On the Effect of Adaptive and Non-Adaptive Analysis of Time-Series Sensory Data

Şefki Kolozali*, *Member, IEEE,* Daniel Puschmann*, Maria Bermudez-Edo†, and Payam Barnaghi*, *Senior Member, IEEE.*
* Institute for Communication Systems (ICS), University of Surrey, Guildford, UK
† School of Information Technology and Telecommunications, University of Granada, Granada, Spain

Abstract—With the growing popularity of Information and Communications Technologies (ICT) and information sharing and integration, cities are evolving into large interconnected ecosystems by using smart objects and sensors that enable interaction with the physical world. However, it is often difficult to perform real-time analysis of large amount on heterogeneous data and sensory information that are provided by various resources. This paper describes a framework for real-time semantic annotation and aggregation of data streams to support dynamic integration into the Web using the Advanced Message Queuing Protocol (AMQP). We provide a comprehensive analysis on the effect of adaptive and non-adaptive window size in segmentation of time series using SensorSAX and SAX approaches for data streams with different variation and sampling rate in real-time processing. The framework is evaluated with 3 parameters, namely window size parameter of the SAX algorithm, sensitivity level and minimum window size parameters of the SensorSAX algorithm based on the average data aggregation and annotation time, CPU consumption, data size, and data reconstruction rate. Based on a statistical analysis, a detailed comparison between various sensor points is made to investigate the memory and computational cost of the stream-processing framework. Our results suggests that regardless of utilised segmentation approach, due to the fact that each geographically different sensory environment has got different dynamicity level, it is desirable to find the optimal data aggregation parameters in order to reduce the energy consumption and improve the data aggregation quality.

Index Terms—Smart Cities, Internet of Things, Time Series Analysis, Adaptive Segmentation

1 Introduction

Data collection, transmission, and processing of observations should be efficient in a stream-processing environment. IoT resources can constantly transmit raw sensory data, which can lead to communication overhead and high-energy consumption. Data aggregation approaches allow coping with large volumes of data and reducing the size of real-time data streams. On the one hand, sensory observations can be very dynamic and their analysis can be computationally expensive. On the other hand, while it is desirable to reduce the computational cost, general practice in sensory environments is to reduce the communication overhead, since the power consumption of computation for sensory devices is usually significantly lower than data transmission, as given in [1], [2]. For that reason, it is important to use a lightweight, adaptive, yet effective time-series analysis approach in stream processing.

In time-series data analysis, Symbolic Aggregate Approximation (SAX) is one of the most computationally low cost data aggregation approach [3]. It transforms a time-series into a discretised series of letters e.g. a word. However, it uses a non-adaptive window size in segmentation process (i.e. fixed window size) of time-series analysis. While it is common to use the same parameterisation for all sensors in the analysis of the same type of data streams (e.g. traffic, parking), it effects the performance of the time-series analysis approach. Utilisation of an adaptive window size in segmentation enables to overcome this issue with a function to dynamically predict the window size depending on the distribution or deviation of time-series data streams. In this context, SensorSAX has been introduced in [4] as an extension of SAX, which uses adaptive window size in segmentation of time series analysis. It adapts the window size based on the standard derivation of the data segment.

Nonetheless, utilisation of the same parameterisation for non-adaptive and adaptive window size in segmentation may not be ideal to deal with geographically and dynamically different sensory environments. Such dynamicity difference can cause time-series analysis problems for both approaches. Therefore, in this study, we provide a comprehensive analysis for both adaptive and non-adaptive segmentation approaches, namely SensorSAX and SAX, to examine how utilisation of the same parameterisation for all sensors that are located in different geographical locations can effect the overall performance of the data aggregation approaches based on memory, computational cost and aggregation quality for real-time stream processing framework[1]. We also present an information model to provide a machine readable representation for aggregated IoT data streams. This enables to describe the parameterisation that has been used in the data stream analysis along with the reliability information of the data streams. In order to investigate the performance of the framework with both of the approaches, traffic and parking datasets are collected from an open data platform, called Open Data Aarhus

1. The real-time stream processing framework has been developed in the scope of CityPulse project: http://www.ict-citypulse.eu/

(ODAA)[2], which contains city related information generated by various sensors deployed in the City of Aarhus. The framework is evaluated with 3 parameters, namely window size parameter of the SAX algorithm, sensitivity level and minimum window size parameters of the SensorSAX algorithm based on the average data aggregation and annotation time, CPU consumption, data size, and data reconstruction rate. The present experimental research is a continuation of our previous work [5], which has been extended with an adaptive data aggregation approach in real-time stream processing.

The remainder of the paper is organised as follows. Section 2 demonstrates the proposed framework for stream processing and details the functional components along with the examined data aggregation approaches. Section 3 presents an information model to express summarisation and reliability of data streams. Section 4 details an evaluation of the proposed framework and Section 5 presents a discussion on the results. Section 6 describes the related work both in the semantic annotation and aggregation aspects of stream processing. Section 7 concludes and describes the future work.

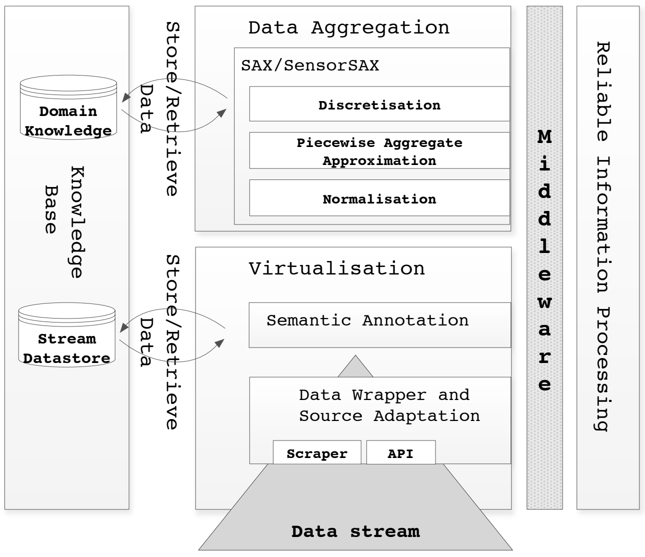


Fig. 1: Real-time IoT stream processing components of the CityPulse smart city framework.

2 IoT Stream Processing Framework

The CityPulse framework provides an infrastructure to address the complex task of stream processing by providing large-scale data analysis and real-time intelligence functionalities. Sensory data streams can have rapid changes due to dynamic environment and resource constraint platforms. Processing and detecting an event in dynamic IoT data streams is a more challenging task compare to the conventional data streams. Using energy efficient adaptive segmentation approach along with information management of IoT data streams are challenging tasks.

We provide energy and process efficient solution combining data aggregation and pattern creation approaches to respond to real-time requests considering resource limitations of devices that provide IoT data. We utilise the domain knowledge that involves the geographical locations of the sensors to interpret the aggregated data streams and detect higher-level events (i.e. machine interpretable or human understandable events) from the data streams. The real-time stream processing framework aims to semantically annotate IoT stream data by taking into account dimensionality reduction and reliability. The framework involves four main units: a) virtualisation, b) middleware, c) data aggregation, and d) reliable information processing. Figure 1 depicts the architecture of the framework.

2.1 Virtualisation

The virtualisation component facilitates access to heterogeneous data sources and infrastructure concealing the technical facets of data streams such as location, storage structure, access format, and streaming technology. The system designates various data wrappers to encompass a large number of input formats, while it provides a unified format as output which is defined in the system. IoT resource virtualisation allows semantic annotation of the resources (e.g. sensors, actuators, data repositories, citizens) in a manner which enables a device to interpret the meaning of observed phenomena.

2. http://www.odaa.dk

2.2 Middleware

There are many solutions that offer communication in distributed systems, such as SOAP [6], REST [7], [8], [9], Web Sockets [10], [11], and HTTP polling [12], [13]. The shortcomings of the existing solutions are combinations of coupling of space (i.e. sender and receiver need to hold references about each other), time (i.e. the components need to interact at the same time) and synchronisation (i.e. the individual components block their activity while waiting for other processes to finish). To overcome these issues we use a publish/subscribe mechanism which decouples time, space and synchronisation. Furthermore the message delivery logic is handled via a message broker, decoupling it from the application layer.

We use the Advanced Message Queue Protocol (AMQP) which has been introduced in [14] as an open standard for message oriented middleware. The protocol divides the message brokering task into exchanges and message queues, whereby the exchange decides which messages will be pushed into which queue. This leads to enhanced flexibility for developers and avoids the need for static implementations. In order to handle scalability issues which arise in the context of IoT and smart city data, we propose aggregating the data before sending it to the middleware. This phenomenon was demonstrated in [4], where it has been pointed out that the data abstractions can help to reduce data traffic and ultimately even energy consumption at the sensory level. The messages in our system have three fields. The first field is “message types”: it defines the type of message. The second field is “metadata”: it contains location and time information as well as information about the data source. The third field is “data”: which consists of the raw values and identifier.

We have defined three types of messages: *transform*, *store* and *forward*. Generally the messages include all the three types. In our case, the subscriber can perform some computations on the data, or store it for later evaluations and then publish the transformed data. For instance, a subscriber can add semantic annotations to the data, while another one performs pre-processing. This approach allows different components to work asynchronously on stream data. The semantic annotations can be instantly accomplished.

β_i \ a	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	0.32	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 1: Breakpoints that divide a Gaussian distribution in an arbitrary number from 3 to 10 of equiprobable regions

2.3 Data Aggregation

2.3.1 Symbolic Aggregate Approximation

SAX is a low cost data aggregation approach [3], which transforms a time-series into a discretised series of letters e.g. a word. It divides a time series data into equal segments and then creates a string representation for each segment. The algorithm involves 3 main steps, namely Normalisation, Piecewise Aggregation Approximation (PAA) and discretising of the aggregated data. Initially, time series data is normalised to obtain average and standard deviation of one before converting it to PAA. The computation of normalisation phase is given as:

$$Z = \frac{x - \mu}{\sigma}, \tag{1}$$

where μ is the mean and σ is the standard deviation of a time series data, x. Afterwards, PAA divides the original data into desired number of windows and calculates the average of data falling into each window. This results in a reduction of data size. A shorter window length n results in a better reconstruction of the original data, however more data space is needed to store the data and eventually it will result in higher energy consumption by higher communication costs. The calculation of each window of PAA segments is given as:

$$\overline{Z_i} = \frac{w}{n} \sum_{j=\frac{n}{w}(i-1)+1}^{\frac{n}{w}i} Z_j, \tag{2}$$

where $\overline{Z_i}$ represents the i-th element of a time series data, Z, of length n, and w represents the number of segments. This results in a reduction of data size from n to n/w data points. Once time series data transformed into PAA coefficients, symbolising the PAA representation into a discrete string is the final stage. Considering the fact that normalised time series data provides a Gaussian distribution, discretisation phase allows to obtain symbolic representation of data by mapping PAA coefficients to breakpoints that are produced according to the alphabet size 'a', which determines equal-sized areas under the Gaussian curve. Table 1 shows the gaussian breakpoints for values of alphabet size, 'a', from 3 to 10. The definition for breakpoints are given below:

Definition 1. (Breakpoints). breakpoints are a sorted list of numbers $B = \beta, ..., \beta_{a-1}$ such that the area under a N(0,1) Gaussian curve from β_i to $\beta_{i+1} = 1/a$, where β_0 and β_a are defined as $-\infty$ and ∞, respectively.

