S. Krajči (ed.): ITAT 2018 Proceedings, pp. 108–115
CEUR Workshop Proceedings Vol. 2203, ISSN 1613-0073, © 2018 Andrea Galloni, Balázs Horváth, and Tomáš Horváth

Real-time Monitoring of Hungarian Highway Traffic from Cell Phone Network Data

Andrea Galloni, Balázs Horváth, and Tomáš Horváth

Department of Data Science and Data Technologies,
Faculty of Informatics,
ELTE – Eötvös Loránd University in Budapest
{andrea.galloni,balazs.horvath,tomas.horvath}@inf.elte.hu,
http://t-labs.elte.hu/

Abstract: A lightweight model for real-time monitoring of the load of Hungarian highway traffic is presented in the paper. The input data of the model are cell phone network event records provided by Magyar Telekom Nyrt., the major Hungarian telecommunication company. The output is a classification of the level of crowdedness of the Hungarian highways inferred from the activity level of the mobile telecommunication infrastructure. While processing, a data-stream is flowing through a chain of simple but efficient data structures. For computing anomalies against the usual behavior of the traffic at given segments of the highway, so-called break-points, known from the SAX representation of time-series, are utilized which require cheap computation. The model is implemented as a server application able to feed a client web-based visualization application implemented for demonstration purposes. The experiments, performed on anonymized data covering one month of cell phone records, show that the presented model is computationally cheap, it efficiently runs even on low-end hardware such that Raspberry Pi.

Keywords: Anomaly Detection, Mobile Data Analytics, Visualization

1 Introduction

A method resulting from an industrial research project is presented in this paper. The presented method has been developed for a specific and well-defined use case: inferring the level of the mobility traffic load on Hungarian highways from cell phone network data. The data have been provided by the industrial partner, *Magyar Telekom Nyrt.*, the major Hungarian telecommunication company, a subsidiary of *Deutsche Telekom AG*.

Experimental outcomes are positive and promising as the underlying core model is computationally light and simple. The data structures used and the overall system architectural-design could be, possibly, exploited for other applications or use cases. More specifically, the presented model can be used where there is the need to detect anomalies in time series given a set of nodes and the logs providing quantitative information describing the activity of such nodes over time. Even if the developed framework is specifically build to infer information regarding the Hungarian highways mobility infrastructure through the analysis of the mobile telecommunication infrastructure, the core model can be adapted to different scenarios and different data logs such as tower cell crowdedness or Internet backbones nodes activity monitoring.

1.1 Related Work

Nowadays we are witnessing to a constant increasing speed of networks, furthermore the capability to store and process conspicuous amount of data can be performed at affordable prices. This new scenario enables telecom operators to store and process big quantities of logs triggered by a countless number of events. Along the past years, the research community proposed several models regarding the possibility to infer or predict information regarding the status of the mobility infrastructure analyzing the mobile telecommunication event logs. Furthermore, the evolution of new telecommunication technology standards such as 5G will bring more efficient and accurate localization techniques leading to more precise analysis and estimations [7].

In [6], authors present quite a complex model able to estimate the traffic flow making use of anonymized temporal series of cell handover logs, building state diagrams and using Markov Models in order to detect car accidents.

On the other hand, in [8], authors propose a framework mining several heterogeneous data sources making use of the MapReduce programming-model (more precisely using Apache Hadoop) in order to process big amounts of data in a reasonable amount of time involving high-end hardware and clusters of machines. The authors of this contribution are able to estimate the traffic volume and the speed of the traffic flow.

In [9], the authors provide a full overview of methodologies providing a possible list of necessary steps in order to infer traffic information such that i) location data collection, ii) terminal classification in order to determine which mobile terminals are located on the road and which means of transport they are in, while iii) map matching phase in order to link the extracted location data with the mobility infrastructure, iv) the route determination process used to determine the path of the vehicles while the last step is to perform v) the estimation of the traffic state.

In order to estimate traffic flows the use of origin/destination matrices have been tried out in [10] and [11],

however, as pointed out in [6], these solutions might be computationally expensive from the technical point of view and hard to scale when the size of the user set tends to grow. On the other hand, from the law regulation perspective these solutions might see limitations on real deployment scenarios due to privacy concerns and strict regulations, especially the ones applying within the European Union.

