.78	30.31
MICE	23.80	37.78	23.84	20.03	15.46	36.66	24.52	28.92
	09AVC	**10AVC**	**11AVC**	**12AVC**	**13AVC**	**14AVC**	**15AVC**	**16AVC**
Interpolation	12.71	6.80	9.49	18.23	4.78	25.89	18.25	7.41
LATC	15.14	8.26	10.59	23.41	7.68	23.51	22.34	9.68
LCR	24.07	14.14	18.13	36.57	14.13	41.73	34.40	10.28
LCR-2d	22.37	12.86	17.00	33.81	13.16	41.12	34.82	10.25

MICE	20.28	11.22	15.49	30.33	11.77	37.68	33.82	9.82

For a combination of historical and future data we look at random two-, four- and eight-week sequences and impute the datapoints of the middle day (Table 6).

Table 6 Average RMSE of daily imputation solutions on historic and future data

	01AVC	02AVC	03AVC	04AVC	05AVC	06AVC	07AVC	08AVC
Interpolation	10.22	23.14	7.35	9.52	8.30	16.93	11.84	9.54
LATC	17.51	31.90	13.20	17.26	14.42	29.53	21.05	16.29
LCR	19.98	45.88	18.57	15.91	12.46	36.45	22.58	22.07
LCR-2d	17.99	33.86	19.95	14.07	11.61	25.65	19.07	18.48
MICE	21.78	34.95	24.45	19.41	15.67	39.71	27.03	28.81
	09AVC	10AVC	11AVC	12AVC	13AVC	14AVC	15AVC	16AVC
Interpolation	11.43	5.08	9.11	15.00	4.21	16.29	14.86	6.72
LATC	19.14	9.72	13.64	24.78	10.46	29.91	29.65	10.36
LCR	18.93	10.27	13.90	27.94	10.59	32.55	27.31	7.82
LCR-2d	16.27	9.36	11.72	21.90	9.43	24.58	20.26	8.10
MICE	23.66	10.76	16.53	30.44	12.59	33.46	31.93	9.36

Interpolation solution is again the best performing out of all the methods. Interpolation, LCR and LCR-2d have noticeable benefits from being run with both historic and future data.

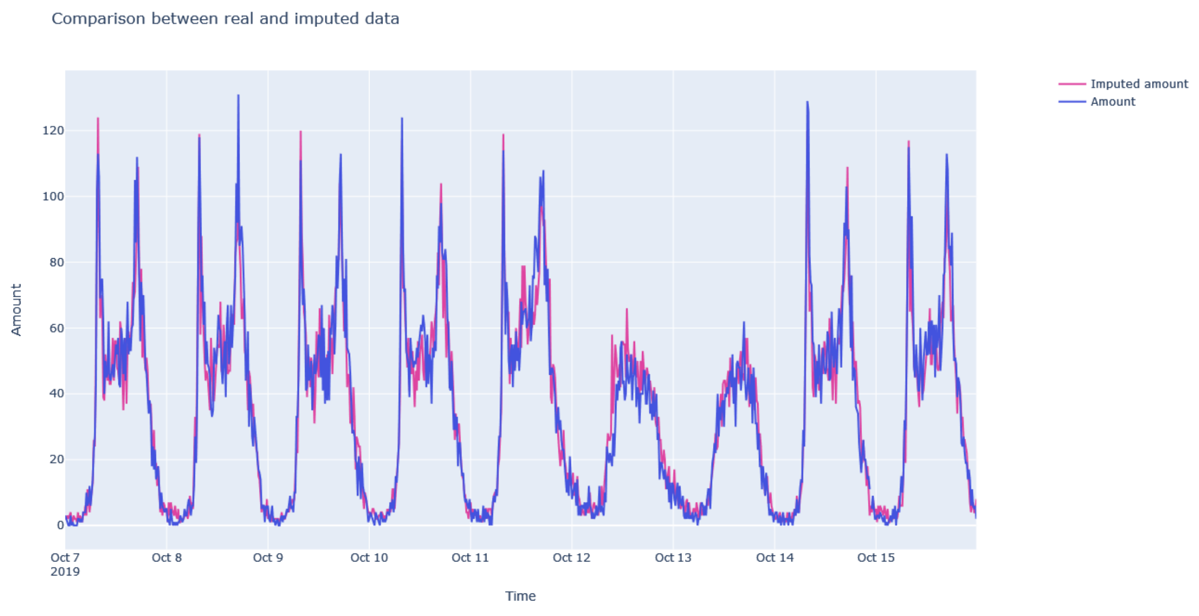


Figure 36: An example of some imputed days from testing the interpolation solution on the AVC05 sensor

6.3 Evaluation conclusion

We tried out multiple different solutions both for AD and imputations. For anomaly detection we found that relying on a strong seasonality and using days as datapoints gives much better results than searching for anomalies from time series sequences generated by a sliding window. We admit, however, that our ad hoc process of evaluating AD solutions with no labeled data is probably not thorough enough to give any conclusive evidence.

Regarding imputation we established that a simple interpolation approach using daily and weekly dependencies given our data gives more accurate results than the state-of-the-art solutions.

7. Extended use cases

The AD service was first created to process and output bulk data. That is the results were constructed in a single batch, not in an incremental fashion. It also made some assumptions regarding the underlying data, namely that the nighttime periods are the only non-anomalous time when data gaps can exist and that there is a clear distinction between workdays and weekends. Albeit these points it is possible to use the AD service with incremental data, like daily, weekly or monthly. It can also be used on other types of traffic, not only vehicles.

7.1 Anomaly detection model

After performing anomaly detection, the LOF models are stored in the local filesystem. This enables the AD solution to be used with smaller data sets. The intended use case is to detect anomalies first with a bulk data set and then continue onwards with for example daily or weekly data. Since trends and seasonalities can change over time and bring about model drift, the AD model should be retrained after some period.

7.1.1 Model lifespan

To measure how the results of the AD solution with a fixed model diminish over time we looked at three time periods with a dense number of holidays and trained models at an increasing lag in respect to these periods. The chosen periods were 2020-04-15 to 2020-07-08, 2021-06-15 to 2021-09-07 and 2022-12-15 to 2023-03-09. We based our models on the data of the five sensors with the greatest number of days usable in anomaly detection: the 06, 07, 09, 11 and 12 AVC controllers (Table 7).

Table 7: Sensors by the number of days usable in anomaly detection. The days qualify to be usable if their data gaps are no more than an hour long.

Sensor	Number of days usable in AD
06 AVC	1279
07 AVC	1274
09 AVC	1260
12 AVC	1243
11 AVC	1230
14 AVC	1205
15 AVC	1186
16 AVC	1124
05 AVC	1077
02 AVC	1068
04 AVC	977
08 AVC	944
01 AVC	826
03 AVC	803
10 AVC	738
13 AVC	724

We measured the models' performance over lags of 4, 8, 12, 24, 36 and 48 weeks (Figure 37). We used the same quality measurement solution as in subchapter 6.1 Anomaly Detection Results.

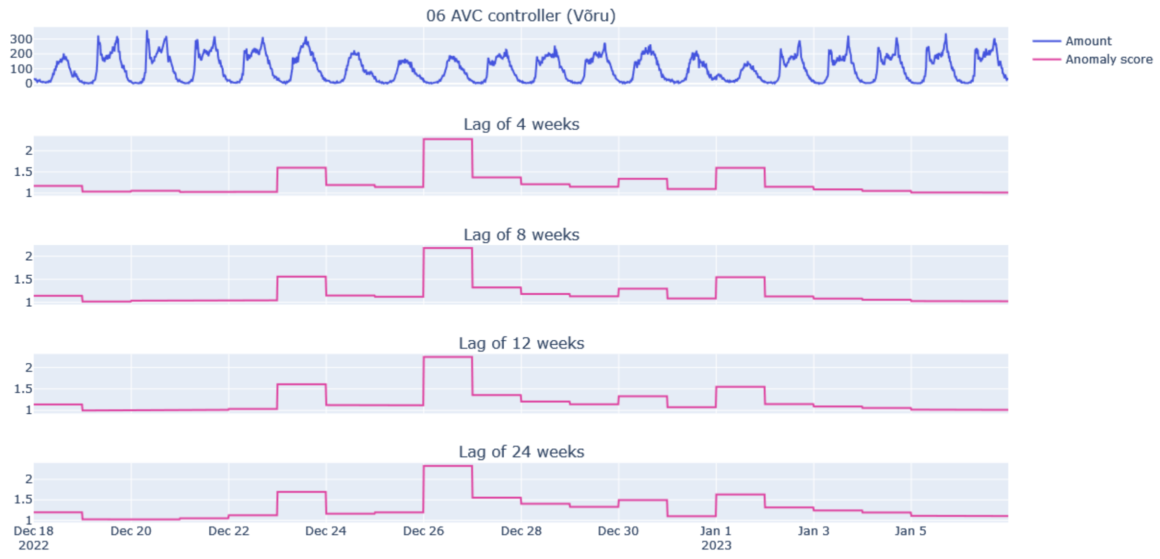


Figure 37: Excerpt from a comparison of models with different lags from the training time in respect to the validation period

After aggregating the results by lags we see that the outcome of the AD solution does not depend on the model's lag (Table 8). This means that in the timeframe of our data the trends and seasonality are quite constant and an anomalous day could be identified even with a dated model.