The break lines are distributed vertically according to the Gaussian distribution, the first letter of the alphabet represents the smallest PAA coefficient, 'a', and the greatest PAA coefficient is represented by the last letter of the alphabet. The definition to obtain the symbolic representation is given below:

Definition 2. (Word). A subsequence C of length n can be represented as a word $\hat{C} = \hat{c}_1, ..., \hat{c}_w$ as follows. Let β_i denote the i_{th} element of the alphabet, i.e., β_1 = a and β_2 = b. then the mapping from a PAA approximation $\bar{C}$ to a word $\hat{C}$ is obtained as follows:

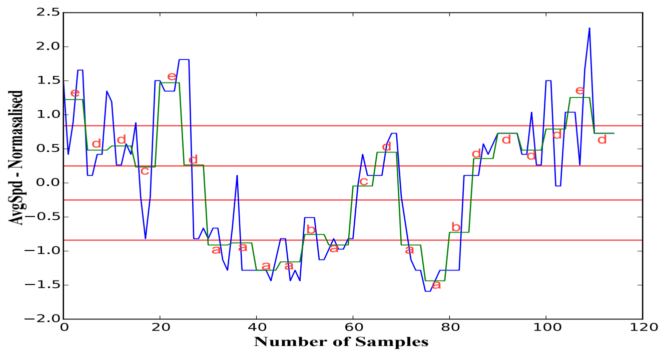
$$\hat{c} = \beta_j, \iff \beta_{j-1} \leq \bar{c} < \beta_j \tag{3}$$

Example 1. Let's assume that we have a time-series data, time-series (c) $= \{2, 3, 4.5, 7.6\}$. Following the steps given above, we apply $z - score$ (i.e. normalise) and obtain time-series (z) =$\{-0.93, -0.52, 0.09, 1.36\}$. Here we use SAX with window size of '2' and alphabet size of '4': it leads to a set of PAA coefficients of $\{-0.72, 0.72\}$. Finally, we map the PAA coefficients into SAX symbols by using the cut off ranges of β and β_{a-1}, given in Table 1, $\{-0.67, 0, 0.67\}$, and obtain corresponding SAX word $\{ad\}$. Given that the first PAA coefficient is smaller than -0.67 and the second coefficient is greater than 0.67, the former assigned to 'a' and latter assigned to 'd'.

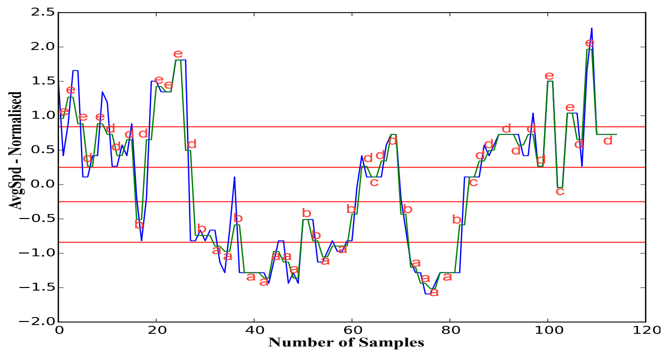
2.3.2 SensorSAX

SensorSAX has been introduced in [4] as an adaptive version of SAX. SAX is highly domain and data dependent. The window size has to be chosen manually and stay fixed in the original approach. Since data coming from IoT streams usually evolves over time, the optimal value for the window length does not remain the same.

On the contrary, SensorSAX uses SensorPAA instead of using original PAA approach. The approach overcomes the problem of finding the right parametrisation for the aggregation algorithm. Instead of having a fixed window size, SensorPAA adapts the window size based on the standard derivation of the data segment. Two boundaries are introduced: a minimum window size to ensure that in highly fluctuating streams there is still data aggregation and a maximum window size. However, the most important parameter is the threshold value, namely sensitivity level (sl), which is compared to the standard deviation (σ) of the data segment processed, to determine the window size based on the variations in the data stream. For that it finds the maximum window size, say w^*, in $w_{min} \leq w* \leq w_{max}$, such that for $1 \leq i \leq w^*$ where $\sigma(c_i) \leq sl$ holds. For such w^*, it computes average $\bar{c}$ of $\{c_1, ..., c_{w^*}\}$ by:



(a) Data aggregation using SAX with window size, 30 minutes, for average speed observation of traffic data stream.



(b) Data aggregation using SensorSAX with minimum window size is 1 and sensitivity level is 0.3 for average speed observation of traffic data stream.

Fig. 2: A real time average speed data obtained from a pair of sensor points is mapped into SAX word with the window size of 30 minutes for SAX, and minimum window size of "1", sensitivity level of "0.30", and word size of "1" for SensorSAX.

$$\bar{c} = \frac{1}{w^*} \sum_{i=1}^{w^*} c_i \tag{4}$$

Then, it finds the breakpoints β_j to obtain a sax letter, $\hat{c}$, such that $\beta_{j-1} \leq \bar{c} < \beta_j$. These letters, $\hat{c}$, form a SAX word, $\hat{C}$. SAX words have got a fix length and allows to have different letters in the same word (e.g. "aabe" with a word length of "4"). The experiments of Ganz *et al.* [4] show that this extension leads to a better reconstruction rate to the original raw data than the original SAX approach, as well as reduced throughput and transmission cost.

However, although the proposed system used a multi-resolution data aggregation approach by finding the optimum window size, w^*, it used a fixed SAX word length, where sensors had to wait until it reaches to a predefined word length regardless of the change in the data and letters. The utilisation of fixed word length can be helpful for an offline system to analyse the patterns in the dataset, but it may critically delay the actuation capacity of a system in a real-time scenario. Moreover, considering the fact that it is possible to have a letter "e" right after a letter "a", sending such a substantial change in the observed environment to the system after reaching to a fixed word length can cause a delay in real-time complex event processing. Moreover, it is also worth to point out that the performance of an aggregation approach can change depending on not only its parameterisations but also variation in data and frequency of data streams. Therefore, we revisit the performance of SAX and SensorSAX and their utilisation in a (near) real-time scenario, where sensors have to instantly report any change to the system for further processing based on the sensitivity level, σ.

Figure 2 depicts the data captured for average speed via the corresponding sensor points and illustrate SAX and SensorSAX patterns created from a data stream that has been collected from a busy road of the city of Aarhus. As can be seen, SensorSAX has managed to obtain a better SAX word representation when the sensitivity level is very high (i.e. small standard deviation), and triggered a new window depending on the change of events. This enabled it to capture a similar shape to the original data stream. On the contrary, since original SAX approach uses a fix window size, it couldn't perform as well when there is a dramatic change in the street. This can be clearly seen when one follows the representation of the data at each peak points given in the graphs.

In listing 1, we describe a set of sensor recordings obtained from the sensor platforms and represent a segment in PAA, shown in Figure 2, as well as temporal entities using the Stream Annotation Ontology. As the proposed semantic model is directly connected to the PROV-O Ontology, we can track the provenance of the information. For instance, in this case the raw data is coming from a public provider, and it has been processed with the stream analysis algorithm SAX, then it has been stored as a stream observation in SAO ontology. This provenance tracking can be used to measure the reliability of the information. With the reliability results, the application developer or the user can make the decision to trust the information or not. We can also annotate QoI concepts, sch as `freshness` of the data, taken from the database field *timestamp*; `availability`, taken from the database field *status*; `granularity`, taken from the database field *VehicleCount*.

2.4 Reliable Information Processing

The dynamic and heterogeneity of IoT environments involve changes and prone to errors in the data, specially when dealing with crowd sourced data. The methods for information extraction and data processing, however, require accuracy and trust issues to be taken into account. This module measures and process accuracy and trust in data acquisition, federation and aggregation. It integrates techniques for monitoring and testing, ensuring reliable information processing. For example, it provides fault tolerance mechanisms when malfunctioning or disappearing sensor are detected, or providing conflict resolution strategies when data analysis result in conflicting information. Provenance also plays an important role in smart cities applications. These applications acquire data from heterogeneous sources, some of them more reliable (e.g. government data), and some of them less reliable (e.g. crowd sourced data). Based on user or application preferences, the application provider could choose to use less reliable data in cases that it has more up to

```
@prefix sao: <http://example.com#> .
@prefix ssn: <http://purl.oclc.org/NET/ssnx/ssn#> .
@prefix qoi: <http://example.com/QoSQoI.owl#> .
@prefix tl: <http://purl.org/NET/c4dm/timeline.owl#> .

:government a foaf:Organisation, prov: Agent .
:sefki a foaf:Person, prov:Agent ;
      foaf:givenName "Sefki" ;
      foaf:mbox <mailto:s.kolozali@surrey.ac.uk>
      prov:actedonBehalfOf :ccsrSurrey ; .
:sensorRec1 a sao:StreamData, ssn:SensorObservation ;
      prov: wasAttributedTo :government .
:traffic-sensor-recording-619 a sao:StreamEvent ;
    prov:used [ a sensorRec1] ;
    sao:time [a tl:Interval;
      tl:at "2014-10-02T08:25:00"^^xsd:dateTime;
      tl:duration "PT30m"^^xsd:duration;
    ] ;
    prov:wasAsscoatedWith :sefki ; .
:freshness-traffic-619 a qoi:Freshness ;
      qoi:value "2014-10-02T08:25:00"^^xsd:dateTime .
:sax_AverageSpeedSample a Piecewise Aggregate Approximation;
      rdfs:label "The paa representation of the traffic sensor
      recording obtained from Aarhus City.";
      sao:value "e";
      sao:alphabetsize "5"^^xsd:int ;
      sao:segmentationsize "6"^^xsd:int ;
      prov:wasGeneratedBy traffic-sensor-recording-619;
      qoi:hasQoI freshness-traffic-619 .
```

Listing 1: An excerpt from an RDF data annotated for a set of sensor recordings given in Figure 2 based on the Stream Annotation Ontology.

date information. The reliable information processing module performs provenance analysis to assert the reliability of the data. In the following section, we describe a data model and semantic annotation approach to include the reliability and provenance information in the stream descriptions.

3 Data Modelling and Semantic Annotation

Smart city applications use data from different stream sources. Therefore the amount of traffic generated by these applications can be voluminous, particularly for real time applications in environments with resource constrains devices, for example sensors with limited bandwidth, memory or power. On the one hand, the proposed data model should be lightweight in order to reduce the traffic and processing time. On the other hand, it should explicitly represent the meaning and relationships of terms in vocabularies. We present a lightweight data model, which uses well-known models to represent IoT data stream. The ontology contains 3 main modules, namely Stream Annotation Ontology (SAO), Quality of Service and Quality of Information (QoS|QoI), and provenance. Figure 3 shows an overview of the proposed information model. Some of this modules has been adapted from the IoT.est model [15]. In the next subsections we describe these three modules.

3.0.1 *SAO module*

The proposed SAO expresses the features of a data stream. It describes content-derived data about IoT streams and provides concepts such as `sao:StreamData`, `sao:Segment`, `sao:SegmentAnalysis` on top of the TimeLine[3] [16] and IoTest models [15]. The SAO uses a broad definition of the `StreamEvent` concept in order to express an artificial classification of a time region, corresponding to a particular stream data. It also extends the sensor observations described in SSN Ontology (`ssn:Observations`) through a concept [17], `sao:StreamData`, that allows to describe `sao:Segment` or `sao:Point` linked to time intervals or time instants. Figure 4 shows the basic structure of the ontology.

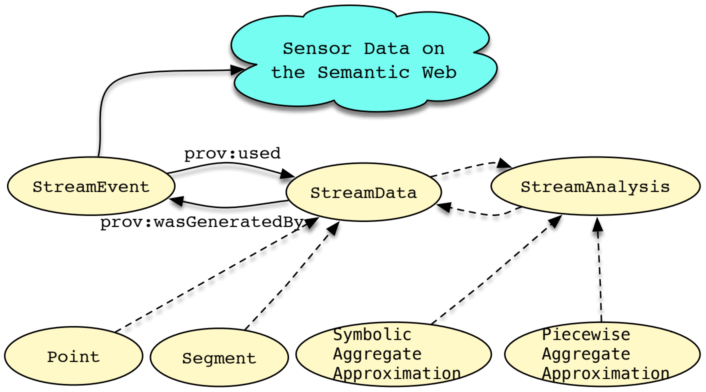

Sensor Data on the Semantic Web
prov:used
StreamEvent
StreamData
StreamAnalysis
prov:wasGeneratedBy
Point
Segment
Symbolic Aggregate Approximation
Piecewise Aggregate Approximation

Fig. 4: Depiction of the main concepts and relationships along with some of the sub concepts that are used in the experiments from the Stream Annotation Ontology.