Within this contribution, a minimalistic approach to traffic load detection is introduced exploiting just events triggered by the active utilization of the *User Equipments* (Calls, SMS and Mobile Data Usage) without involving logs related to lower level signalling protocols such as cell handover event logs. The aim of this research was to discover at which extent and precision is possible to infer reliable traffic analysis with minimalistic datasets and minimal computing costs. A server application able to feed a web-based visualization application has been implemented for demonstration purposes. Experiments provided on real but anonymized data covering one month of cell phone records show that the the presented model is promising and is able to efficiently run even on low-end hardware.

The rest of this paper is organized as follows: Section 2 gives a description of the available data and the procedure utilized to match telecom data with geographical-map data. In Section 3, a detailed description of the system architecture and the core estimation model are provided. In the Section 4 the process of traffic load classification is described. The following Section 5 contains experimental results and measurements. Finally, in Section 6 some conclusions and plans for future work are provided.

2 Data Sources and Framework Initialization

An important part of the presented framework is its initialization to the specific use case, such that highway traffic load monitoring, in our case, what consists of understanding the data and selecting the towers to be considered relevant by the framework.

2.1 Telecom Data

The industrial partner has granted access to a dataset[1] containing several `.csv` (Comma Separated Values) files organized in a daily basis containing two main kinds of information such that

Call Detail Record (CDR) data concern the activity logs of each user interacting with the network on a daily basis. A CDR is produced by a telephone equipment that documents the details of a call or other telecommunications events (e.g. notifications, short message service or signaling protocols) that involves the telecom provider infrastructure.

The dataset is composed by several files accounting a size of 500GB. The entire information contained within the dataset covers a range of 31 days, more precisely between 15th September 2016 and 15th October 2016, where all the unique identifiers referencing to the users have been anonymized on a daily basis. The overall number of logs is around 200 million records per day.

In order to work with just the useful data all the unnecessary information contained in the dataset such as the nature of event logs and other information regarding customer related data (e.g. phone and events identifiers) have to be discarded by the framework.

At the end of this process, the CDR files contains three kind of attributes as presented in Table 1, namely, the *Unique User Identifier* (UUID) re-anonymized on a daily basis (to prevent tracking of user movement across more days), the *date-time* information related to the log event and the *Tower Identifier* (TID) providing information from which cell-tower the event has been triggered.

Cell Reference (CR) data provide informations about the mobile radio-towers and their positioning within the Hungarian territory. As illustrated in the Table 2, CR data contain three attributes, namely, the *Tower Identifier* (TID) which connects the CDR data with the CR data, the *Latitude* and the *Longitude* regarding the given tower. Due to how the industrial partner gathered and anonymized the data, process on which the authors were not involved, some records contained within the CDR files hold UUIDs with *NULL* value or in some cases hold inconsistent TIDs. Namely some TIDs contained in the CDR logs do not match any of the TIDs in the CR data. The UIIDs inconsistencies are uniformly spread over the locations while for the inconsistent TIDs is not possible to draw any conclusion about the geographical regions affected. In case of those inconsistent logs, it is not possible to get the subject performing the action or the location of the event. For this reason all the affected records, which are around 30% of all the records, are affected and have to be handled (discarded) by the framework during the on-line computation.

2.2 Geographic Maps Data

In order to get information about Hungarian highways the research relied on OpenStreetMap[2] (OSM) data. The open project makes available data about roads, trails, cafes, railway stations and other basic map features from all around the world.

[1] Due to the signed non-disclosure agreement between the academic and the industrial partner of the project, each sample of the data, e.g. the Tables 1 and 2, presented in this paper are synthetic, i.e. contain fictive information.

[2] `https://www.openstreetmap.org/`

UUID	date-time	TID
6776554S3449	2016-09-15 13:30:41	10B32F03E10CB
777865354435	2016-09-15 00:27:50	10DA0232324BC
677655453449	2016-09-15 17:44:08	00443344DFFEA

Table 1: A synthetic CDR data sample with fictive UUID, date-time and TID data, for illustration purpose .