Table 8: Mean anomaly detection labels aggregated over sensors and periods in respect to lags.

Lag in weeks	**Mean AD label on holidays and shortened workdays**
4	0.98
8	0.98
12	0.98
24	1.00
36	0.98
48	0.97

The training data size was determined to be a fixed 48 weeks for every model, the origin of this number will be demonstrated in the next subchapter.

7.1.2 Model training size

We use a similar approach to evaluate the optimal length of training data the model needs to be able to differentiate between normal and anomalous days. In the previous example the variable we changed was the lag between the end of training time and the validation period. Now we vary the training size but fix the lag to be right before the validation period (Figure 38).

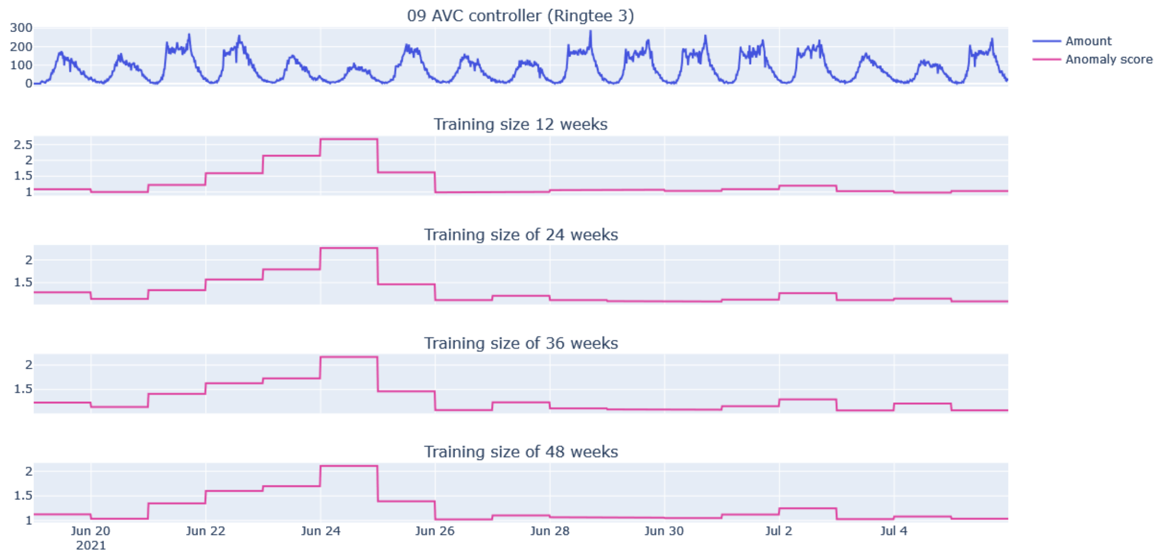


Figure 38: Excerpt from a comparison of models with different training times periods in respect to the validation period

After aggregating the results, we don't see a drop in quality for any particular training size (Table 9). These results probably suffer from the fact that holidays are not very nuanced anomalies to identify. With a labeled data set identifying more intricate anomalies than holidays these results would surely drop.

Table 9: Mean anomaly detection labels aggregated over sensors and periods in respect to training size.

Training size in weeks	Mean AD label on holidays and shortened workdays
12	0.92
24	0.90
36	0.98
48	0.93

7.2 Compatibility with other traffic sensors

The city of Tartu showed an interest in the AD and imputation service to also be able to handle data from other traffic sensors. In addition to the AVC controllers surrounding Tartu there are also inner-city sensors that track vehicles, bikes and pedestrians. We processed a large number of these sensors' data and report whether or not our AD service is able to accommodate for this different type of traffic data.

Some presumptions that we hold over the vehicle traffic data do not hold true for other traffic data. For one, nighttime is not the only period where it is normal for the traffic to stop. As seen in Figure 39 it is normal for there to be no cyclists during colder months. In the context of the AD and imputation service this means that the winter days will not be utilized in anomaly detection since AD clusters only data with small data gaps. In addition, the data gaps in winter data will be filled with imputed values, even though these gaps should be left empty.

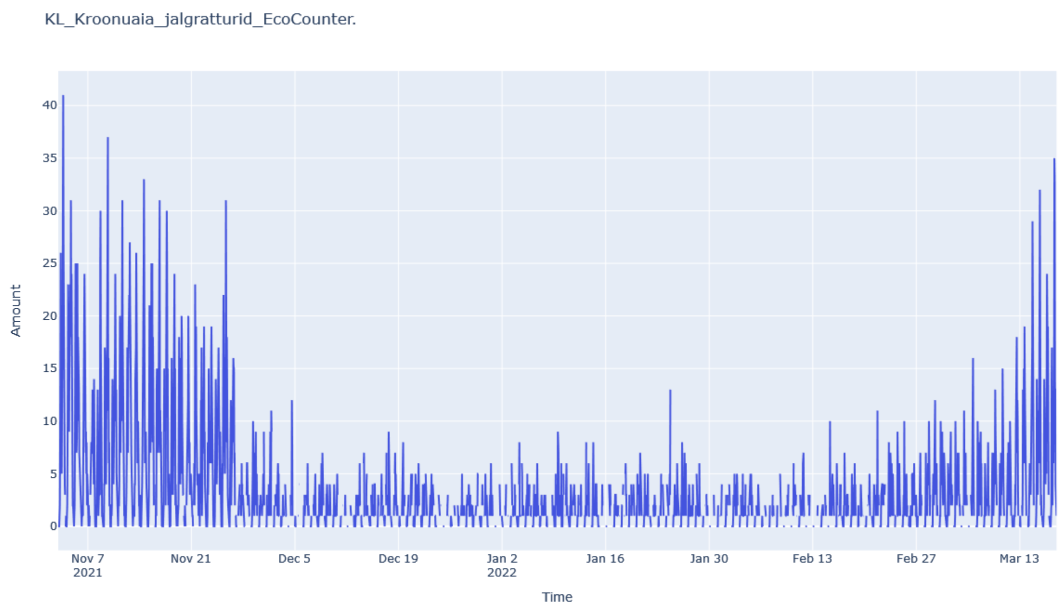


Figure 39: Demonstrating the low number of cyclists during winter months

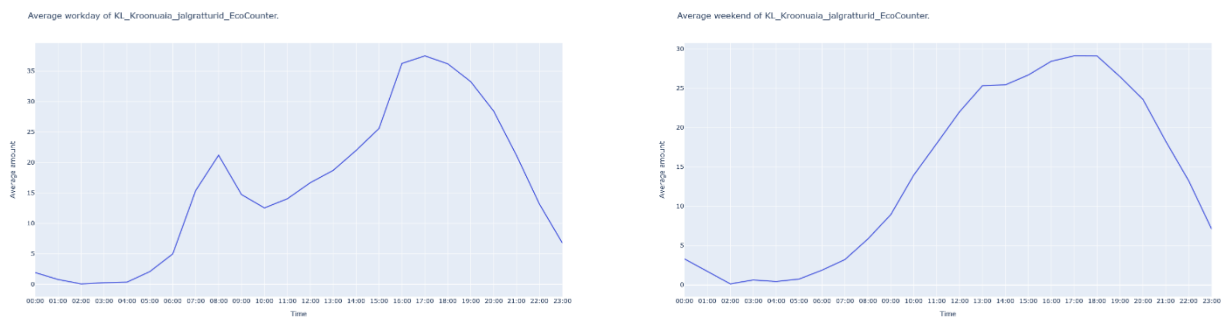


Figure 40: Average workday and weekend of the Kroonuaia cyclist' EcoCounter

We also assumed that traffic data always has a strong seasonality. It still holds true that there is a lot less traffic during night, but the morning and evening rush hours do not stand out as much as in vehicle data (Figure 40, 41).