The dimensionality reduction of data stream or stream transformations obtained through shifted (overlapping) windows can results in a data rate different from the sample rate of the original sensor observation. Using the SAO Ontology, we can describe a data stream and a timeline instance to link the segment description with the time extent of a temporal entity representing the data stream. Thus, we can express a data stream as a time interval on the universal timeline, and also relate such interval with the corresponding interval on the discrete timeline along with its discrete sampling rate. With regards to the previous conceptualisations of sensory data, the SAO ontology deals with representation of aggregated stream data and temporal characteristics. It is free from deep taxonomical structure, and does not attempt to describe data streams the detailed interrelationships or computation of data stream.

3.0.2 *QoS and QoI module*

Quality of Service (QoS) has been widely studied in sensor networks, and has well defined and measurable properties ,such as throughput, jitter or packet loss, inherited from the field of network communications [18]. However, although it has been considered as one crucial item in data networks, the Quality of Information (QoI) is still not well defined and sometimes difficult to measure [19], [20]. In our model, we have designed the QoI module based on the IoT.est model, and enhanced it with some related concepts, as well as with the help of experts in the field of data networks

3. Timeline Ontology extends OWL-Time with various timelines (e.g. universal or discrete), temporal concepts, such as instants and intervals, and interval relationships. Available at: http://motools.sourceforge.net/timeline/timeline.html

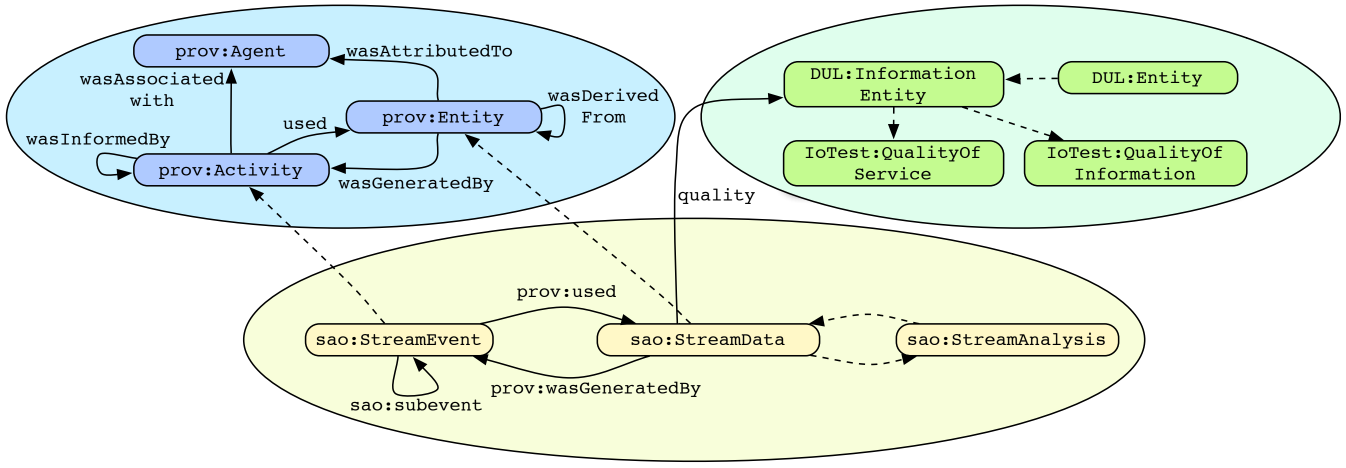


Fig. 3: Describing a stream annotation work flow using the Stream Annotation Ontology.

and data applications. Some of the quality concepts (e.g. `Completeness`, `Age`, `Frequency`) could be directly annotated from the raw stream data and others need data analysis to be quantified. The process is performed in the reliable information processing component of the framework shown in Figure 1. In our data model the provenance module only contains a few object properties that link the SAO module with the prov-o ontology[4]. Provenance annotation helps in tracking the source of information and evaluates trustworthiness of different sources of information. Provenance can also track the algorithms, sampling rate and other useful processing properties. These annotations specifies the reliability of the information and the adequacy of the data for a particular application.

4 Experiments and Evaluations

One of the key issues in heterogeneous ecosystem of smart cities is real-time traffic data analysis. Enabling smart cities to efficiently manage traffic and parking data and provide alternative routes will not only help in reducing transportation cost but also pollution that has been caused by traffic congestion. As a use case scenario, we use public traffic[5] and parking data[6] that has been obtained from the city of Aarhus in Denmark. Our experimental dataset consists of two sets of stream samples: (i) traffic sensor observations (i.e. average speed and vehicle count) (ii) parking sensor observations (vehicle count), which are collected during October 2014. We chose these two sets of sensory data to compare one set with higher sampling frequency with one set at a low frequency. Each traffic sensor has been reported to the system in every 5 minutes, and each parking sensor has been reported to the system approximately every 30 to 35 minutes. In total, there are 449 traffic and 8 parking sensors that are turned into open data. The performance of the stream processing framework is evaluated by changing the settings of SAX and SensorSAX to examine the effects (i.e. change) on the output, namely Time, CPU%, data size, and reconstruction rate (RCR) obtained by using euclidean distance between the original and the reconstructed data for sensory observations, namely average speed (AvgSpd) and vehicle count (VcCnt). In the rest of the paper, we will use the abbreviations of RCR-AvgSpd and RCR-VcCnt for the reconstruction rate of average speed and vehicle count observations, respectively. In addition, we tested the effect of minimum window size (MWS) and sensitivity level (SL) parameters of SensorSAX algorithm in stream processing and aggregation. The level of accuracy estimated by these metrics is analysed utilising a one-way Analysis of Variance (ANOVA). The independent variables were Time (in seconds), CPU percentage, data size (in MegaBtye), and reconstruction rate of sensory observations. The overall results computed by taking average of independent variables of traffic and parking sensors, such as average time that has been taken for aggregation and annotation of a data stream in 1 month duration. The Holm-Sidak procedure [21] and a risk α of .05 were used in the ANOVA tests. In addition, we have used the following definitions in our interpretations of the effect sizes: small effect size ($\eta^2 \leq .01$), medium effect size ($.01 \leq \eta^2 \leq .06$) and large effect size ($.06 \leq \eta^2 \leq .14$). ANOVA level of significance are reported using the F-statistics, F, and probability, p. The evaluations were performed on a Personal Computer (PC) running Ubtuntu 14.04 operating system with an Intel Core i5-3470 3.2GHz processor and 8GB RAM memory.

4.1 Results of the Data Aggregation Process

The performance of 4 SAX models and 48 SensorSAX models with different parameters for 1 month of traffic and parking data streams are reported based on ANOVA in Table 2 and based on descriptive analysis in Table 3. For the sake comprehensive analysis of SensorSAX, we used greater number of parameters (i.e. 12 different SL and 4 different MWS parameters, 48 in total) to examine its effects in a dynamic environment. The system performed the annotation of raw sensory observation in average 33.3 seconds with 6.6% CPU usage and 12.7MB data size for each traffic data stream, and 86.8 seconds with 13.4% CPU usage and 1.8MB

4. http://www.w3.org/TR/prov-o/
5. The data available at: http://www.odaa.dk/dataset/realtids-trafikdata
6. The data available at: http://www.odaa.dk/dataset/parkeringshuse-i-aarhus

data size for annotation of the raw sensory observations with no aggregation process for each parking data stream. Figure 5 depicts the average results of Time, CPU%, DS, and RCR - Avg Spd and RCR - VcCnt for both data streams.

		Traffic Observations			Parking Observations		
Source	Dependent Variable	df	F	η^2	df	F	η^2
Aggregation Method	Time	52	1972.1	$0.8^{\star\star\star}$	52	3371.8	$1.0^{\star\star\star}$
	CPU%	52	2329.44	$0.8^{\star\star\star}$	52	17.8	$0.6^{\star\star\star}$
	Data Size	52	1844.7	$0.8^{\star\star\star}$	52	95.3	$0.9^{\star\star\star}$
	RCR - AvgSpd	52	436.3	$0.5^{\star\star\star}$	-	-	-
	RCR - VcCnt	52	572.7	$0.6^{\star\star\star}$	52	7.1	$0.4^{\star\star\star}$

TABLE 2: Results of one-way analyses of variance for the stream annotation system based on the raw and SAX stream data. η^2 is the partial eta squared measure of effect size. Significance level: ${}^{\star}p < .05, {}^{\star\star}p < .01, {}^{\star\star\star}p < .001$. Time: aggregation and annotation time, CPU%: CPU consumption, Data Size: size of aggregated data, RCR-AvgSpd: error in reconstruction rate for average speed observation, RCR-VcCnt: error in reconstruction rate for vehicle count observation.

4.1.1 TIME

The results has shown that there was highly significant effect of time factor with a very large effect size for both traffic and parking data stream ($p < .001$, $\eta^2 = .08$ and $\eta^2 = 1.0$ respectively) in Table 2. The average performance of the system for aggregation and annotation of the sensory observation was in range of 12.9 and 4.2 seconds for traffic data stream and in range of 2.7 and 10.0 seconds for parking data stream. For SensorSAX, the average performance of the system was in range of 46.9 and 9.1 seconds for the traffic data streams and in range of 10.5 and 7.5 seconds for the parking data streams. At first glance to the results in Table 3, it is possible to see that there was a highly significant change in terms of time cost for aggregation and annotation with SAX on traffic and parking data streams. While there was highly significant effect on the majority of the SensorSAX parameters for the traffic data streams, only a few cases showed a highly significant change for the parking data streams. For the traffic data stream performance of SensorSAX, the results indicate that highly effect was slightly decreased to significant effect with MWS=6 – SL= 0.6 ($p < .01$), and no effect found with MWS= 4 – SL<= 0.9 and MWS=6 – SL> 0.6

As can be seen in Figure 5 and Table 3, for the performance of SensorSAX on parking data, there was a decay curve in the effect of the time factor at MWS=1, where it begins with a high significant effect for cases of SL<= 0.1 ($p < .001$), followed by a period of moderate decrease to significant effect between 0.2 <=SL<= 0.3 ($p < .01$), and then to a period of small significant effect between 0.4 <=SL<= 0.6 ($p < .05$). Additionally, there as highly significant effects for MWS=2 – SL<= 0.05, significant effects for MWS=2 – SL<= 0.05 – SL= 0.1 and MWS=4 – SL<= 0.01 ($p < .01$), and small significant effects for MWS=4 – SL<= 0.05 and MWS=6 – SL<= 0.01 ($p < .05$).

While the obtained dramatic decrease in the results of SAX and SensorSAX can be explained by the fact that data aggregation process reduces the amount of time that is spent on the data annotation, it is evident that the small numbers in the SensorSAX parameters of MWS and SL increase the amount of time that is spent on stream processing task up to a level, where it led to even a longer time than the annotation of raw data as we can see in the case of SensorSAX with MWS=1 and 0.01 <=SL<= 0.3. It is worth to point out that although the sampling frequency of the parking data stream is in every half an hour in average, which is much less frequent than the traffic data streams, since there is no fixed sampling frequency, we had to make more frequent calls to fetch the raw data for the corresponding time period for the parking data. This programming factor did not show any effect on the other factors (i.e. CPU%, Data Size, RCR) of the aggregation methods due to the fact that either segmentation time was fixed or incremental.