TID	Latitude	Longitude
6776554S3449	46.291311	17.366325
777865354435	46.282342	17.357244
677655453449	46.366745	17.364112

Table 2: A synthetic CR data sample with fictive TID, Latitude and Longitude data for illustration purpose.

First of all information about the Hungarian country borders have been extracted, then the data regarding only highways has been kept obtaining a `.json` file containing informations about each highway divided in several segments, where each segment contains information describing a small section of the highway (e.g.: name, type, speed limit) and its location. The length of each highway's section depends on the topography of the area, the density of the population and the radio-technology of the cell towers of the operator. The outcome of this phase, i.e. the detected borders, can be observed in the Figure 6.

Detecting Relevant Towers In order to monitor the highways infrastructure traffic and exclude irrelevant information from the data model an additional filtering and selecting step has been performed. After this phase, only the relevant towers that have a strict correlation with the highways infrastructure have been kept.

The density of the cell tower placement and its spatial characteristics represent a crucial issue in terms of space-resolution within the developed monitoring system. Provided the mostly flat characteristics of the Hungarian landscape, is possible to assume that the cell towers displacement is mostly not conditioned by the topographic properties of the surrounding areas but rather follows the density of the population over the whole territory. This characteristic of the cell towers placement is due to several factors such as scalability, laws of physics and signal processing theory. Figure 1 illustrates the density of the city of Budapest and its east country side area from which it is possible to observe that in rural areas cell towers are placed close to the highway in order to provide the radio-signal to travelers.

In order to detect cell towers which lead to the crowdedness status of that specific section of the highway, an ad-hoc algorithm have been developed based on a *QuadTree* [1] data structure. The generated tree contains all the coordinates of the cell towers that are listed in the CR data. Then, for every highway section the closest tower cell has been found querying the tree structure. After this process,

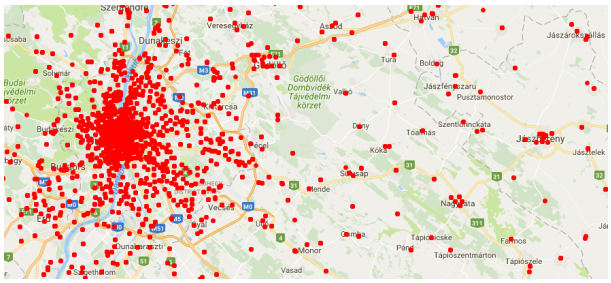

Figure 1: Density of cell towers for Budapest Urban and Eastern Rural Area, an illustrative example.

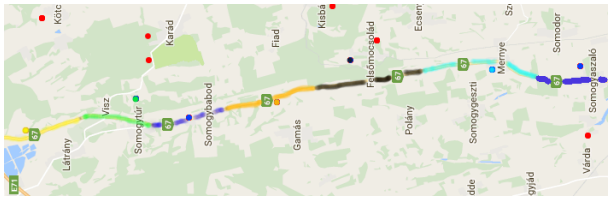

Figure 2: Relevant towers identification and towers competence attribution, an illustrative example.

as shown in Figure 2, each highway section is linked to a specific cell tower while all the towers not related with the highways will not be considered by the framework while processing.

3 The System Architecture

The framework has its roots in several components, i.e. the following data structures (called dictionaries, according to Python notation) and logical units.

3.1 Data Structures

Relevant Towers Dictionary (RTD) RTD is one of the main data structure on which most of the others are based on. In fact the RTD represents an `HashMap` having as keys the identifiers (TID, see Table 2) of all the relevant towers and as values the geographical coordinates of given towers as strings. RTD is the result of the module for detection of relevant towers, described above. This dictionary remains constant in the framework and changes only if there are changes in geographical locations of cell towers such that a new tower is placed near the highways, for example.

User at Relevant Towers Dictionary (URTD) URTD is a `HashMap` and it has as keys all the UUIDs (Unique User IDs, see Table 1) of the users whose previous logs were triggered by relevant towers, namely, those logs who had their TID appearing in the RTD keys, and, as values the TID of the tower the given user has been related to for the last time. At the startup of the system, this data structure is empty and at the beginning of a new day, due to the UUID re-anonymization on a daily basis, it is reinitialized.