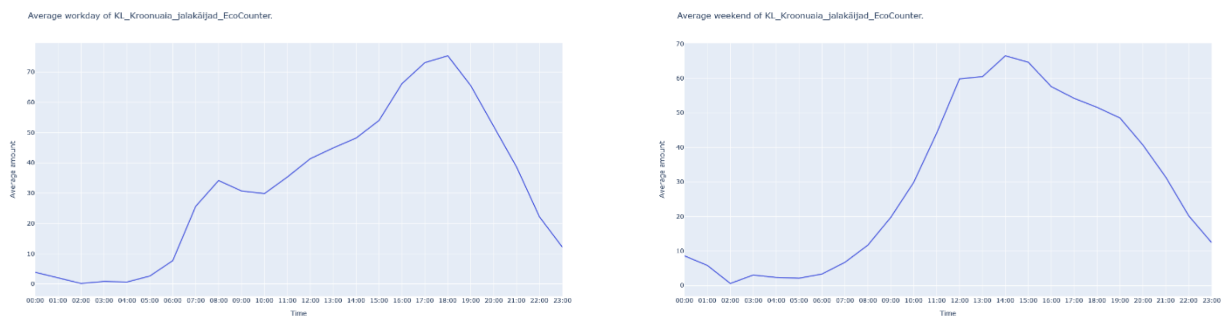


Figure 41: Average workday and weekend of the Kroonuaia pedestrian' EcoCounter

The results of anomaly detection in bicycle and pedestrian data are substantially worse than in vehicle data. The reason could be that holidays do not have that big of an impact on light traffic. The volumes of non-vehicle traffic are already quite small so changes due to special occasions may not prove to be significant enough for the AD solution to pick them up (Table 10, 11).

Table 10: Mean of the anomaly labels provided by the AD solution on the bicycle sensors

Sensor	**Mean AD label on holidays and shortened workdays**
ECO Counter (Riia-Lembitu) Bicycle.counter	1.00
ECO Counter (Riia - Raudtee) Bicycle counter	0.50
KL_Anne kiir 1_jalgratturid_kesklinna suunast_EcoCounter	0.36
KL_Väike kaar_jalgratturid_EcoCounter	0.36
KL_Turu sild_jalgratturid_out_EcoCounter	0.35
ECO Counter (Riia - Raudtee) BicycleOut.counter	0.33
KL_Raudtee-Kabeli_jalgratturid_EcoCounter	0.31
KL_Mõisavahe 1_jalgratturid_kesklinna suunas_EcoCounter	0.29
KL_Turu sild_jalgratturid_in_EcoCounter	0.29
KL_Mõisavahe 1_jalgratturid_kesklinna suunast_EcoCounter	0.27
KL_Mõisavahe 2_jalgratturid_kesklinna suunast_EcoCounter	0.27
ECO Counter (Riia-Lembitu) BicycleIn.counter	0.25
KL_Näituse_jalgratturid_EcoCounter	0.25
KL_Kroonuaia_jalgratturid_EcoCounter	0.21
KL_Mõisavahe 2_jalgratturid_kesklinna suunas_EcoCounter	0.18
ECO Counter (Riia - Raudtee) BicycleIn.counter	0.00
ECO Counter (Riia-Lembitu) BicycleOut.counter	0.00
KL_Anne kiir 1_jalgratturid_kesklinna suunas_EcoCounter	0.00

Table 11: Mean of the anomaly labels provided by the AD solution on the pedestrian sensors

Sensor	**Mean AD label on holidays and shortened workdays**
ECO Counter (Riia-Lembitu) PedestrianIn.counter	1.00
KL_Turu sild_jalakäijad_in_EcoCounter	0.92
KL_Turu sild_jalakäijad_out_EcoCounter	0.81
ECO Counter (Riia - Raudtee) PedestrianOut.counter	0.57
ECO Counter (Riia-Lembitu) PedestrianOut.counter	0.56
ECO Counter (Riia - Raudtee) PedestrianIn.counter	0.40
KL_Raudtee-Kabeli_jalakäijad_EcoCounter	0.38
KL_Mõisavahe 1_jalakäijad_kesklinna suunast_EcoCounter	0.33
KL_Mõisavahe 2_jalakäijad_kesklinna suunas_EcoCounter	0.32
ECO Counter (Anne_jalakäijad kesklinna suunas)	0.29
KL_Mõisavahe 1_jalakäijad_kesklinna suunas_EcoCounter	0.24

KL_Anne kiir 1_jalakäijad_kesklinna suunast_EcoCounter	0.17
KL_Mõisavahe 2_jalakäijad_kesklinna suunast_EcoCounter	0.17
KL_Kroonuaia_jalakäijad_EcoCounter	0.14
KL_Näituse_jalakäijad_EcoCounter	0.10
KL_Väike kaar_jalakäijad_EcoCounter	0.05

8. Conclusion and future directions

This thesis gave an overview on the data quality of vehicle traffic sensors of the city of Tartu. We measured different anomaly detection (AD) and imputation solutions to better the data and make it more accessible to further analysis. The best AD approach separates the traffic data time series into daily data points and then clusters them with the local outlier factor (LOF). Albeit the significant number and length of gaps, the data seems intact enough for most sensors that a distinction between normal and anomalous days can be made. For imputing the numerous data gaps, we divide the problem space into an imputation of small and big gaps. We use linear interpolation on the seasonally decomposed time series to fill small data gaps. For big gaps we combine the interpolations of daily and weekly split time series. We then described and created a service that uses the chosen anomaly detection and imputation solution on the Tartu traffic data.

The service outputs its findings to the Cumulocity data platform. The city of Tartu can then download and utilize this data instead of the original data. The partially synthetic data can be used in further analysis and evaluation. Today the data of all 16 highway traffic sensors in the time range of 2019-04-19 to 2023-06-01 has been processed and uploaded to Cumulocity.

We also looked at Tartu’s other traffic sensors, namely the pedestrian and cyclist counters. As there is an interest in a more general solution for sensor data quality improvements, we applied the anomaly detection and imputation service to the light traffic sensor data. The results were substantially worse, but this could also be attributed to our limited validation capabilities.

8.1 Future directions

The AD and imputation service serves its purpose, but would benefit from a number of enhancements. For one, the anomaly detector should take the temporal dependency between the days also into account and not just observe days in a vacuum. That way we could detect outliers that are fairly normal in a global sense, but anomalous when compared to their neighboring days. Also, after the initial bulk analysis on the historic data is done, next activations of the service should download the new data by itself, not needing a set of CSV files to be downloaded in prior.

Since validation is a difficult topic when working with unlabeled data it would be interesting to develop a synthetic dataset modeling the trends and seasonality of Tartu’s traffic. This dataset could be then used to better automate the development and testing of data quality improvement systems for the city of Tartu.

Finally, we could consider an online learning approach for the anomaly detection model, giving it the capability of continuously ingesting streaming data. This approach would circumvent the questions of model updating and training size.

References

1. IoT connected devices by vertical 2030 | Statista [Internet]. [cited 2023 Aug 11]. Available from: https://www.statista.com/statistics/1194682/iot-connected-devices-vertically/

2. Kavandatud tegevused – Tark Tartu [Internet]. [cited 2023 Aug 11]. Available from: https://tarktartu.ee/avaleht/kavandatud-tegevused/

3. Helping CIOs Understand "Smart City" Initiatives | Forrester [Internet]. [cited 2023 Aug 11]. Available from: https://www.forrester.com/report/Helping-CIOs-Understand-SmartCity-Initiatives/RES55590

4. Chen T. Smart grids, smart cities need better networks. IEEE Netw. 2010 Mar;24(2):2–3.

5. IoT Platform | Cumulocity IoT | Software AG [Internet]. [cited 2023 Aug 11]. Available from: https://www.softwareag.com/en_corporate/platform/iot/iot-analytics-platform.html

6. Lehtsalu Urmo, Kärner Toomas, Kallas Priit. Telia Eesti Internet of Things Report. 2017.

7. Cumulocity IoT's domain model - Cumulocity IoT Guides [Internet]. [cited 2024 Jan 3]. Available from: https://cumulocity.com/guides/concepts/domain-model/

8. Cook AA, Misirli G, Fan Z. Anomaly Detection for IoT Time-Series Data: A Survey. IEEE Internet Things J. 2020 Jul 1;7(7):6481–94.