4.1.2 CPU%

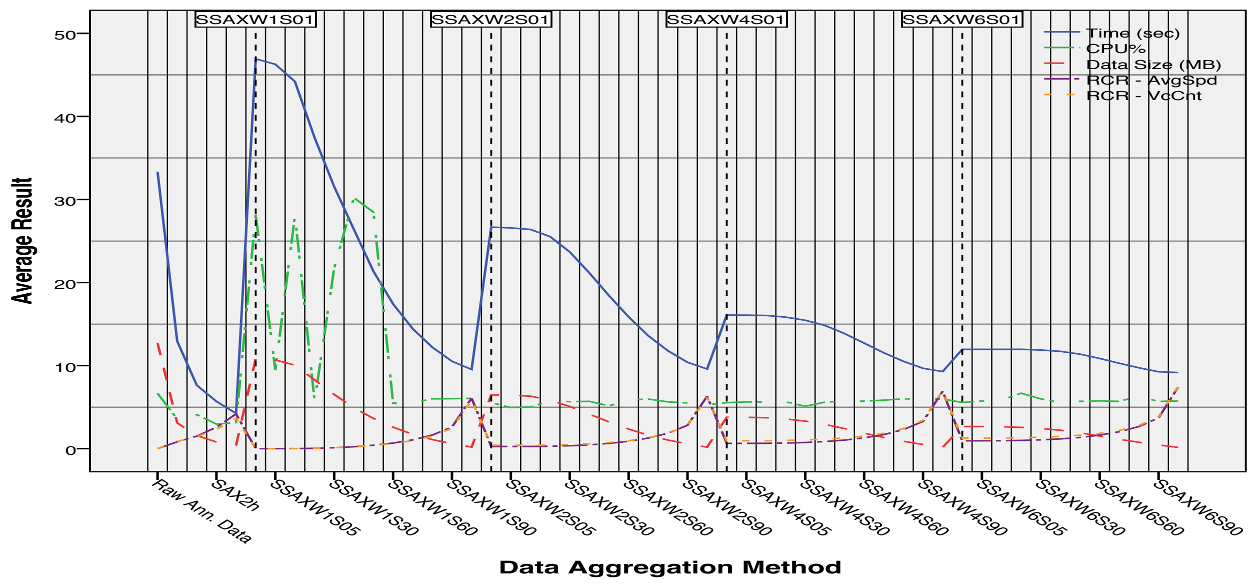
Highly significant effect was found with a very large effect size for both traffic and parking data streams in terms of CPU consumption ($p < .001$, $\eta^2 = .08$ and $\eta^2 = 0.6$ respectively). While there was highly significant effect of the SAX performance on the CPU% consumption for the traffic data stream, no significant difference was found for the parking data streams, except the small significant effect at WS=1h. On the contrary, the results indicate that there was only a few cases where there was a highly significant effect on the CPU% for traffic data streams — which were MWS=1 with 0.01 <=SL<= 0.5, MWS=2 with 0.05<=SL<=0.1, MWS = 4 – SL=0.3, MWS=6 – SL=0.2 ($p < .05$) — and highly significant effects found on the majority of the SensorSAX cases for the parking data streams ($1 <$ MWS< 2 with 0.01 <=SL<= 1.00, and MWS=4 with 0.01 <=SL<= 0.7, except MWS=6. The computational cost of SensorSAX with small MWS and SL values seems to have high computational cost.

For the traffic data streams, a few number of cases with high CPU% usage can be interpreted as utilisation of small numbers in MWS and SL forms a highly sensitive system, where it adds upon the computational cost of the raw data annotation with aggregation method, and causes to consume a larger percentage of the CPU. This simple fact also reflected on the Time factor. On the other hand, the high CPU% consumption for the parking data streams can be simply explained by the fact that parking sensors are reported to the system with a very low frequency (i.e. in every half an hour) and they have got a lower variance compare to the traffic sensors. Evidently, the reflection of this fact can be also seen in the data size results, where there is a great deal of similarity among different sensitivity levels.

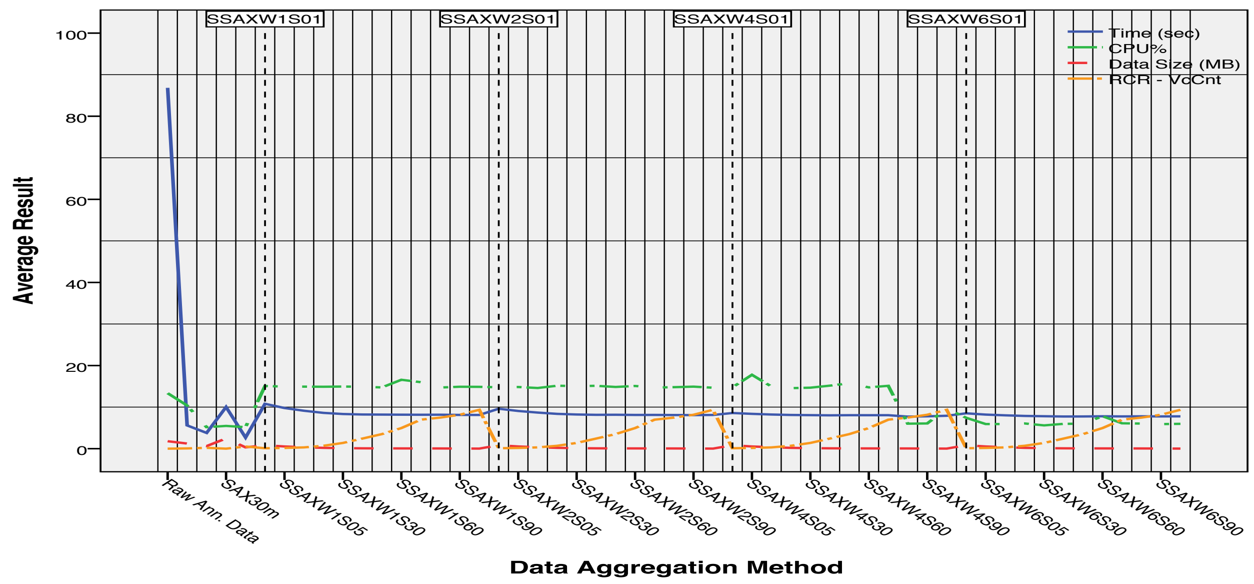
Overall, there are only a few cases that CPU% of the aggregation methods are higher than the raw data annotation, and majority of the cases are very close to the raw annotation. Although it is desirable to reduce the computational cost, it is worth to note that the general aim in sensory devices is to reduce the communication overhead. The power consumption of computation in sensory devices is usually significantly lower than data transmission, as given in [1], [2]. Thus, utilisation of a close CPU% value to the raw data annotation can be foreseen for IoT applications.

4.1.3 Data size

We found highly significant effect and a large effect size of the aggregation methods for both the traffic and parking



(a) The average performance measurements of raw data annotation process, SAX with annotation process, SensorSAX with annotation process computed based on different factors across the traffic data streams.



(b) The average performance measurements of raw data annotation process, SAX with annotation process, SensorSAX with annotation process computed based on different factors across the parking data streams.

Fig. 5: Comparison between the average time in seconds (Time) measured (solid line in blue), average CPU% (dashed green line), average data size in MegaByte (dashed red line), reconstruction rate of average speed (RCR - AvgSpd) observation (dashed purple line) and average reconstruction rate of vehicle count (RCR - VcCnt) (dashed orange line) based on Euclidean distance measured for traffic data streams. The dashed vertical lines correspond to the different window size for SensorSAX. The average performance of each factor for the annotation of raw data, SAX with annotation process, and SensorSAX with annotation process computed across the traffic data streams are reported in Figure 5a and parking data streams are reported in Figure 5b. Each vertical rectangle that is given on the x-axis represents a data aggregation model with a different parameterisation. The first dashed vertical line represents the transition from SAX to SensorSAX approach, where the rest of the dashed vertical lines represents the change of minimum window size parameter of the SensorSAX approach.

data streams ($p < .001$, $\eta^2 = .08$ and $\eta^2 = 0.9$ respectively). For the traffic data streams, highly significant effects on the data size was seen frequently for most of the SensorSAX parameters 1 <=WS<= 6 and 0.01 <=SL <= 0.8 ($p < 0.001$), no effects on SL= 0.9, and again significant effects with SL= 1.00 ($p < 0.01$). However, although there was highly significant effects in the data size with SAX - $30m$ <=WS<= $2h$ ($p < 0.001$), it is important to note that SAX with WS=4h did not show any significant effect on the data reduction for the traffic data streams, which was lower than it was expected beforehand. On the contrary, for the parking data streams there was only a small effect of

SAX with WS=4h ($p < 0.05$) and no effect for other SAX parameters on the parking data streams. It is suggested that this could be due to the very low sampling frequency of data stream.

It is worth to point out that the results have been obtained by simulating the data aggregation methods and creating an RDF document for each aggregated time-series segment as it is assumed to be sent to the other components via the message bus. Overall, while the average annotated data for 1 month observations of each sensor was 12.7MB for traffic data streams and 1.8MB for parking data steams, SAX methods managed to reduce the data size within the range of 3.1MB and 0.4MB and in range of 2.5MB and 0.3MB for the traffic data streams, and SensorSAX obtained results in range of 10.8MB and 0.1MB for the traffic data streams and in range of 0.8MB and $0.1E-1$MB for the parking data streams. Compare to SAX algorithm, SensorSAX possesses more parameters, which increases the number of triples in RDF documents, and it affects the overall data size. Considering the fact that data that is annotated for 1 month of traffic observations is approximately 250M triples, it is apparent that the obtained experimental results for the aggregated data streams are important for real-time IoT stream processing.

4.1.4 Reconstruction Rate

The experiments showed that there were highly significant effects with a very large effect sizes for the error in RCR-AvgSpd and RCR-VcCnt observations of the traffic data streams ($p < .001$, $\eta^2 = 0.5$ and $\eta^2 = 0.6$), and RCR-VcCnt observation of the parking data streams ($p < .001$, $\eta^2 = .04$). Reconstruction rate results of SAX and SensorSAX ($p < .001$) showed a highly significant effect for both the AvgSpd and VcCnt sensor observations for the traffic data streams, except SAX with WS=4h, where there was no effect for the VcCnt, and SensorSAX with $3 <=$MWS$<= 4$ and SL$= 0.9$ for the AvgSpd. There was significant effects on the former and no effect on the latter.

On the contrary, there is a noticeable difference in the reconstruction rate of the VcCnt for the parking data streams, which can be interpreted in three categories: in the first category, there were highly significant results for the SensorSAX with $1 <=$WMS$<= 6$ and $0.01 <=$SL$<= 0.2$ ($p < .001$), then the results followed by significant effect for $0.3 <=$SL$<= 0.4$ including WMS$= 1$ with SL$= 0.5$ ($p < .01$), and small significant effects for most of the cases of SL$= 0.5$ ($p < .05$), and no significant effect was found for $0.6 <=$SL$<= 1.00$, an exception of WMS$= 1$ with SL$= 0.6$ ($p < .05$).

Although it is acceptable to see substantially different results for the cases where $0.9 <=$SL$<= 1.00$ for the SensorSAX, there were a few cases with no effect of the SL$= 0.9$ and a small or significant effect of the SL$= 1.00$. However, it is clear that additional work is required before a complete understanding can be reached for these cases.

4.2 Influence of the SensorSAX parameters

We have used a two way analysis of variance to obtain detailed performance measurement of the SensorSAX. The results of the ANOVA test is given in Table 4. Figure 6 depicts the overall average results for the sensitivity and minimum window size parameters.