Tower ID Counter Dictionary (TIDCD) TIDCD is a `HashMap` and it has as keys (TIDs) all the relevant towers while it has as values counters representing the number of users whose last log have been related to that specific TID.

3.2 The Status Maintainer Module (SMM)

As soon as a log event is triggered, the SMM is delegated to keep track of the last location of the user (attaching to him/her the proper TID in the URTD) and increases or decreases the counter of the proper TID within the TIDCD according to the situation. This module acts as a supervisor moving the subscribers from a tower to another, keeping the model up to date with the information provided by the event logs. SMM is delegated to interpret and take decisions based on the information provided from the logs with the following functionality:

As soon as the application is started, both the URTD and the TIDCD Hash Maps are empty. As the first incoming log having a TID present in the RTD key-set is processed, the SMM will fill the URTD with the UUID as key and the TID as value, then, it will initialize the counter of the specific TID key within the TIDCD to 1. If a second log from the same user will come but, this time, with a different and relevant TID then the counter of the old TID (the last "location" of the user) within the TIDCD will be decreased by one unit and suddenly the corresponding URTD's value will be updated to the actual TID inferred from the log querying the RTD and, finally, the corresponding TIDCD value (for the actual TID the user is connected to) will be increased. In the last case, if the subscriber's handset will trigger a log which is not related to a relevant tower, the counter of the old TID within the TIDCD will be decreased by one and the key within the URTD corresponding to the specific UUID will be removed (the user has left the set of towers delegated to monitor the highways mobility status). All the event logs (records) containing a non-relevant TID (user is not on a highway) and UUID not stored in URTD (user was not on a highway before) are immediately discarded. Figure 3 provides an illustration of the SMM logic.

3.3 The Evaluator Module (EM)

The EM is the core module of the framework, which is responsible for classifying the status of the load of a specific highway segment. It uses as an evaluation model described in the next section where the number of classes is determined by a parameter.

The EM module is triggered every time a time frame (a parameter of the framework) expires. At this point it is time to evaluate the status of the whole system. At this stage the EM iterates over all the TIDCD, computes all the means and standard deviations using the past data necessary to evaluate the actual status of all the relevant TIDs and then finally classifies every segment of a given highway into one of the predefined classes. For example, in case of 10 classes, the class 5-6 corresponds to normal traffic, 7-8 to higher while 9-10 to very high traffic, 3-4 to lower and 1-2 to very low traffic on a given segment of the highway.

3.4 The Notification Delegate Module (NDM)

The NDM is the part of the framework responsible to constantly collect the result of the EM and update the clients about the highways status sending all the needed informations as a `json` payload. While this process takes place, an ID conversion is performed. In fact, the back-end and the front-end of the framework, due to security reasons, do not share the same internal IDs for representing the TIDs. Once finished, the control is passed to the Notification Delegate Module responsible for communicating with the client (described below).

4 Classification of the Traffic Load

In order to classify the load of a segment of highway, one should consider the past data as a reference for the evaluation of the new entries. Given the topographic features of the Hungarian landscape it is expected that the activity of cell towers close to the highway have a low *static noise* (in [6] authors highlights that systems relying on cell telecom logs for traffic estimations within urban areas could suffer high loss of precision due to event logs triggered by non-traveling users). In fact, in this specific case the majority of the event logs are generated mostly by travelers and it is easily possible to observe that the number of travelers in a given time-range tend to be similar for different week-days.

Provided the latter observation, it is possible to build a model such that in order to classify the load of a specific road segment in a precise time-range of the day (e.g.: between 12:00 and 12:15) compares the value to be classified against the values of load within the same time range of the previous days. Here, a sort of seasonality of the traffic has to be considered, e.g. there is more heavier load during the rush hours while less traffic at nights.