9. Al-Amri R, Murugesan RK, Man M, Abdulateef AF, Al-Sharafi MA, Alkahtani AA. A review of machine learning and deep learning techniques for anomaly detection in iot data. Applied Sciences (Switzerland). 2021 Jun 2;11(12).

10. Fahim M, Sillitti A. Anomaly Detection, Analysis and Prediction Techniques in IoT Environment: A Systematic Literature Review. IEEE Access. 2019;7:81664–81.

11. Wenig Phillip.Wenig@Hpi.De P, Schmidl Sebastian.Schmidl@Hpi.De S, Papenbrock T. TimeEval: A Benchmarking Toolkit for Time Series Anomaly Detection Algorithms. Proceedings of the VLDB Endowment. 2022;15(12):3678–81.

12. Benchmark result analysis | Anomaly Detection in Time Series: A Comprehensive Evaluation [Internet]. [cited 2024 Jan 3]. Available from: https://timeeval.github.io/evaluation-paper/notebooks/Benchmark-result-analysis.html

13. Breunig MM, Kriegel HP, Ng RT, Sander J. LOF: Identifying Density-Based Local Outliers.

14. Senin P, Lin J, Wang X, Oates T, Gandhi S, Boedihardjo AP, et al. Time series anomaly discovery with grammar-based compression. EDBT 2015 - 18th International Conference on Extending Database Technology, Proceedings. 2015;481–92.

15. Linardi M, Zhu Y, Palpanas T, Keogh E. Matrix profile goes MAD: variable-length motif and discord discovery in data series. Data Min Knowl Discov. 2020 Jul 1;34(4):1022–71.

16. Li Y, Li Z, Li L. Missing traffic data: comparison of imputation methods; Missing traffic data: comparison of imputation methods. Available from: https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/iet-its.2013.0052

17. Chapter 8 ARIMA models | Forecasting: Principles and Practice (2nd ed) [Internet]. [cited 2024 Jan 1]. Available from: https://otexts.com/fpp2/arima.html

18. Du W. Pypots: a python toolbox for data mining on partially-observed time series a preprint [Internet]. 2023. Available from: https://coveralls.io/

19. Chen X, Lei M, Saunier N, Sun L. Low-Rank Autoregressive Tensor Completion for Spatiotemporal Traffic Data Imputation. 2021 Apr 30; Available from: http://arxiv.org/abs/2104.14936

20. Chen X, Cheng Z, Saunier N, Sun L, Member S. Laplacian Convolutional Representation for Traffic Time Series Imputation.

21. Azur MJ, Stuart EA, Frangakis C, Leaf PJ. Multiple imputation by chained equations: what is it and how does it work? 2011;

22. Laius Silver. Integratsiooniteek Cumulocity IoT platvormi ajalooliste andmete analüüsiks Pythonis. 2019.

23. Moritz S, Bartz-Beielstein T. imputeTS: Time series missing value imputation in R. R Journal. 2017 Jun 1;9(1):207–18.

24. Zhong M, Sharma S, Liu Z. Assessing Robustness of Imputation Models Based on Data from Different Jurisdictions. Transportation Research Record: Journal of the Transportation Research Board. 2005 Jan;1917(1):116–26.

25. Hyndman RJ, Athanasopoulos G. 6.3 Classical decomposition | Forecasting: Principles and Practice (2nd ed) [Internet]. 2018 [cited 2023 Aug 11]. Available from: https://otexts.com/fpp2/classical-decomposition.html

26. Pedregosa FABIANPEDREGOSA F, Michel V, Grisel OLIVIERGRISEL O, Blondel M, Prettenhofer P, Weiss R, et al. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research [Internet]. 2011 [cited 2024 Jan 1];12(85):2825–30. Available from: http://jmlr.org/papers/v12/pedregosa11a.html

27. Zhao Y, Nasrullah Z, Li Z. PyOD: A python toolbox for scalable outlier detection. Journal of Machine Learning Research. 2019 May 1;20.

28. Seabold S, Perktold J. Statsmodels: Econometric and Statistical Modeling with Python. PROC OF THE 9th PYTHON IN SCIENCE CONF [Internet]. 2010 [cited 2023 Aug 11]; Available from: http://statsmodels.sourceforge.net/

29. Harris CR, Millman KJ, van der Walt SJ, Gommers R, Virtanen P, Cournapeau D, et al. Array programming with NumPy. Nature 2020 585:7825 [Internet]. 2020 Sep 16

[cited 2023 Aug 11];585(7825):357–62. Available from: https://www.nature.com/articles/s41586-020-2649-2

30. Mckinney W. Data Structures for Statistical Computing in Python. In 2010.

31. Souris Christoph. c8y-api · PyPI [Internet]. [cited 2023 Aug 11]. Available from: https://pypi.org/project/c8y-api/

32. Glossary - Cumulocity IoT Guides [Internet]. [cited 2023 Aug 11]. Available from: https://cumulocity.com/guides/concepts/glossary/#measurement

33. Keogh E, Lin J. Clustering of time-series subsequences is meaningless: Implications for previous and future research. Knowl Inf Syst. 2005;8(2):154–77.

34. Tartu Rattaralli 2023 liiklusinfo [Internet]. [cited 2023 Aug 11]. Available from: https://tartumaraton.ee/et/tartu-rattaralli-2023/liiklusinfo

Appendices

Appendix I: Code repository

The created AD and imputation service can be found here: https://github.com/Joonaspraks/tartu-traffic-ad-service