		Traffic Observations			Parking Observations		
Source	Dependent Variable	df	F	η^2	df	F	η^2
SL	Time	11	5446.8	$0.7^{\star\star\star}$	11	91.6	$0.8^{\star\star\star}$
	CPU%	11	1653.2	$0.5^{\star\star\star}$	11	12.4	$0.3^{\star\star\star}$
	DS	11	5954.18	$0.8^{\star\star\star}$	11	101.7	$0.8^{\star\star\star}$
	RCR - AvgSpd	11	1565.8	$0.5^{\star\star\star}$	—	-	-
	RCR - Vc Cnt	11	1999.2	$0.5^{\star\star\star}$	11	20.9	$0.4^{\star\star\star}$
MWS	Time	3	18575.2	$0.7^{\star\star\star}$	3	163.5	$0.6^{\star\star\star}$
	CPU%	3	10744.2	$0.6^{\star\star\star}$	3	570.1	$0.8^{\star\star\star}$
	DS	3	7854.3	$0.5^{\star\star\star}$	3	3371.8	0
	RCR - AvgSpd	3	256.3	$0.03^{\star\star\star}$	—	-	—
	RCR - Vc Cnt	3	591.9	$0.08^{\star\star\star}$	3	0	0
SL × MWS	Time	32	1297.2	$0.7^{\star\star\star}$	33	10.9	$0.5^{\star\star\star}$
	CPU%	32	1829.6	$0.7^{\star\star\star}$	33	9.9	$0.5^{\star\star\star}$
	DS	32	682.9	$0.5^{\star\star\star}$	33	0	0
	RCR - AvgSpd	32	0.4	0.01	-	-	-
	RCR - Vc Cnt	32	0.6	0.01	33	0	0

TABLE 4: Results of two-way analyses of variance for the data aggregation of traffic and parking data streams. η^2 is the partial eta squared measure of effect size. Significance level: $^{\star}p < .05$,$^{\star\star}p < .01$,$^{\star\star\star}p < .001$. SL: Sensitivity Level; MWS: Minimum Window Size.

4.2.1 Sensitivity

We found highly significant and large effects of the SL on all of the measurements, namely TIME, CPU%, DS, RCR - AvgSpd and RCR - VcCnt for both the traffic and parking data streams ($p < .001$, η^2 was in range of 0.3 and 0.8).

For the Time factor, overall results were lower than the annotation of raw data, which was 33.3 seconds. However, the results indicate that the performance was lower with MWS=1: it started with longer time than the annotation of raw data and continued in the same way until SL$<= 0.2$, and followed by a sharp decrease in the results from 31.5 down to 9.5 seconds. The results showed that the aggregation and annotation process took shorter time period than the annotation of raw data for the rest of the parameter settings of the SensorSAX. For the parking data streams, there was a smoother decrease and all of the time measurements were below time for the annotation of the raw data.

In terms of CPU% consumption, it was possible to see a similar pattern for the parking data stream that the results were higher than the annotation of the raw data until SL$<= 0.2$. On the contrary, we obtained lower CPU% only after SL$>= 0.6$, except the anomaly at SL$= 02$. This can be explained by the fact that the high CPU% occurred with MWS=1, which has effected the overall results.

Additionally, for the RCR, it is apparent that the reconstruction rate shows a smoothly increasing curve after SL$= 0.3$ for the traffic data streams, whereas there was a sharper curve for the parking data streams. While it was apparent beforehand that there would be a smooth curve in the overall error of RCR. On the contrary, It was expected to have the intersection point between Data Size × RCR for the SensorSAX parameters would be between 0.1 and 0.3, but it was interesting to have such an intersection at SL$= 0.7$ for the SensorSAX results.

4.2.2 Minimum Window Size

Highly significant effects of the MWS was found on all of the results for the traffic data streams ($p < .001$). While there was very large effect size for Time, CPU% and DS factors (η^2 was in range of 0.5 and 0.7), it was large effect size for the RCR - AvgSpd ($\eta^2 = 0.08$) and medium

SAX		SensorSAX												
WS	-	MWS	0.01	0.05	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.00
30m	12.9^{***}	1	46.9^{***}	46.2^{***}	44.2^{***}	37.5^{***}	31.5^{***}	26.4^{***}	21.3^{***}	17.4^{***}	14.4^{***}	12.2^{***}	10.5^{***}	9.5
1h	7.6^{***}	2	26.7^{***}	26.6^{***}	26.4^{***}	25.5^{***}	23.7^{***}	21.2^{***}	18.4^{***}	15.9^{***}	13.6^{***}	11.8^{***}	10.4^{***}	9.6
2h	5.7^{***}	4	16.1^{***}	16.1^{***}	16.0^{***}	15.8^{***}	15.5^{***}	14.8^{***}	13.9^{***}	12.7^{***}	11.5^{***}	10.5^{***}	9.7	9.3
4h	4.2^{***}	6	11.9^{***}	11.9^{***}	11.9^{***}	11.9^{***}	11.8^{***}	11.7^{***}	11.3^{***}	10.8^{**}	10.3	9.7	9.3	9.1

(a) Traffic data streams: the average performance measurements based on Time factor in seconds. The average time spent for raw data annotation process was 33.3 seconds.

SAX		SensorSAX												
WS	-	MWS	0.01	0.05	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.00
30m	3.6^{***}	1	28.2^{***}	9.5^{***}	28.1^{***}	5.7	21.7^{***}	30.2^{***}	28.5^{***}	5.5	5.5	5.9	6.0	6.1
1h	4.1^{***}	2	5.5	4.9^{***}	5.0^{***}	5.8	5.7	5.7	5.2^{**}	5.9	6.0	5.6	5.5	5.3^{*}
2h	2.9^{***}	4	5.5	5.6	5.6	5.6	5.1^{***}	5.6	5.7	5.7	5.8	6.0	6.0	6.0
4h	3.5^{***}	6	5.6	5.7	5.8	6.7^{***}	6.0	5.6	5.7	5.8	5.7	6.3^{**}	5.7	5.7

(b) Traffic data streams: the average performance measurements based on CPU% consumption factor. The average CPU% consumption spent for raw data annotation process was 6.6%.

SAX		SensorSAX												
WS	-	MWS	0.01	0.05	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.00
30m	3.1^{***}	1	10.8^{***}	10.7^{***}	10.0^{***}	8.3^{***}	6.5^{***}	4.9^{***}	3.6^{***}	2.6^{***}	1.7^{***}	1.1^{***}	0.6	0.2^{**}
1h	1.6^{***}	2	6.5^{***}	6.4^{***}	6.3^{***}	5.8^{***}	5.1^{***}	4.2^{***}	3.2^{***}	2.4^{***}	1.6^{***}	1.0^{***}	0.6	0.2^{**}
2h	0.8^{***}	4	3.8^{***}	3.8^{***}	3.7^{***}	3.6^{***}	3.3^{***}	2.9^{***}	2.4^{***}	1.8^{***}	1.3^{***}	0.9^{***}	0.5	0.2^{**}
4h	0.4	6	2.7^{***}	2.7^{***}	2.6^{***}	2.6^{***}	2.4^{***}	2.2^{***}	1.9^{***}	1.5^{***}	1.1^{***}	0.8^{**}	0.4	0.1^{**}

(c) Traffic data streams: the average performance measurements based on data size factor in MegaBtye. The average data size for raw data annotation process was 12.7MB.

SAX		SensorSAX												
WS	-	MWS	0.01	0.05	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.00
30m	0.8^{***}	1	**0.4E-4*****	**0.6E-2*****	**0.7E-2*****	**0.4E-1*****	0.1^{***}	0.2^{***}	0.4^{***}	0.7^{***}	1.1^{***}	1.6^{***}	2.6^{***}	6.1^{***}
1h	1.5^{***}	2	0.3^{***}	0.3^{***}	0.3^{***}	0.3^{***}	0.3^{***}	0.5^{***}	0.6^{***}	0.9^{***}	1.3^{***}	1.9^{***}	2.8^{***}	6.3^{***}
2h	2.6^{***}	4	0.6^{***}	0.6^{***}	0.6^{***}	0.7^{***}	0.7^{***}	0.9^{***}	1.0^{***}	1.3^{***}	1.6^{***}	2.3^{***}	3.2^{**}	6.9^{***}
4h	4.2^{***}	6	0.9^{***}	0.9^{***}	0.9^{***}	1.0^{***}	1.1^{***}	1.2^{***}	1.3^{***}	1.6^{***}	2.0^{***}	2.6^{***}	3.7	7.4^{***}

(d) Traffic data streams: the average performance measurements based on reconstruction rate of average speed computed by euclidean distance.

SAX		SensorSAX												
WS	-	MWS	0.01	0.05	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.00
30m	0.9^{***}	1	**0*****	**0.3E-3*****	**0.4E-2*****	**0.4E-1*****	0.1^{***}	0.2^{***}	0.4^{***}	0.6^{***}	1.0^{***}	1.5^{***}	2.4^{***}	5.8^{***}
1h	1.5^{***}	2	0.4^{***}	0.4^{***}	0.4^{***}	0.4^{***}	0.5^{***}	0.6^{***}	0.7^{***}	0.9^{***}	1.3^{***}	1.8^{***}	2.8^{***}	6.2^{***}
2h	2.3^{***}	4	0.9^{***}	1.0^{***}	1.0^{***}	1.0^{***}	1.1^{***}	1.1^{***}	1.3^{***}	1.5^{***}	1.8^{***}	2.4^{***}	3.4^{***}	6.9^{***}
4h	3.7	6	1.3^{***}	1.3^{***}	1.3^{***}	1.3^{***}	1.3^{***}	1.5^{***}	1.6^{***}	1.8^{***}	2.1^{***}	2.7^{***}	3.8^{***}	7.5^{***}

(e) Traffic data streams: the average performance measurements based on reconstruction rate of vehicle count computed by euclidean distance.

SAX		SensorSAX												
WS	-	MWS	0.01	0.05	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.00
30m	10.0^{***}	1	10.8^{***}	9.8^{***}	9.1^{***}	8.6^{**}	8.3^{**}	8.2^{*}	8.2^{*}	8.2^{*}	8.2	8.2	8.1	8.1
1h	5.6^{***}	2	9.6^{***}	9.1^{***}	8.7^{**}	8.4^{*}	8.2	8.1	8.2	8.1	8.1	8.1	8.1	8.1
2h	3.8^{***}	4	8.6^{**}	8.4^{*}	8.2	8.1	8.1	8.0	8.1	8.1	8.1	7.7	7.8	7.9
4h	2.7^{***}	6	8.5^{*}	8.2	8.0	7.9	7.8	7.7	7.8	7.8	7.8	7.8	7.8	7.8

(f) Parking data streams: the average performance measurements based on Time factor in seconds. The average time spent for raw data annotation process was 86.8 seconds.

SAX		SensorSAX												
WS	-	MWS	0.01	0.05	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.00
30m	5.5	1	15.1^{***}	14.9^{***}	14.9^{***}	14.9^{***}	12.8^{***}	12.9^{***}	12.8^{***}	15.0^{***}	13.4^{***}	14.7^{***}	14.9^{***}	14.9^{***}
1h	10.3^{*}	2	14.8^{***}	14.8^{***}	14.6^{***}	15.2^{***}	15.1^{***}	15.1^{***}	14.9^{***}	15.1^{***}	14.7^{***}	14.8^{***}	14.9^{***}	14.7^{***}
2h	5.2	4	14.8^{***}	17.8^{***}	15.0^{***}	14.6^{***}	14.7^{***}	15.2^{***}	15.7^{***}	14.8^{***}	15.1^{***}	6.0	6.1	9.7^{*}
4h	5.1	6	7.4	5.9	5.9	6.1	5.6	6.0	6.0	7.8	6.1	6.0	5.9	6.0

(g) Parking data streams: the average performance measurements based on CPU% consumption factor. The average CPU% consumption spent for raw data annotation process was 13.4%.