An other, interesting phenomenon is that sometimes a traffic anomaly can become normal with time. For example, consider a longer construction work on a highway causing the close of some parts (e.g. lanes) of the road. In

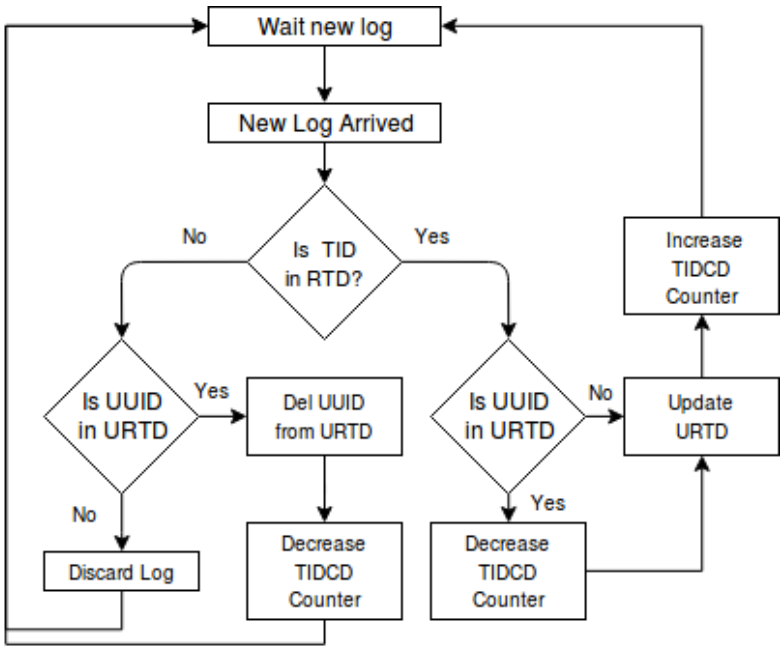

Figure 3: The logic diagram of the Status Maintainer Module

the time of its appearance, since it is sudden for the traffic, it is considered an anomaly and results in traffic jams. However, with time, the traffic normalizes such that people get used to it (e.g. start using alternative routes) and the notion of heavy load changes.

All of the above observations lead to the straightforward use of time-series for representing the given problem of traffic load monitoring and classification.

4.1 Utilizing Breakpoints

The proposed model for classification of traffic load in various highway segments utilizes well-known concepts in Symbolic Aggregate Approximation (SAX) of time series [2, 3]. SAX makes use of Piecewise Aggregate Approximation (PAA), a computationally very cheap method [4] since it operates with *arithmetic mean* and *standard deviation*, both very cheap operations.

SAX allows a time series of arbitrary length n to be reduced to a string of arbitrary length w, ($w \ll n$) with the alphabet size $a > 2$ (the number of letters used to represent the time-series using SAX). In this process the data is divided into w equal sized "frames". The mean value of each frame is calculated and a vector of these values becomes a reduced representation. For further details, refer to [2, 3].

Assuming that the distribution of the values over the time-series follows a normal distribution, it is possible to subdivide the time-series into so called *breakpoints B*. Breakpoints are a sorted list of numbers $B = (\beta_1, \cdots, \beta_{a-1})$ such that the area under a $N(0,1)$ Gaussian curve from β_1 to $\beta_{i+1} = 1/a$ where β_0 and β_a are defined as $-\infty$ and $+\infty$, respectively. The advantage of utilizing breakpoints is that they do not need further computation but may be determined by simply looking them up in a statistical table, as illustrated in the table 3 containing the breakpoints dividing a Gaussian distribution in an arbitrary number (from 3 to 10) of regions.

The final key-concept of the proposed model is that it does not represent the data in the past with a SAX representation, however the system classifies a new entry using the breakpoints generated making use of the data recorded in past in a SAX-like fashion just performing a lookup on a small table.

An efficient way to detect anomalies in time series is that it is enough to compare the breakpoint determined for the actual time frame to the breakpoints determined for the corresponding time frame(s) in the past, for example, the same time of the day before or the same time of the same day a week before, etc. Depending on the number of classes to which the framework should classify the new entry in the dataset, the right column of the statistical table has to be implemented in the system and then looked up. In order to give a wider freedom of tuning, this value has been kept as a parameter of the framework.