Appendix II: Data gaps by duration

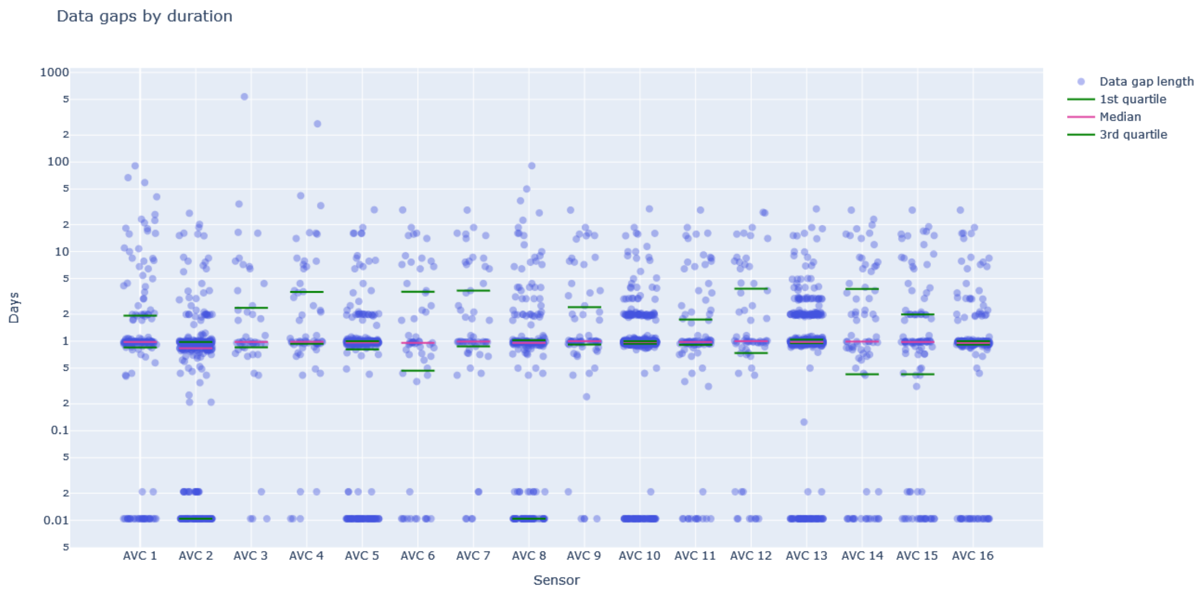
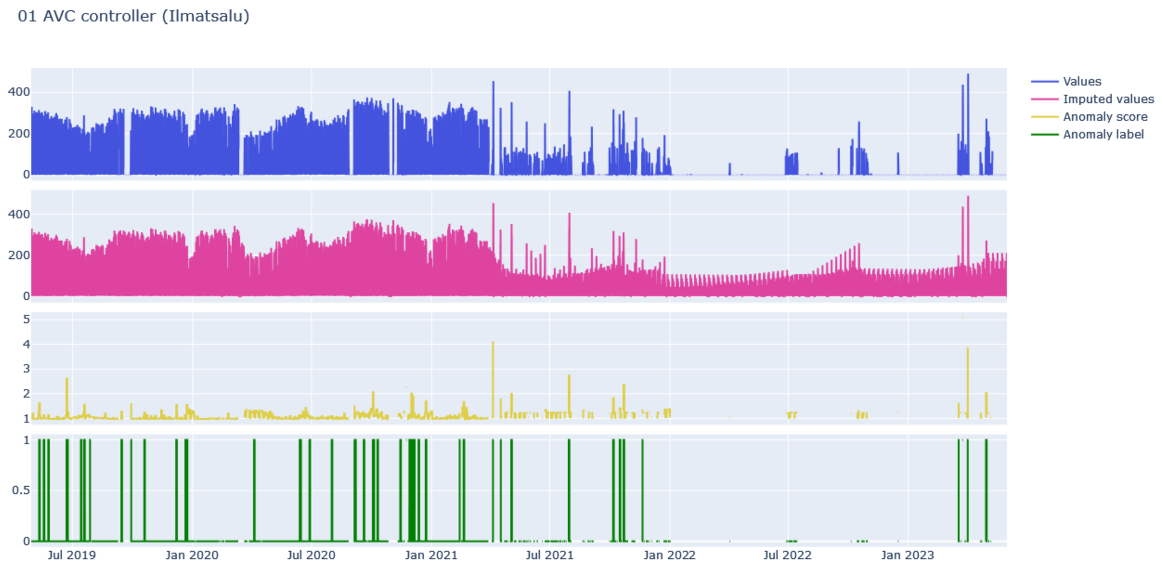
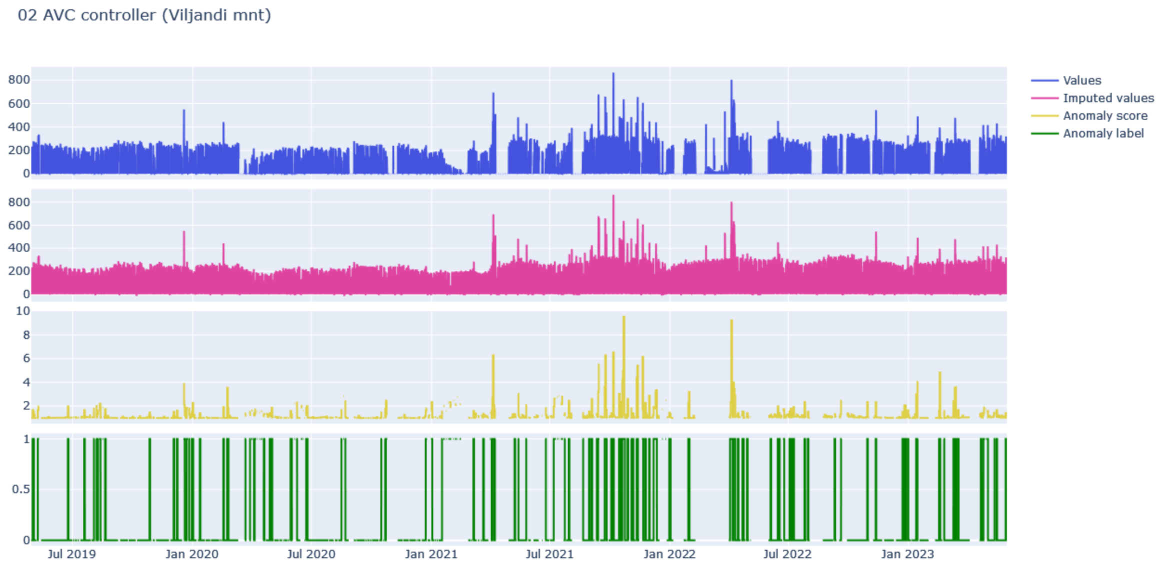


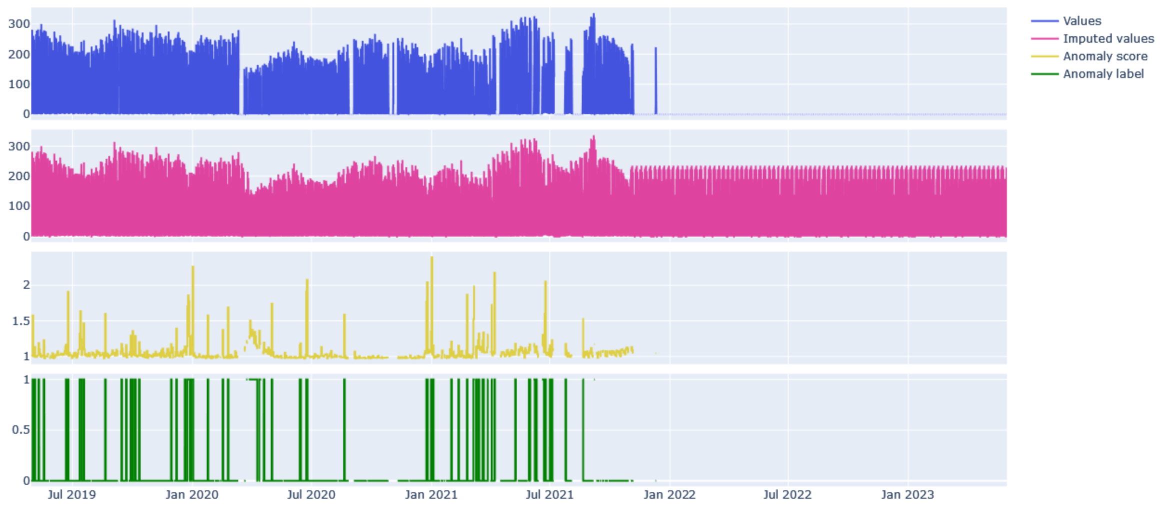
Figure 42: Data gaps by duration. A logarithmic scale was used to fully display all the datapoints without hiding the distribution of the data.