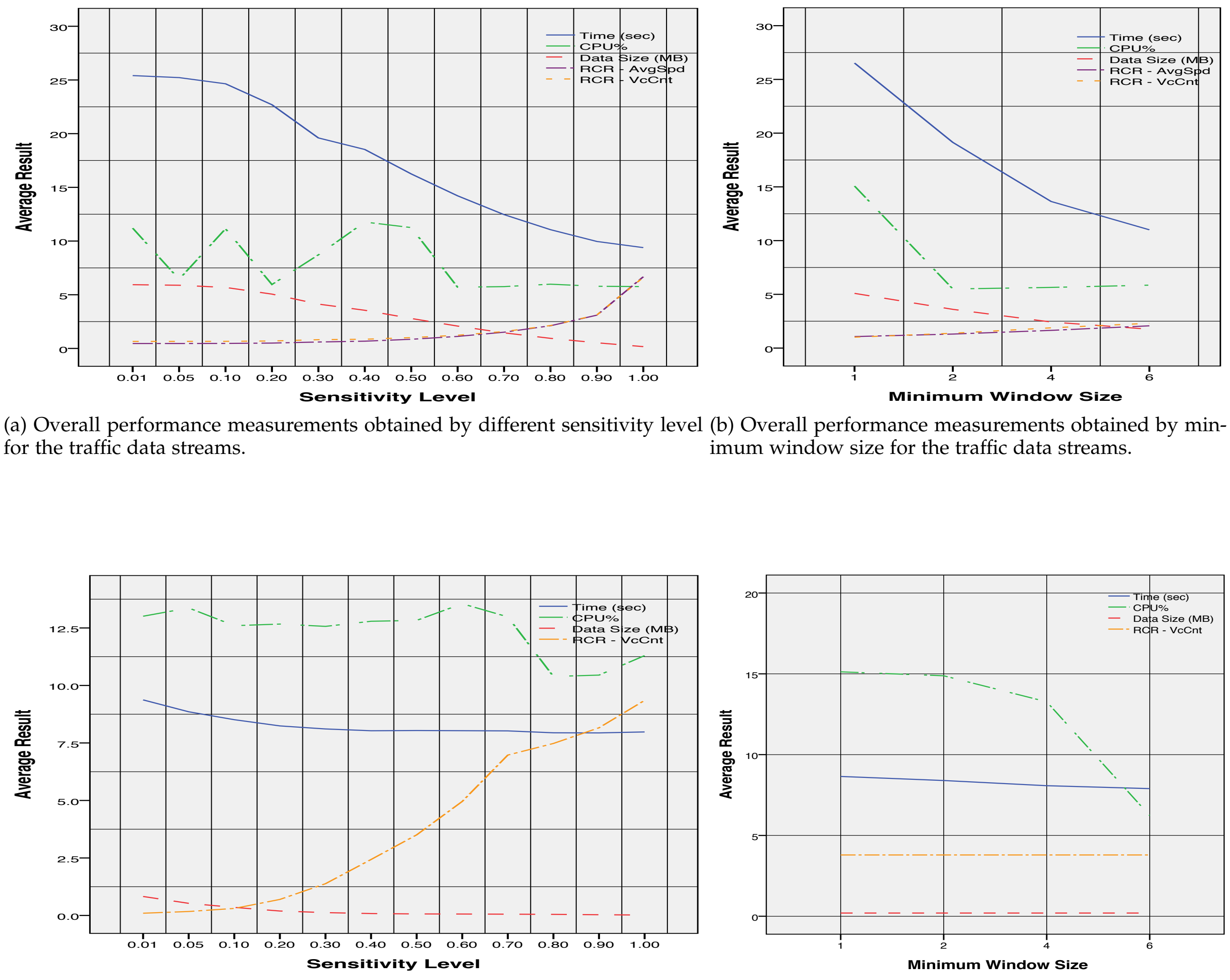
SAX		SensorSAX												
WS	-	MWS	0.01	0.05	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.00
30m	2.5	1	0.8^{***}	0.5^{***}	0.4^{***}	0.2^{*}	0.1^{*}	0.1	0.1	0.1	0.1	0.1	$0.3E-1$	$0.2E-1$
1h	1.2	2	0.8^{***}	0.5^{***}	0.4^{***}	0.2^{*}	0.1	0.1	0.1	0.1	0.1	0.1	$0.3E-1$	$0.2E-1$
2h	0.6	4	0.8^{***}	0.5^{***}	0.4^{***}	$0.1E-1^{*}$	0.2	0.1	0.1	0.1	0.1	0.1	0.1	$0.3E-1$
4h	0.3^{*}	6	0.8^{***}	0.5^{***}	0.4^{***}	0.2^{*}	0.1	0.1	0.1	0.1	0.1	0.1	$0.3E-1$	$0.1E-1$

(h) Parking data streams: the average performance measurements based on data size factor in MegaBtye. The average data size for raw data annotation process was 1.8MB.

SAX		SensorSAX												
WS	-	MWS	0.01	0.05	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.00
30m	$0.5E-1^{***}$	1	0.1^{***}	0.2^{***}	0.3^{***}	0.7^{***}	1.1^{***}	2.1^{***}	3.4^{**}	4.7^{*}	6.8	7.5	8.2	9.4
1h	$0.6E-1^{***}$	2	0.1^{***}	0.2^{***}	0.3^{***}	0.7^{***}	1.4^{**}	2.4^{**}	3.5^{*}	4.9	7.0	7.5	8.2	9.4
2h	0.2^{***}	4	0.1^{***}	0.2^{***}	0.3^{***}	0.7^{***}	1.4^{**}	2.4^{**}	3.5^{*}	4.9	7.0	7.5	8.2	9.4
4h	0.5^{***}	6	0.1^{***}	0.2^{***}	0.3^{***}	0.7^{***}	1.4^{**}	2.4^{**}	3.5^{*}	4.9	7.0	7.5	8.2	9.4

(i) Parking data streams: the average performance measurements based on reconstruction rate of vehicle count computed by euclidean distance.

TABLE 3: The window size and minimum window size parameters are denoted as WS and MWS, respectively. Results represent average results for the performance of SAX and SensorSAX algorithms in terms of Time, CPU%, Data Size, and Reconstruction Rate of average speed and vehicle count observations for traffic and parking data streams, which are given in Figure 5. Significance level: $*p < .05$, $**p < .01$, $***p < .001$.



(a) Overall performance measurements obtained by different sensitivity level for the traffic data streams.

(b) Overall performance measurements obtained by minimum window size for the traffic data streams.

(c) Overall performance measurements obtained by different sensitivity level for the parking data streams.

(d) Overall performance measurements obtained by minimum window size for the parking data streams.

Fig. 6: Comparisons between the overall average performance measurements of sensitivity level (SL) and minimum window size (MWS) parameters of SensorSAX in terms of the average time in seconds (Time) measured (solid line in blue), average CPU% (dashed green line), average data size in MegaByte (dashed red line), reconstruction rate of average speed (RCR - AvgSpd) observation (dashed purple line) and average reconstruction rate of vehicle count (RCR - VcCnt) (dashed orange line) based on Euclidean distance measured depicted in Figure 6a and Figure 6b for traffic data streams, respectively – and in Figure 6c and Figure 6d for parking data streams, respectively. The annotation of raw data streams are not shown.

effect size for the RCR - VcCnt ($\eta^2 = 0.03$). There was highly significant effect solely with very large effect size on Time and CPU% consumption for the parking data streams ($p < .001$, $\eta^2 = 0.6$ and $\eta^2 = 0.8$, respectively). Figure 6b shows a fairly consistent results with the statistical analysis results. While the significant effect in all results of the traffic data stream can be seen in Figure 6b, it can be observed from Figure 6d that although there was highly significant effect of TIME, it has shown a small and steady decrease in the results of parking data streams. Similar behaviour was observed in the case of DS and RCR, with no sudden changes. Having no significant effect on the DS and RCR can be interpreted as while MWS is a very effective approach for a highly frequent data, it may loose its effect at lower frequencies. This is simply because of the fact that having a low sampling frequency means, data represents a larger time period, which is more likely to have a bigger variance in the data. Therefore, the MWS effect was lost at a lower frequency.

4.2.3 Interaction between sensitivity and minimum window size

The interaction between SL and MWS shown highly significant effect with very large effect size on the TIME, CPU%, and DS factors ($p < .001$, η^2 in range of 0.5 and 0.7) and TIME and CPU% for the parking data streams

($p < .001$, $\eta^2 = 0.5$ for both of them) . On the contrary, there was no effect of this interaction on the DS factor of the parking data streams. It is evident that the interaction did not show any significant effect on the RCR of neither stream types.

5 Discussions

In our experiments, we used traffic and parking data streams, which report data from the same town, with different sampling frequency and data variation. We found that in most of the cases having a change in parameterisation of SAX and SensorSAX caused a significant change not only in the results of data size and reconstruction rate, but also CPU% and processing time. When we take into account all the intersection points for data aggregation processes given in Figure 5, the highest performance were obtained for SAX with WS=1h and SensorSAX with SL= 0.7 and MWS= 2. While in this particular case, SAX performed better than SensorSAX in terms of processing time, it was evident that SensorSAX managed to reduce the size of data to the same level by decreasing the error in reconstruction rate by a factor of 0.2 in euclidean distance. Although this result seems to be a small difference, it should be taken into account that SensorSAX annotation involves an additional triple for its parameters, which can lead to a larger data reduction size for a month of data. With regards to the Time factor, SensorSAX was yielded lower performance with most of the SL and MWS parameters.

We found that the intersection points given in Figure 5 depict the best performance in terms of all factors. However, when we further examined the results for each sensor individually, it is possible to see that there is a large variation in the results of SensorSAX. Figure 7b depicts the average and standard deviation of the sensory observations that we used in the experiments and Figure 7a shows the comparison between all SAX models and one of the best SensorSAX models in terms of reconstruction rate (i.e. RCR-AvgSpd: $0.7E^{-2}$ and RCR - VcCnt: $0.4E^{-2}$), which couldn't perform as well in terms of data reduction (i.e. data size: 10MB). It can be seen from this particular comparison that the results obtained by SAX models for traffic sensors are very sparse for the reconstruction rate. This can be explained by the fact that utilisation of fixed window size for data stream which is high in variance can dramatically reduce the data compression quality. In the meantime, it carried out a constant behaviour for data reduction regardless of geographical location and environmental changes of the sensors. On the other hand, it can be seen that SensorSAX obtained very sparse results for data reduction. Despite having a better performance with SAX for the majority of sensors, the results indicate that SensorSAX has clearly outperformed most of the SAX models in terms of both data reduction and reconstruction rate for some of the sensors in this particular case. When we further investigated this issue, we found out that this was due to the fact that the sensors closer to the city centre have got a larger data variation compare to the ones in the outskirts of the city. Since SensorSAX is an adaptive segmentation approach and triggered based on events, it performs excellent with small parameterisation in terms of reconstruction rate in high data variations, but it cannot perform as well in high data variations in terms of data reduction, which occurred in city centre in the experiment. data variation in the city. Figure 8 depicts the sensors that are located in the city of Aarhus, and one of the sensors that has been installed outskirts of the town centre, which obtained excellent results both in terms of reconstruction rate and data reduction with small parameters of SensorSAX model, SensorSAX MWS=1 and SL=0.10. It was possible to see similar cases for approximately 30 different sensors where there was excellent performance of SensorSAX both in reconstruction rate and data reduction. Thus, the results indicate that the system need to use optimum parameterisation for not only different type of data streams, but also for the sensors of the same type of data stream that can have different different dynamicity and environmental change even though they are not very far from each other.