Historical Data Representation The data from the past are stored in an efficient-to-load binary format making use of Python's Pickle module. Files are stored in a daily format in the form of a `HashMap` of arrays having as keys the TIDs. Once loaded, the files are reshaped in *MxN* matrices, one for each TID, where M is the number of timeframes and N represents the number of days to consider in the past. All the tests in this research were performed setting the time frames at 15 minutes and considering the last

	a							
β_i	3	4	5	6	7	8	9	10
β_1	-0.43	-0.67	-0.84	-0.97	-1.07	-1.15	-1.22	-1.28
β_2	0.43	0	-0.25	-0.43	-0.57	-0.67	-0.76	-0.84
β_3		0.67	0.25	0	-0.18	-0.32	-0.43	-0.52
β_4			0.84	0.43	0.18	0	-0.14	-0.25
β_5				0.97	0.57	0.43	0.14	0
β_6					1.07	0.67	0.43	0.25
β_7						1.15	0.76	0.52
β_8							1.22	0.84
β_9								1.28

Table 3: Statistical table for determining the values of breakpoints

15 days in the past. This representation have been chosen because once the system is running in real-time with a continuous data-stream then it is easy to shift the matrix data and recompute all the means and standard deviations efficiently. This shifting and re-computation operation would be performed once per day at a given time (e.g. midnight) exactly when the system is subject of a reinitialization or when the log re-anonymization process is performed.

5 Experiments

Given the lack of open systems providing precise information about the traffic flows over time we validated the system over specific traffic congestions. The goal of the experiment is to validate the method looking for a correlation between the highways status and the telecommunication infrastructure. In order to validate the results of the developed system authors asked the collaboration of *utinform*[3], a department of *Magyar Közút Nonprofit Zrt.* whose responsibility is to collect information and monitoring the roads traffic. They gather information from heterogeneous data sources: the employees of the company monitoring the highways, the county directorates and the engineering departments employees. Utinform, in order to cross-validate the crowd sourced data sets, is also cooperating with institutions such as police, disaster management, and public transport which have their own feed to post their information to the system. Furthermore, the system has a crowd source interface, where people can submit experienced traffic anomalies. The goal of utinform is to provide fast and validated traffic information to the drivers in whole Hungary.

The data received include information regarding traffic congestions on all the Hungarian highways. The dataset contained data from October the 1st to October the 14th, 2016 regarding congestions with a length of at least 3 kilometers. Table 4 shows the validation data and the results of the proposed framework.

Setting the number of classes to 10. We define a correct detection when there is at least 50% overlap (in terms of time) over the validation data with traffic classification level equal to 9 or 10. With this setup eleven out of fourteen (79%) congestions have been successfully detected. The time series have been quantized with time windows of 15 minutes, thus during the testing phase the classification takes place each fifteen minutes.

It is interesting to note that the majority of the anomalies are detected earlier than the data provided by utinform. This can be a proof that the proposed solution is able to spot congestions when these are shorter than three kilometers. The outcome is similar for what concerns the end of the congestions, in this case the proposed solution tends to detect the end of an anomaly later than the validation data set. Three congestions out of fourteen have not been detected, however this can be due to at least two reasons: the telecom operator (because of industrial secrecy concerns) did not provide to us any detail regarding the range of antennas or their direction, thus, there might be a chance of inaccuracies along the phase of matching the geographical maps with the towers' positions in order to define the competence of each tower w.r.t. the highways segments. Another reason could be due to data inconsistencies. In fact, as mentioned in Section 2, one third of the data have been dropped.

Figure 4 represents the classification value over time slices of fifteen minutes for the highway segment suffering for the congestion described in Table 4 on row three. The red (in black and white print: the dark) vertical line represents the time slot when the anomaly is detected on the validation set. Here, a congestion is defined when the classification values are equal or greater than 9. On the other end, Figure 5 represents the number of handsets (an estimation of the number of travelers) involved in the congestion.

The application framework have been completely developed in `Python 3.6` and it have been demonstrated to be very efficient. Although the system is designed to work on-line receiving a stream of data in real-time, in order to measure the performances of the model, we decided to exclude the streaming and the networking module, finally we tested the model in off-line mode. One log per time is read and suddenly processed. With this approach we

[3] `http://utinform.hu/`

Cong. Start Time	Cong. End Time	Det. Start Time	Det. End Time
11:00	11:40	10:30	12:15
06:35	07:45	07:15	07:45
09:30	22:10	09:00	00:00
17:55	09:05	none	none
06:52	07:30	07:00	08:15
14:50	16:50	16:45	17:30
08:40	18:10	08:15	18:45
14:00	17:10	14:45	19:25
10:00	11:20	08:45	13:45
17:23	20:15	16:15	20:00
07:20	08:50	07:30	10:30
02:55	04:40	none	none
15:00	16:00	15:15	16:15
06:45	07:25	none	none

Table 4: Experimental Results indicating congestion (Cong.) start and end times as well as detection (Det.) start and end times.