Appendix III: AD and imputation service's output

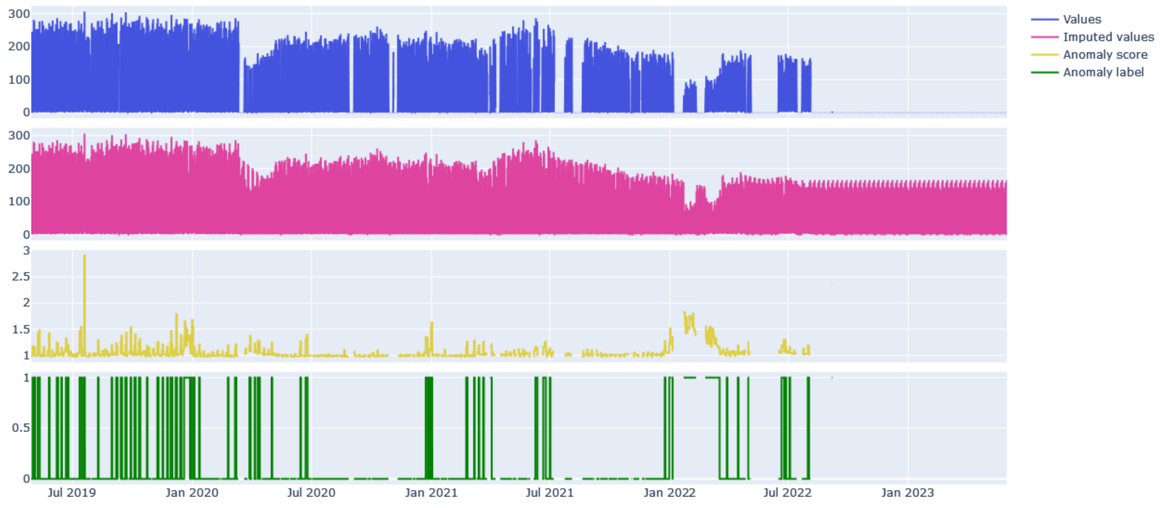




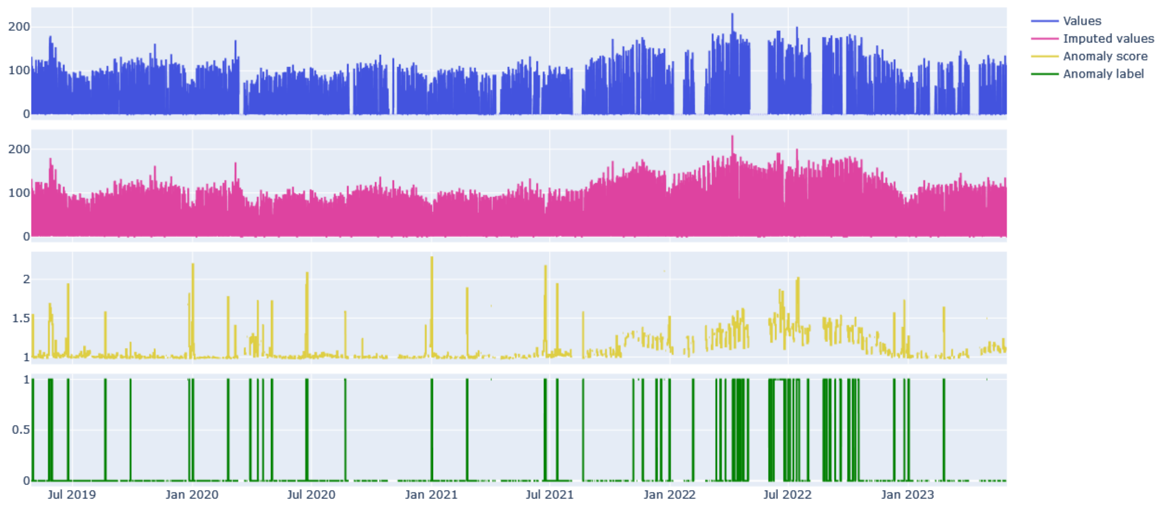
03 AVC controller (Riia mnt LS)



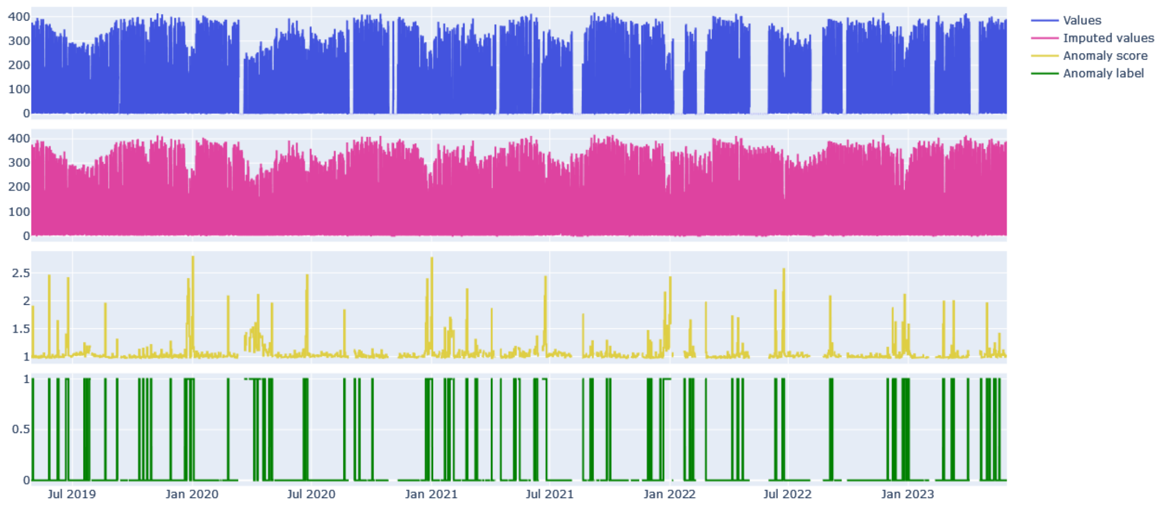
04 AVC controller (Riia mnt LV)



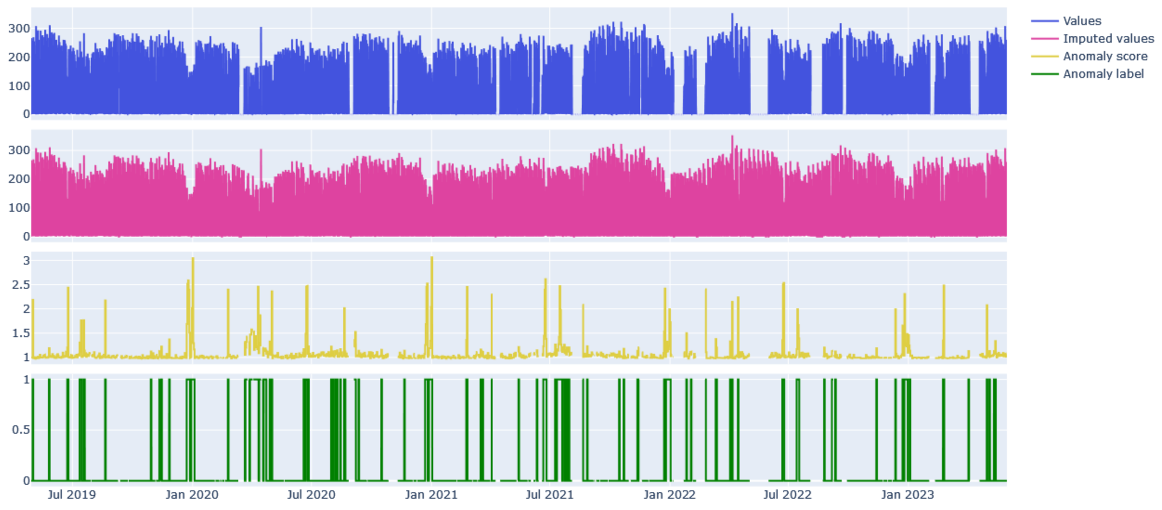
05 AVC controller (Roopa)



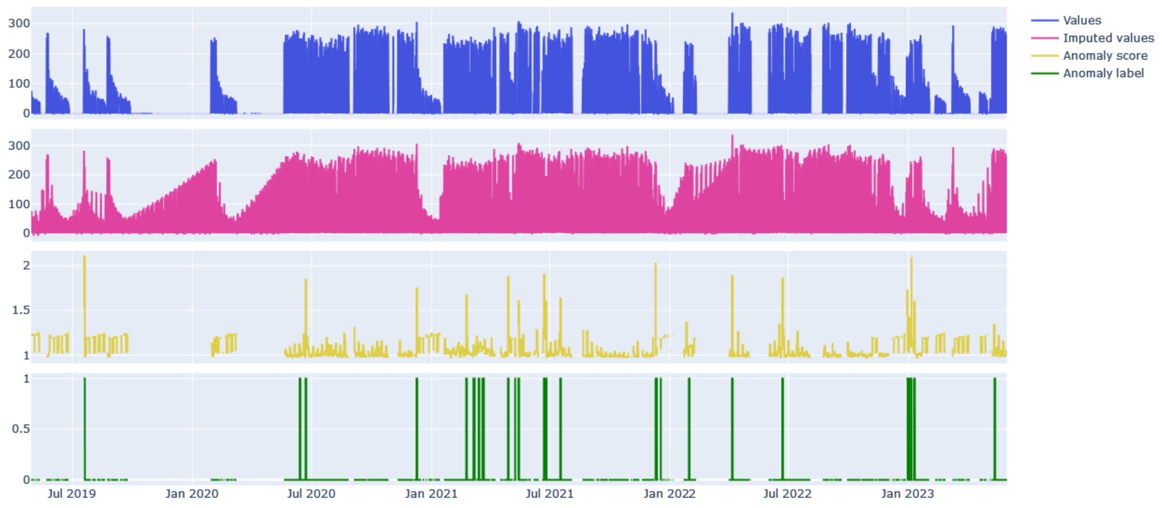
06 AVC controller (Võru)