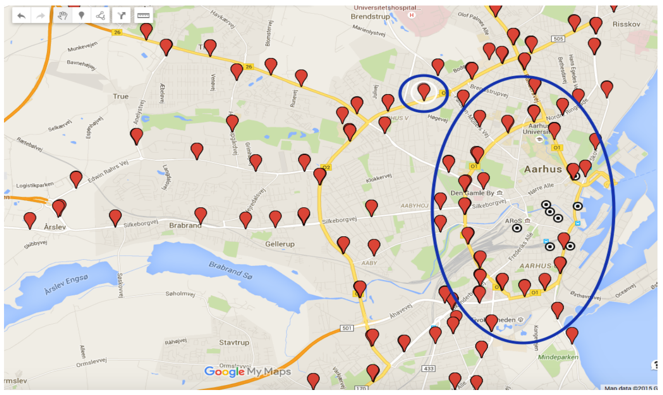
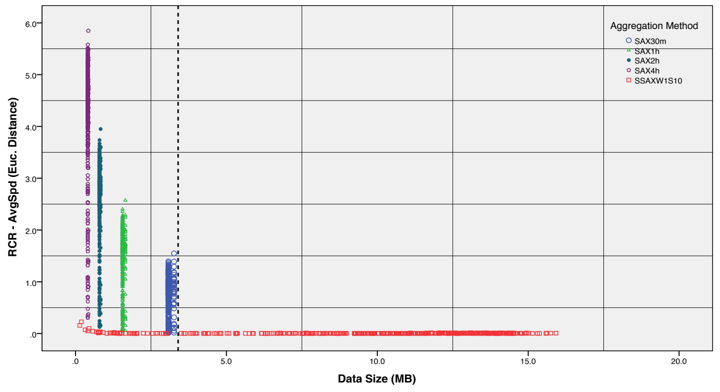


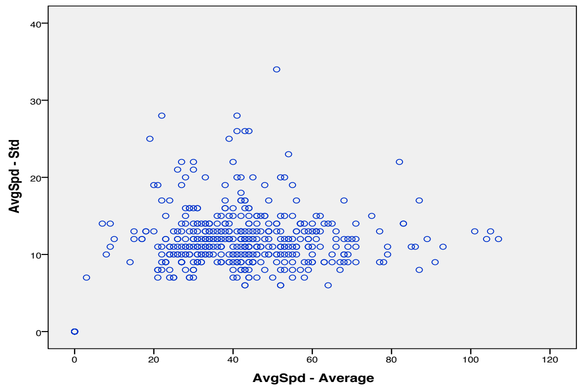
Fig. 8: A visual representation of geographical coordinates on Google Map for road traffic sensors provided by city of Aarhus, Denmark. The sensor given in a small circle represents location of the one of the high performance obtained for SensorSAX with MWS=1 and SL=0.10: Time: 12.9 seconds, CPU 28%, 1.28MB for data size, and 0.021 for RCR-AvgSpd and 2.95E-16 RCR - VcCnt. The larger circle depicts some of the sensors closer to the city centre and higher in road traffic.

Sensors can observe more than one phenomena in parallel, such as humidity, temperature, and in our case average speed and vehicle count in the city road traffic. Another important factor in adaptive segmentation of data aggregation occurred in the parallel search of optimum window sizes for multiple observation type of the sensors. We found many cases where variance in average speed were reached to the limit and vehicle count was still lower than the sensitivity level. Due to the fact that we have designed our system to report the phenomena whenever either observation reaches to the sensitivity level, this was another factor that should be taken into account in such dynamic environments as another trade-off in the performance of the adaptive segmentation data aggregation methods.

The results clearly demonstrate the advantage as well as disadvantage of transforming a time-series analysis approach from time dependent to data centric approach. There seems to be a clear trade-off between two approaches. SAX is not an adaptive segmentation approach and has to inform the system in a fixed time frequency for an actuation even in a sudden environmental change. Furthermore, SAX needs a



(a) The visualisation of reconstruction rate and data size performances of 449 sensors for 4 models SAX and 1 SensorSAX model.



(b) The visualisation of the average and standard deviation of all sensory observations of the City of Aarhus.

Fig. 7: Figure 7a illustrates a scattered representation of the results of SAX with WS=30m, 1h, 2h, 4h and SensorSAX with MWS=1 and SL=0.10 in terms of data size and reconstruction rate of average speed for 449 traffic sensors. The vertical dashed line represents the high performance variance of SensorSAX for different sensors in terms of data size. Figure 7b depicts the graphical representation of the average (i.e. x-axis) and standard deviation (i.e. y-axis) of the raw average speed traffic observations for 449 sensors.

preprocessing to obtain the optimum window size, while SensorSAX dynamically adapts its window size without the need of a training phase. On the contrary, SensorSAX enables to inform the system when there is a change in the sensory observations. However, although SensorSAX has got many advantages by having additional parameters to function as an adaptive segmentation method, our experiments suggest that the system needs to use different parameters depending on the type, frequency and distribution of data stream. There were some cases, where SensorSAX with small numbers of SL and MWS has clearly outperformed SAX approach with compelling results in terms of reconstruction rate, but this at the same time also caused to an increase in Data Size, CPU% and Time in stream processing.

Considering the fact that 1 month of annotated traffic data can produce approximately 250M triples, the obtained experimental results for the aggregated data streams are suitable for a real-time IoT systems. While the results are reported in average result of each individual sensor, it can be also noted that total data size for 449 annotated traffic data streams were 5.7 GB (i.e. 449×12.7 MB), and it has been reduced to various data sizes that were in range of 75.59% to 96.8% for SAX and 14.9% to 99.2% for SensorSAX with different data compression qualities and computational cost. These numbers indicate the importance of having a better data aggregation approach with minimum information loss, which can eventually lead to improve the performance of the applications that rely on the abstraction and correlation techniques.

Overall, in highly dynamic environments where the system receives data streams from multiple heterogeneous sensors, there is no ultimate parameter that can lead to perfect results for all data streams. The parameter selection is highly dependent on the type, sampling frequency and distribution of data stream. It is desirable to find the optimal data aggregation parameters for each data stream, which can be obtained by training the system based on the nature of phenomena being observed by the sensors. In addition, it is worth to point out that focusing solely on the data reduction is not adequate for IoT stream processing. In addition to the data size and reconstruction rate, it is evident that Time, CPU% consumption are highly significant factors in the data aggregation process of IoT stream processing. To have a fair comparison for both approaches on the traffic and parking data streams, we had to use the sax word size of “1” due to very low data frequency. Nevertheless, we believe that the difference will be more clear in higher frequency of data streams, where we can use a greater number of word length.

6 Related Work

Current IoT frameworks for smart cities usually focus on developing services enabling to access and interact with sensors, while leaving data preprocessing and processing to high level applications operating on top of the frameworks. Some of the well known examples that are focused only on providing a data platform, semantic modelling and/or semantic sensor discovery are Km4city [22], SmartSantander [23], Spitfire [24], OpenIoT [25], IoT.est [26], OpenCube [27], iCore [28]. Start-City [29] semantically annotates and aggregates traffic data with traditional aggregation methods (i.e. average, maximum, minimum) to predict spatio-temporal traffic conditions. The ontology here is domain specific (traffic domain), without generic concepts that can be reuse for different kind of sensors. However, none of these platforms involve ontologies to handle generic annotations for heterogeneous data stream nor an effective time-series data analysis approach for sensory domain. CityPulse framework enables to integrate data aggregation approaches into a smart city framework to reduce the communication overhead to the applications and increase the efficiency of both the framework and applications. The preprocessing at this stage focuses on SAX and SensorSAX algorithms on data aggregation and the interoperability between heterogeneous data is supported by adding semantics with SAO ontology.

6.1 Semantics in IoT Application Domain

Utilisation of semantics in the IoT with an extension to the smart city environments provide domain knowledge for sensor networks and services. The SSN ontology [17] is one of the most extended models used in the IoT domain. The SSN Ontology provides a vocabulary for describing concepts such as sensors, outputs, observation value and feature of interests. SSN has been extended to different IoT subdomain such as ontologies for coastal features, services and roles for emergency response. However, although the SSN ontology defines a higher-level scheme for sensor systems, it does not include representation of observation and measurement data. Within the EU FP7 IoT-A project [30] an information model was created. The IoT.est [15] defined semantic representations enhancing the IoT-A model with some service and test concepts. The Open Geospatial Consortium (OGC) describes Observation and Measurement (O&M) concepts for sensory data as a part of the Sensor Web Enablement (SWE) standard [31]. Although the OGC model provides important syntactic descriptions based on XML, it lacks some important semantic features to describe an ontology in more detail and expressing knowledge. There has been a recent study to improve the semantic richness of the O&M ontology where authors translated the XML expressions into Ontology Web Language (OWL) representation [32]. However, the O&M ontology lacks semantics needed for stream IoT data and lacks temporal features to represent time-series observations in detail. Henson et al. [33] proposed a similar approach mapping all XML tags of O&M ontology into OWL concepts. The authors present SPARQL queries to access the annotated data. However, these queries are not efficient for the applications that need to access sensory data in real time, as the SPARQL queries generates significant overload for large scale data.

6.2 Time-series Analysis

Data aggregation and time-series analysis enable to obtain a much smaller storage space and faster stream processing due to size and dimensionality reduction. Data aggregation process should keep a good-quality synopsis of data through identifying and removing superfluous and redundant data. *Lower bounding* principle is an important principle in time-series analysis to assure maintaining the meaning by keeping the distance between two time-series in the reduced space (i.e. aggregated data) less or equal than the true distance of time-series data (i.e. original data). Some of the well-known algorithms for numerical representation of time-series analysis that provide data reduction and satisfy lower bounding principle are Discrete Fourier Transform (DFT) [34], Discrete Wavelet Transform (DWT) [35], [36], Singular Value Decomposition (SVD) [37], and Piecewise Aggregate Approximation (PAA) [38].

DFT is one of the early time-series analysis, which allows each signal to be represented by a complex number known as Fourier coefficients. It uses the first few coefficients to reduce the original space. Although it satisfies the lower bounding principle by using *Parseval's Theorem*, the computational complexity (i.e. $C(n^2)$ time, or $C(n \log n)$ time with algorithm in [39]) and choice of the best number of coefficients are the main challenges of this algorithm [40]. In order to overcome the computational complexity, DWT is proposed as time series analysis technique and the need of entire dataset in analysis of Fourier Transform. While computational complexity of DWT ($C(n)$) is less than DFT and satisfies lower bounding principle, its limitation is that the data length must be a power of two ($n = 2^m$). In parallel, SVN can also perform dimensionality reduction optimally transforming a dataset into a new k-dimensional dataset based on the first ordered k-biggest singular values. However, it uses an entire dataset prior to transformation to perform dimensionality reduction and cannot work incrementally since a new data insertion requires a new global computation. PAA is a simple method compared with other more sophisticated transforms such as DFT and DWT. The computational complexity is low ($C(n)$), and it supports lower bounding principle, which can be simply calculated using the distances on PAA representations [41].

Contrary to the numerical approaches, the discretisation of the original data into symbolic strings has not been considered in great detail. Even though it seems a straightforward solution, it comes with substantial advantages over existing algorithms and data structures that enable the efficient manipulations of symbolic representations in addition to allowing the framing of time. However, general symbolic representation methods are not capable of calculating distance in symbolic space and supporting lower bounding at the same time [42], [43]. Symbolic Aggregate Approximation (SAX) is a symbolic representation technique on time series analysis [3] that is not only capable of providing significant data reduction but also a support for lower bounding principle. Due to the fact that it is based on PAA, it is not computationally expensive, and ensures both a considerable dimensionality reduction and the lower bounding support. However, SAX is highly domain and data dependent and its parameters often have to be chosen manually. The SAX parameters stay fixed in the original approach. Since data coming from IoT streams usually evolves over time, the optimal value for the window length does not stay the same. Therefore, we use an adaptive approach, called SensorSAX, to obtain reduced space of time series data [4].

7 Conclusions and Future Work

In this paper, we proposed a real-time stream processing framework and provided a comprehensive analysis of SAX and SensorSAX approaches using data streams with different sampling rate and data variation. We examined the framework by processing the data streams both annotated raw data and aggregated data. We introduced an information model that ensures that summarisation techniques can be interpreted as time-based events, even where further semantic associations are unavailable. The framework is tested with various parameters of SAX and SensorSAX approaches to find the increase in the performance of the annotation and aggregation of data streams with low computational cost and high reconstruction rate. The results indicate that there was a high variation in the performance of the data aggregation methods depending on not only the change of segmentation parameters, but also type, frequency, and dynamicity of data in the location of sensors. It is evident that there is a need of different parameterisation for each sensory location to obtain the best performance in the real-time analysis. Moreover, due to the fact that SensorSAX depends on the variation of data, the window size will always follow the variation and the reconstruction error will be always low.

