A Practical Evaluation of Information Processing and Abstraction Techniques for the Internet of Things

Frieder Ganz, Daniel Puschmann, Payam Barnaghi, *Senior Member, IEEE,* and Francois Carrez

The term Internet of Things (IoT) refers to the interaction and communication between billions of devices that produce and exchange data related to real world objects (i.e. Things). Extracting higher-level information from the raw sensory data captured by the devices and representing this data as machine-interpretable or human-understandable information has several interesting applications. Deriving raw data into higher-level information representations demands mechanisms to find, extract and characterise meaningful abstractions from the raw data. This meaningful abstractions then have to be presented in a human and/or machine-understandable representation. However, the heterogeneity of the data originated from different sensor devices and application scenarios such as e-health, environmental monitoring and smart home applications and the dynamic nature of sensor data make it difficult to apply only one particular information processing technique to the underlying data. A considerable amount of methods from machine-learning, the semantic web, as well as pattern and data mining have been used to abstract from sensor observations to information representations. This paper provides a survey of the requirements and solutions and describes challenges in the area of information abstraction and presents an efficient workflow to extract meaningful information from raw sensor data based on the current state-of-the-art in this area. The paper also identifies research directions at the edge of information abstraction for sensor data. To ease the understanding of the abstraction workflow process, we introduce a software toolkit that implements the introduced techniques and motivates to apply them on various data sets.

Index Terms—**Internet of Things, Data Abstraction, machine-learning, Semantic Web, Software Tools**

I. Introduction

THERE is a growing trend towards integrating real world data into the Internet. The Internet of Things (IoT) aims to develop technologies and create infrastructures that enable integration of billions of sensory devices and real world objects that provide different capabilities and produce and exchange data. It is predicted that by the next 5-10 years there will be around 50 billion Internet connected devices that will produce 20% of non-video traffic on the Internet [9]. This leads to a Big Data challenge[2], a term often referred to as a tremendous volume of highly variable streaming data that requires sophisticated mechanisms to make it available and valuable for the end-user.

In the past, extensive research has been conducted in terms of the technologies close to the sensor hardware such as communication protocols [3], energy efficiency [28], heterogeneous sensor device integration [1], and programming languages [37]. Significant progress has also been made in accessing and representing the dynamic real-world data on the Internet, for instance some of the recent developments are reported in the area of the semantic sensor web [42]. However, the question, how the sensor data is transferred from its raw form into higher-abstraction representations and eventually how it is made accessible and understandable for humans or interpretable by machines and decision making systems remains still open [46].

In pervasive computing, especially in smart-* environments commonly used sensors monitor physical attributes such as light, temperature, noise, movement and humidity. The data communicated by sensors consist of time-series values that are sampled over a defined period and then transmitted to a sink/gateway for further processing.

Time-series data is not as easy interpretable as for instance a document, video or any other data available on the Internet. Platforms such as Xively[1] (former Cosm) or Nimbits [2] allow publishing and visualisation of streaming data from sensor devices, however, they lack processing and analytic features; The data remains in the same raw condition and makes it difficult to detect interesting information, especially with regards to the vast amount of sensors that will be connected to the Internet in the future and lead to consequent challenges that form the Big Data issue in IoT.

In the research domain of sensor networks there are well investigated topics such as event and pattern detection, data mining and context-aware computing [50]. However, most approaches use raw sensor data for their analysis in a specific application domain [14, 47, 31, 48, 15] where it can be assumed which events and particular information is going to be detected. With the emerging large volumes of heterogeneous data and their various application scenarios, new domain independent approaches are needed that can abstract from the underlying data and enable a human/machine interpretable representation of the data. Sensor abstraction from raw data has two major advantages: a) As a replacement of raw sensor data, abstractions can be used for further processing and annotation. Abstractions are less granular as raw data and therefore require less data-space and communication traffic. b) Abstractions are easier to understand by the end-user or to be interpreted by automated machine processes. For instance, instead of transmitting the raw samples [-5 C, -3 C, ..., -2 C, 0 C, -4 C] it might be more valuable to transmit an abstract concept such as ”cold”. Furthermore, a higher abstraction level leads to a greatly reduced communication cost. However, this

Manuscript submitted January 30, 2014

Frieder Ganz, Daniel Puschmann, Payam Barnaghi and Francois Carrez are with the Centre for Communication Systems Research at the University of Surrey, Guildford, United Kingdom. Email: F.Ganz@Surrey.ac.uk.