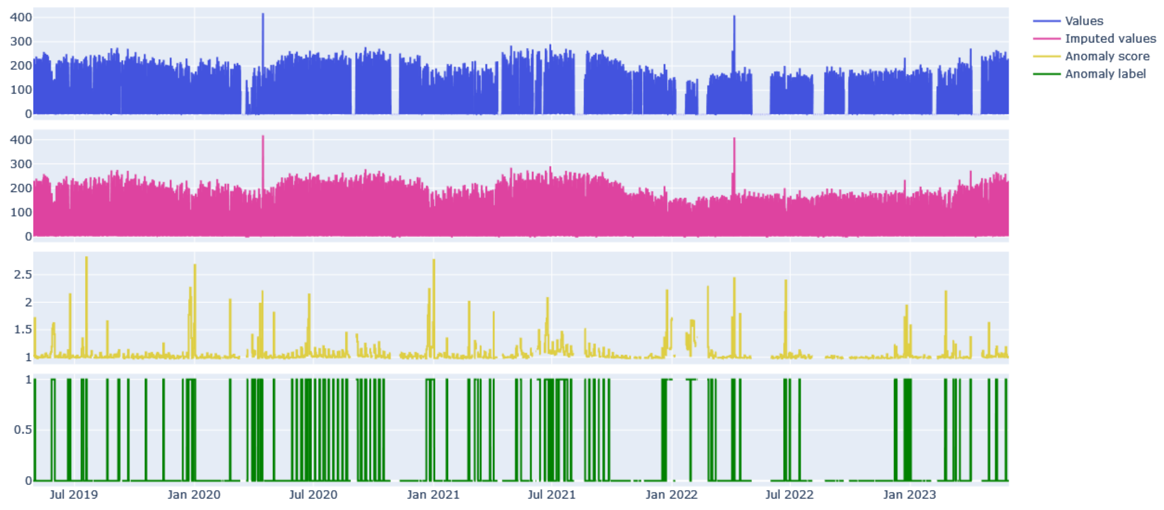
07 AVC controller (Ringtee 1)



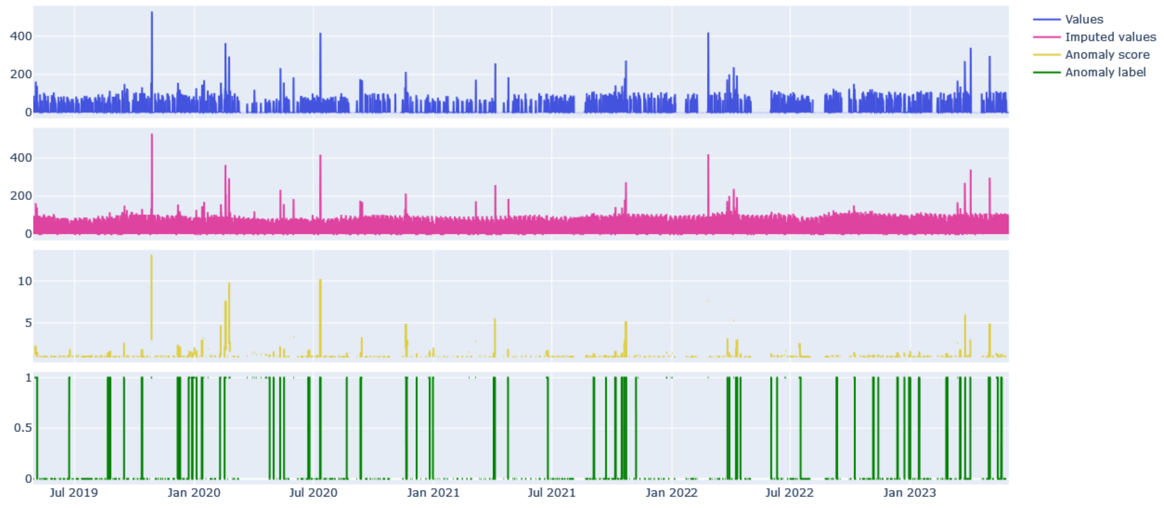
08 AVC controller (Ringtee 2)



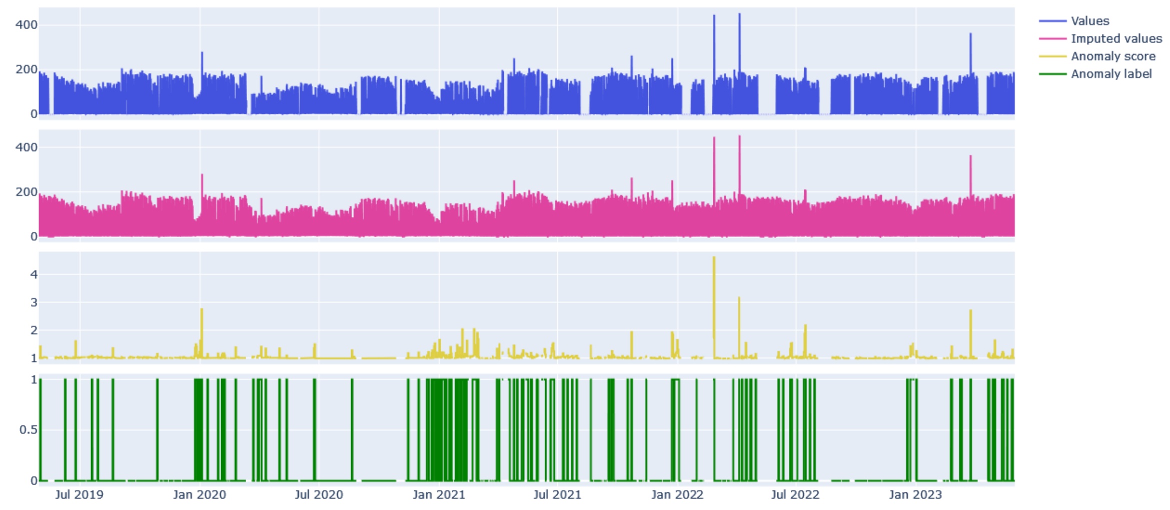
09 AVC controller (Ringtee 3)



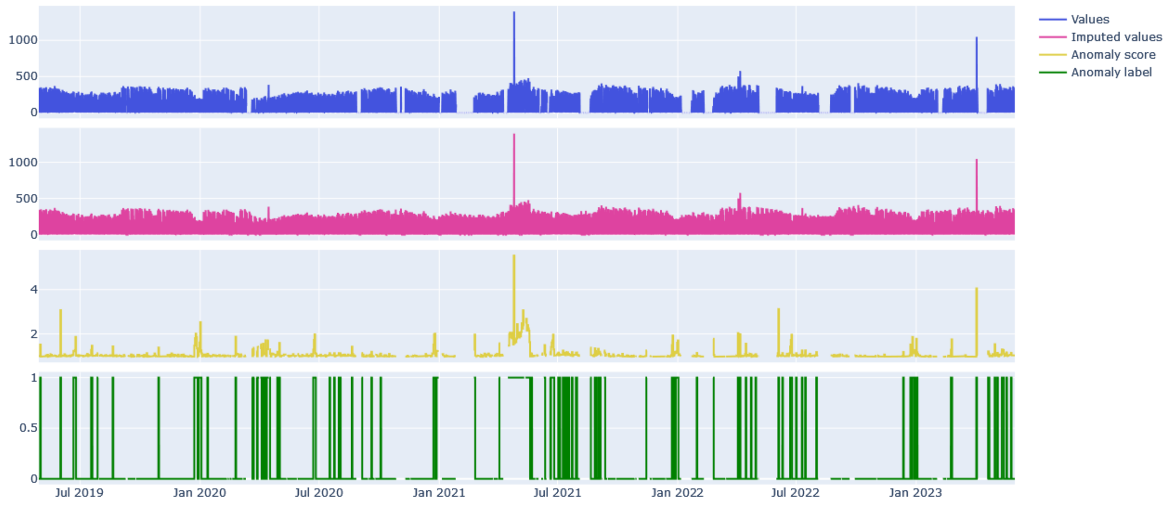
10 AVC controller (Ihaste)