In future work, we will incorporate a wider set of data streams with higher sampling frequency and utilise a deep learning approach in real-time to build an automatic approach by taking into account the presented factors and parameters to optimise the SensorSAX algorithm. We will also investigate how to provide an adaptive sax word size for the sensorSAX algorithm. Further work is also required for the SAO ontology to provide a better coverage for stream analysis techniques commonly used by researchers, as well as to enable better generalisation of the model, and harmonisation with existing research tools.

Acknowledgement

The research leading to these results has received funding from the European Commission's Seventh Framework Programme for the CityPulse project under grant agreement no. 609035.

References

[1] G. Anastasi, A. Falchi, A. Passarella, M. Conti, and E. Gregori, "Performance measurements of motes sensor networks," in *Proceedings of the 7th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. New York, NY, USA: ACM, 2004, pp. 174–181.

[2] A. Hameurlain, J. Küng, and R. Wagner, "Energy-aware data processing techniques for wireless sensor networks: a review," *Transactions on Large-Scale Data- and Knowledge-Centered Systems*, vol. 6790, pp. 117–137, 2011.

[3] J. Lin, E. Keogh, S. Lonardi, and B. Chiu, "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*. ACM Press, 2003, pp. 2–11.

[4] F. Ganz, P. Barnaghi, and F. Carrez, "Real world internet data," *Sensors Journal, IEEE*, vol. 13, no. 10, pp. 3793–3805, 2013.

[5] S. Kolozali, M. Bermudez-Edo, D. Puschmann, F. Ganz, and P. Barnaghi, "A knowledge-based approach for real-time iot data stream annotation and processing," *IEEE International Conference on Internet of Things (iThings 2014)*, September 2014.

[6] M. Gudgin, M. Hadley, N. Mendelsohn, J.-J. Moreau, H. F. Nielsen, A. Karmarkar, and Y. Lafon. (2007) Soap version 1.2 part 1: Messaging framework (second edition). [Online]. Available: http://www.w3.org/TR/soap/

[7] P. A. Castillo, J. L. Bernier, M. B. Arenas, J. J. M. Guervos, and P. Garcia-Sanchez, "Soap vs rest: Comparing a master-slave ga implementation," *Neural and Evolutionary Computing*, 2011.

[8] D. Guinard, L. Ion, and S. Mayer, "In search of an internet of things service architecture: Rest or ws-*? a developers' perspective," *Proceedings of the 8th International ICST Conference*, pp. 326–337, 2011.

[9] S. Vinoski, "Putting the "web" into web services: Web services interaction models, part2," *IEEE Internet Computing*, pp. 90–92, 2001.

[10] S.-H. Cha and T. Yun, "Html5 standards and open api mashups for the web of things," *Computer Applications for Web, Human Computer Interaction, Signal and Image Processing, and Pattern Recognition*, pp. 189–194, 2012.

[11] V. Wang, F. Salim, and P. Moskovits, *The definitive guide to HTML5 WebSocket*. Apress, 2013.

[12] G. D. Mandyam and N. Ehsan, "Html5 connectivity methods and mobile power consumption," *W3C Proposed Recommendation*, 2012.

[13] V. Pimentel and B. G. Nickerson, "Communicating and displaying real-time data with websocket," *IEEE Computer Society*, pp. 45–53, 2012.

[14] S. Vinoski, "Advanced message queuing protocol," *IEEE Internet Computing*, vol. 10, no. 6, pp. 87–89, 2006.

[15] W. Wang, S. De, R. Toenjes, E. Reetz, and K. Moessner, "A comprehensive ontology for knowledge representation in the internet of things," in *11th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. IEEE, 2012, pp. 1793–1798.

[16] Y. Raimond, "A distributed music information system," Ph.D. dissertation, School of Electronic Engineering and Computer Science, Queen Mary University, London UK, 2008.

[17] M. Compton, P. Barnaghi, L. Bermudez, R. García-Castro, O. Corcho, S. Cox, J. Graybeal, M. Hauswirth, C. Henson, A. Herzog *et al.*, "The ssn ontology of the w3c semantic sensor network incubator group," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 17, pp. 25–32, 2012.

[18] J. Cardoso, A. Sheth, J. Miller, J. Arnold, and K. Kochut, "Quality of service for workflows and web service processes," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 1, no. 3, pp. 281–308, 2004.

[19] R. L. Cruz, "Quality of service guarantees in virtual circuit switched networks," *Selected Areas in Communications, IEEE Journal on*, vol. 13, no. 6, pp. 1048–1056, 1995.

[20] P. Barnaghi, M. Bermudez-Edo, and R. Tönjes, "Challenges for quality of data in smart cities," *Journal of Data and Information Quality (JDIQ)*, vol. 6, no. 2, p. 6, 2015.

[21] S. Holm, "A simple sequentially rejective multiple test procedure," *Scandinavian Journal of Statistics*, vol. 6, pp. 65–70, 1989.

[22] P. Bellini, M. Benigni, R. Billero, P. Nesi, and N. Rauch, "Km4city ontology building vs data harvesting and cleaning for smart-city services," *Journal of Visual Languages & Computing*, vol. 25, no. 6, pp. 827–839, 2014.

[23] L. Sanchez, L. Muñoz, J. A. Galache, P. Sotres, J. R. Santana, V. Gutierrez, R. Ramdhany, A. Gluhak, S. Krco, E. Theodoridis *et al.*, "Smartsantander: Iot experimentation over a smart city testbed," *Computer Networks*, vol. 61, pp. 217–238, 2014.

[24] D. Pfisterer, K. Römer, D. Bimschas, O. Kleine, R. Mietz, C. Truong, H. Hasemann, A. Kroller, M. Pagel, M. Hauswirth *et al.*, "Spitfire: toward a semantic web of things," *Communications Magazine, IEEE*, vol. 49, no. 11, pp. 40–48, 2011.

[25] J. Soldatos, N. Kefalakis, M. Hauswirth, M. Serrano, J.-P. Calbimonte, M. Riahi, K. Aberer, P. P. Jayaraman, A. Zaslavsky, I. P. Žarko *et al.*, "Openiot: Open source internet-of-things in the cloud," in *Interoperability and Open-Source Solutions for the Internet of Things*. Springer, 2015, pp. 13–25.

[26] S. De, F. Carrez, E. Reetz, R. Tönjes, and W. Wang, *Test-Enabled Architecture for IoT Service Creation and Provisioning*. Springer, 2013.

[27] E. Kalampokis, A. Nikolov, P. Haase, R. Cyganiak, A. Stasiewicz, A. Karamanou, M. Zotou, D. Zeginis, E. Tambouris, and K. Tarabanis, "Exploiting linked data cubes with opencube toolkit," in *International Semantic Web Conference (ISWC)*, 2014.

[28] P. Vlacheas, R. Giaffreda, V. Stavroulaki, D. Kelaidonis, V. Foteinos, G. Poulios, P. Demestichas, A. Somov, A. Biswas, and K. Moessner, "Enabling smart cities through a cognitive management framework for the internet of things," *Communications Magazine, IEEE*, vol. 51, no. 6, pp. 102–111, June 2013.

[29] F. Lécué, S. Tallevi-Diotallevi, J. Hayes, R. Tucker, V. Bicer, M. L. Sbodio, and P. Tommasi, "Star-city: semantic traffic analytics and reasoning for city," in *Proceedings of the 19th international conference on Intelligent User Interfaces*. ACM, 2014, pp. 179–188.

[30] S. De, T. Elsaleh, P. Barnaghi, and S. Meissner, "An internet of things platform for real-world and digital objects," *Scalable Computing: Practice and Experience*, vol. 13, no. 1, 2012.

[31] M. Botts, G. Percivall, C. Reed, and J. Davidson, "Ogc® sensor web enablement: Overview and high level architecture," in *GeoSensor networks*. Springer, 2008, pp. 175–190.

[32] C. A. Henson, H. Neuhaus, A. P. Sheth, K. Thirunarayan, and R. Buyya, "An ontological representation of time series observations on the semantic sensor web," in *1st International Workshop on the Semantic Sensor Web (SemSensWeb 2009)*, 2009, pp. 79–94.

[33] C. A. Henson, J. K. Pschorr, A. P. Sheth, and K. Thirunarayan, "Semsos: Semantic sensor observation service," in *International Symposium on Collaborative Technologies and Systems*. IEEE, 2009, pp. 44–53.

[34] C. Faloutsos, R. Agrawal, and A. Swami, "Efficient similarity search in sequence databases," *Proceedings of the 4th Conference on Foundations of Data Organisation and Algorithms*, pp. 69–84, 1993.

[35] A. Haar, ""zur theorie der orthogonalen funktio- nensysteme". german," in *Mathematische Annalen*, vol. 69, no. 331–371, 1910, pp. 38–53.
[36] Z. R. Struzik and A. Siebes, "The haar wavelet transform in the time series similarity paradigm," in *Principles of Data Mining and Knowl- edge Discovery*. Springer, 1999, pp. 12–22.
[37] K. Chakrabarti, E. Keogh, S. Mehrotra, and M. Pazzani, "Locally adaptive dimensionality reduction for indexing large time series databases," *ACM Transactions on Database Systems*, vol. 27, no. 2, pp. 188–228, 2002.
[38] K. Chakrabarti, E. Keogh, M. Pazzani, and S. Mehrotra, "Dimensionality reduction for fast similarity search in large time series databases," *Journal of Knowledge and Information Systems*, vol. 3, pp. 263–286, 2000.
[39] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex fourier series," *Mathematics of Computation*, vol. 19, pp. 297–301, 1965.
[40] C. Cassisi, P. Montalto, M. Aliotta, A. Cannata, and A. Pulvirenti, *Similarity Measures and Dimensionality Reduction Techniques for Time Series Data Mining*. InTech, 2012, ch. 3, pp. 71–96.
[41] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra, "Dimensionality reduction for fast similarity search in large time series databases," *Journal of Knowledge and Information Systems*, vol. 3, pp. 263–286, 2000.
[42] H. Andre-Jonsson and D. Z. Badal, "Using signature files for querying time-series data," in *First European Symposium on Principles of Data Mining and Knowledge Discovery*, 1997, pp. 211–220.
[43] Y. wu Huang and P. S. Yu, "Adaptive query processing for time-series data," in *The 5th International Conference on Knowledge Discovery and Data Mining*. ACM Press, 1999, pp. 282–286.

Şefki Kolozali is a research fellow at the University of Surrey. He received a B.Sc. degree in Computer Engineering from the Near East University, Nicosia, Turkish Republic of Northern Cyprus, in 2005. He received an M.Sc. degree from the University of Essex, and a Ph.D. degree from Queen Mary University of London. His thesis was titled Automatic Ontology Generation Based On Semantic Audio Analysis. His main research interests include signal processing, machine learning, semantic web technologies, semantic sensor networks, Internet of Things, and Future Internet technologies.

Daniel Puschmann is currently pursuing his Ph.D. degree from the Institute for Communication Systems, University of Surrey. His research is focused on information abstraction and extracting actionable information from streaming data produced in the Internet of Things using stream processing and machine learning techniques.

Maria BermudezEdo is a lecturer (Assistant Professor) at the University of Granada. Her research interests include semantic technologies, Internet of Things and machine learning. Her research has appeared in the proceedings of various peer-reviewed international conferences and journals.

Payam Barnaghi is a Reader at the Institute for Communication Systems Research at the University of Surrey. He's also the coordinator of the EU FP7 CityPulse project. His research interests include machine learning, the Internet of Things, the Semantic Web, adaptive algorithms, and information search and retrieval. He's a senior member of IEEE and a Fellow of the Higher Education Academy.