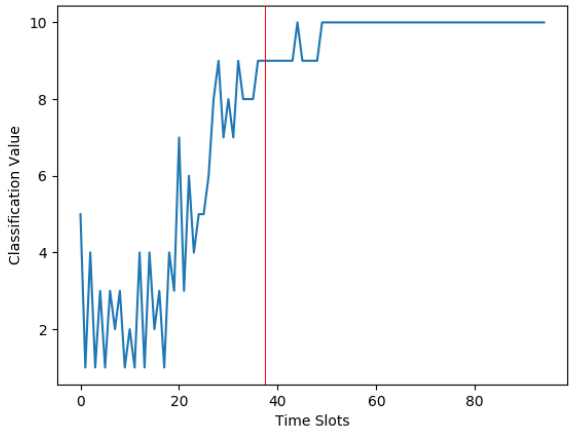

Figure 4: Classification values over time slices and the time of a congestion (red/dark line), an illustrative example.

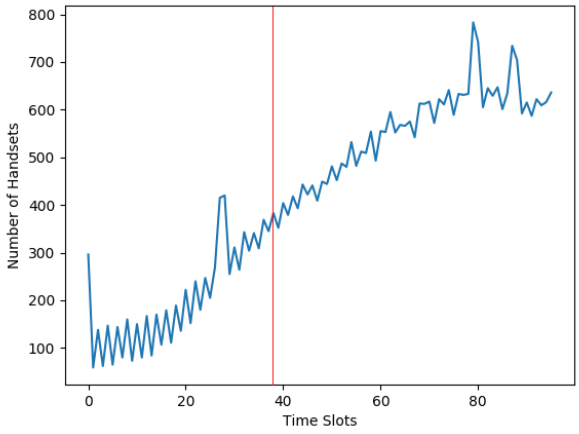

Figure 5: Number of handsets involved in a congestion and the time of the congestion (red/dark line), an illustrative example.

maintained the architectural design of the framework and no-delay straming process has emulated while at the same time avoiding networking related delays. Running the application on an *Intel i7 6700K* with 16GB of DDR4 RAM, on average, manages to process the logs for an entire day within less than 10 minutes with a peak of RAM consumption of 32MB. Furthermore, the application framework have been deployed on a `Raspberry Pi 3` where the running time needed to process an entire set of logs representing one day is in around 1 hour, in average.

5.1 Visualization

First the communication interface need to be mentioned between the back end and the front end part. The front end is a web based application, it uses `HTML` and `JavaScript`, which communications with the Python back end through Web Sockets with `.json` files. At the initialization stage, first the front end asks the back end for mapping of tower IDs and highway sections, as it was mentioned before at the geographic data section. After the initial step, the back end is sending messages which the front end processes and shows on the map. These messages are json files, containing a list of objects with three attributes: ID, value, anomaly. The ID stands for the tower IDs and the value is an int from 0 to 9 showing the traffic load on that segment, and the anomaly flag is giving information that this value is the expected for the given time window or it is deviating from the usual traffic load on that area. In order to keep the low cost functionality of the system, the visualization had

to be carefully built as well. An open-source JavaScript library, `LeafletJS`[4] was used to place layers on the particular highway segments. This library has relatively low cost of handling layers and as it is expected from an application where the traffic load is constantly changing this was a critical feature. Each layer is a colored visualization of the value given to the particular highway section, an example can be seen on the Figure 6, where the spectrum is from blue to red, representing the low to high traffic load.

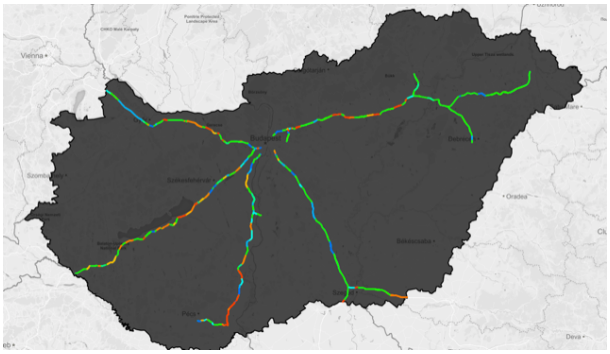

Figure 6: The traffic load visualized on the highway segments, an illustrative example.