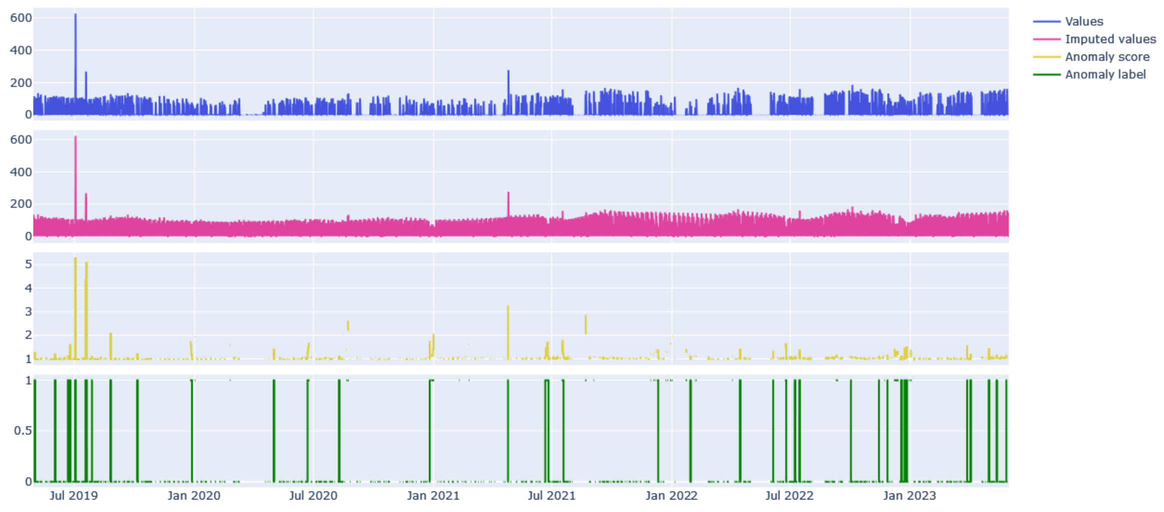
11 AVC controller (Lammi)



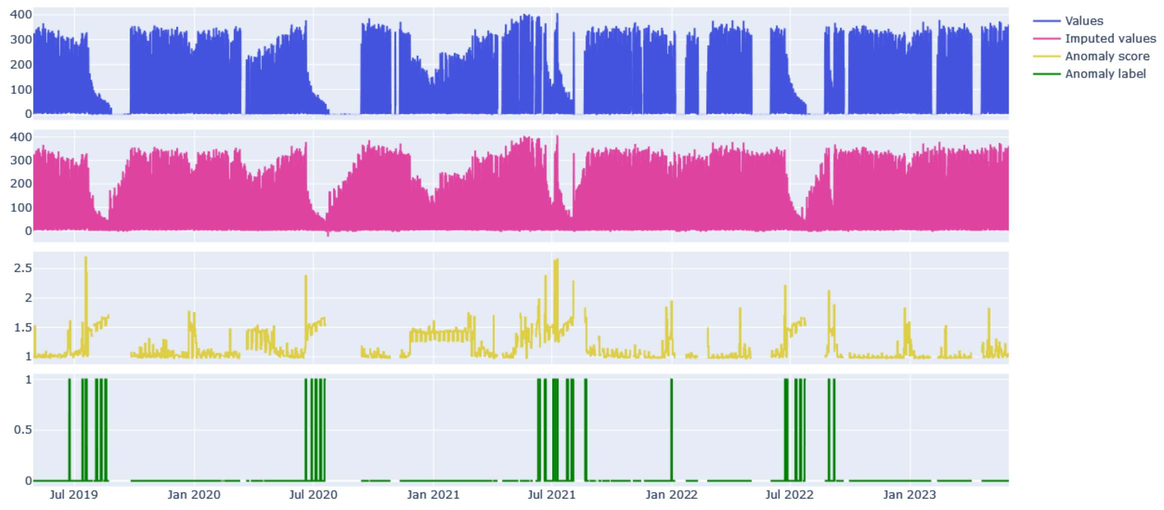
12 AVC controller (Räpina mnt)



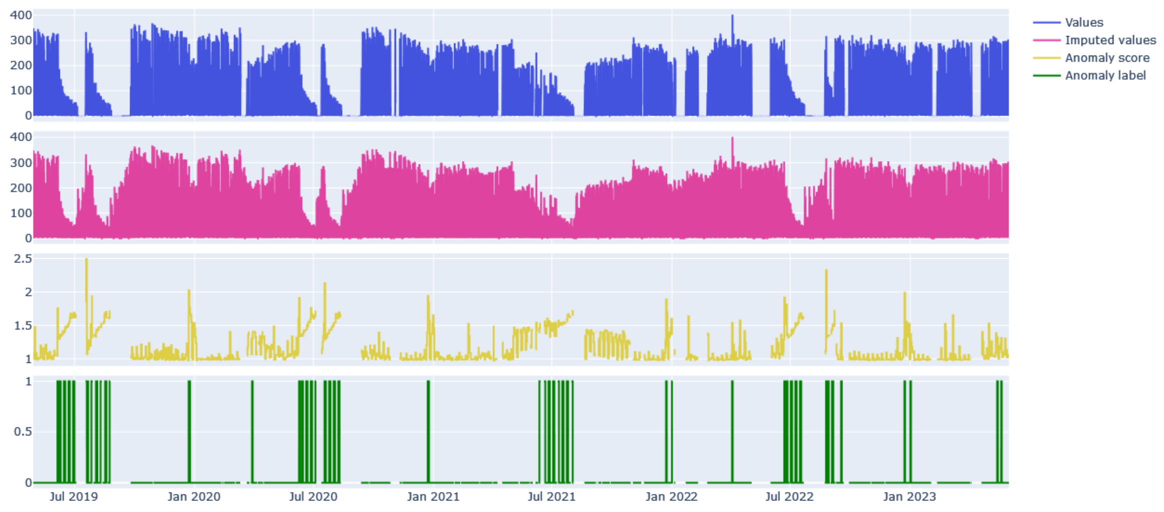
13 AVC controller (Rõõmu)



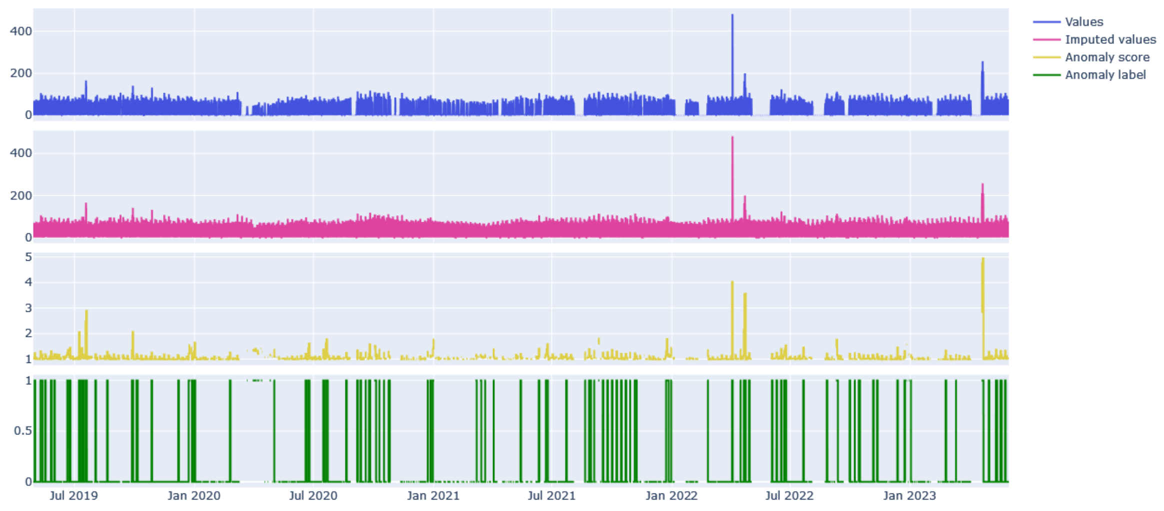
14 AVC controller (Narva mnt)



15 AVC controller (Aruküla)



16 AVC controller (Tiksoja)



Appendix IV: Non-exclusive licence to reproduce the thesis and make the thesis public