6 Conclusions

A lightweight traffic monitoring and traffic jam detection framework has been presented in this paper based on HashMap data structures and methods for breakpoint detection well-known in time series classification. The used concepts require cheap computation and, basically, minimal tuning phase opposite to the case of tuning the hyperparameters of machine learning algorithms. The few parameters of the framework such that the time window or time frames as well as the number of breakpoints can be set according to an available domain knowledge or user expertise. However, to avoid false positives, it is recommended to tune the presented framework before implementing it into a production environment.

The SAX representation had already been used as a tool for time series classification. However, the proposed discretization procedure is unique in that it uses an intermediate representation between the raw time series and the symbolic strings. Furthermore the aim is not to classify full time series as in [5] but rather to classify new incoming single values.

The presented framework was tested using real data. Due to the sensitive nature of the project the presented framework was developed within, the authors cannot disclose the source code nor the data used in experiments, in this time. Experiments show that the proposed framework is promising and worth further development and adaptation to other use-case scenarios.

[4] `http://leafletjs.com`

Acknowledgements

Authors would like to thank Magyar Telekom Nyrt. and utinform.hu. The research has been supported by the European Union, co-financed by the European Social Fund EFOP-3.6.3-VEKOP-16-2017-00001 and the project "Open City services" funded by Magyar Telekom Nyrt.

References

[1] R. Finkel, J.L. Bentley (1974). Quad Trees: A Data Structure for Retrieval on Composite Keys. Acta Informatica 4 (1), 1–9.

[2] Lin, J., Keogh, E., Lonardi, S. and Chiu, B., 2003. A symbolic representation of time series, with implications for streaming algorithms. In Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery (pp. 2-11). ACM.

[3] Lin, J., Keogh, E., Wei, L. and Lonardi, S., 2007. Experiencing SAX: A Novel Symbolic Representation of Time Series. Data Mining and knowledge discovery, 15(2), pp.107-144.

[4] Zhang, Y. and Glass, J., 2011. A piecewise aggregate approximation lower-bound estimate for posteriorgram-based dynamic time warping. 12th Conference of the International Speech Communication Association, pp.1909-1912.

[5] Senin, P. and Malinchik, S., 2013, December. Sax-vsm: Interpretable time series classification using sax and vector space model. In Data Mining (ICDM), 2013 IEEE 13th International Conference on (pp. 1175-1180). IEEE.

[6] Milani, A., Gentili, E. and Poggioni, V., 2009. Cellular Flow in Mobility Networks. IEEE Intelligent Informatics Bulletin, 10(1), pp.17-23.

[7] Hakkarainen, A., Werner, J., Costa, M., Leppanen, K. and Valkama, M., 2015, September. High-efficiency device localization in 5G ultra-dense networks: Prospects and enabling technologies. In Vehicular Technology Conference (VTC Fall), 2015 IEEE 82nd (pp. 1-5). IEEE.

[8] R. Khokale, A. Ghate (2017). Data Mining for Traffic Prediction and Analysis using Big Data. International Journal of Engineering Trends and Technology 48 (3).

[9] Gundlegard, D. and Karlsson, J.M., Road Traffic Estimation using Cellular Network Signaling in Intelligent Transportation Systems, 2009. In: Wireless technologies in Intelligent Transportation Systems. Editors: Ming-Tuo Zhou, Yan Zhang and L. T. Yan. ISBN: 978-1-60741-588-6 2009 pp. 361-392. Nova Science Publishers.

[10] White, J. and Wells, I., 2002. Extracting origin destination information from mobile phone data. In 11th International Conference on Road Transport Information and Control, London, 2002, pp. 30–34

[11] Wideberg, J.P., Caceres, N. and Benitez, F.G., 2006. Deriving Traffic Data from a Cellular Network. In Procedings of the 13th ITS World Congress, London, 2006.