[1]https://xively.com/

[2]http://www.nimbits.com/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2015.2411227, IEEE Internet of Things Journal

will come at the cost of losing some part of the information and also requires context information in which the data has been obtained [43]. The granularity of information required depends on the application and/or the requirements of the end-user.

In this survey paper we focus our attention on approaches and methods that can be used to abstract from the raw data to higher-level representations. Several research in the IoT domain have been carried out to investigate how data can be made accessible via devices. It is still an open challenge how the data can be interpreted in a meaningful way and how actionable information can be extracted from the raw IoT data. The main objective of this work is to survey algorithms and techniques that have been used in the data mining domain and apply them to data analytics tasks in the IoT. To ease the understanding of the different methods, we provide a software toolkit, that incorporates some of the most common techniques in a user friendly manner. In Section 2, we state more precisely the definition of information abstraction and motivations behind its application. Section 3 introduces a workflow with several steps from pre-processing to the representation of abstractions. For each step, we provide some possible algorithms and methods that can be applied. Section 4 gives an overview in the state-of-the-art in information abstraction from a technical and research point-of-view and discusses the current requirements for information abstraction. in Section 5 we shortly introduce our toolkit for knowledge acquisition and information abstraction for sensor data and exemplify it on two use cases. Section 6 concludes the paper and gives an outlook for future work.

II. Definition of Information Abstraction and Knowledge Representation

This section defines and discusses the terms information abstraction from sensor data and its different forms of representation including different levels of abstraction, its distinction to other research areas, and discusses motivation and challenges of creating abstractions from sensor data.

A. What is an Abstraction?

The term abstraction as we use it in this work, is coined in the area of context-aware computing, describing the transition from different levels of context incorporation from a sensing layer to a perception layer and finally to a situation layer [11]. This transitioning process is defined by Chen and Kotz [8] as deriving higher-level context data from lower-context (i.e. raw) sensor data by collecting, aggregating and inferring raw data with additional knowledge from the environment with the goal to adjust the sensor devices behaviour to the current context. With the Internet of Things, where data eventually has to be made available and understandable for the end-user, the focus of abstraction moves from a device point of view to a more user-centric position. Sigg *et al.* [43] define abstraction as the amount of processing applied to the data with the goal to raise the level of context abstraction including the error probability induced by each transition.

In this paper, we define two granularity levels of abstraction with the aim to represent the knowledge with a user-centric focus; lower-level abstraction (or data abstraction) and higher-level abstraction (or semantic abstraction). We define the process of abstraction as the derivation from raw data to more valuable and understandable information.

Lower-level abstractions represent atomic and static information which can be obtained by gathering data from a single local sensor stream and by combining the data with metainformation about the local sensors such as type, range and capabilities. Atomic in this case, means that this is the lowest abstraction level after the processing of raw sensor data. Static in this context means that the abstraction is a single and independent observation made at a fixed point in time and does not include information about a sequence of observations. Mantyjarvi [36] describes this as ”smallest atomic quantity of context information with semantic meaning”. For instance, a door sensor can measure two states, either the door is open or closed (assuming that a door cannot be half-open and must be either opened or closed). The abstractions open and closed fully represent the situation and cannot be further abstracted. Both abstractions do not refer to a sequence of actions over time. Data information can be obtained through data processing techniques such as pattern and event detection that analyse the raw sensor data of a single node and inform the user/network about the occurrence of the event.

Higher-level abstractions however can be inferred by observing several sources of lower-level abstractions to get the global picture about occurring activities and multivariate events. A certain pattern of open and closed doors during specific times of the day and other lower-level abstractions can lead to the higher-level abstractions *beginning of work day* and *end of work day*. Higher-level abstractions can be obtained by machine-learning techniques such as classification and clustering of lower-level abstractions over time. Different approaches such as logical inference with the help of reasoning mechanisms and rule-based systems can be also used for this purpose.

The representation form of the abstraction can vary in different applications for sensor data. Graphical user interfaces including geographical maps can visualize the abstracted data and allow the end-user to perceive information, events and changes in the environment quickly and sometimes even without the need of expert knowledge. Semantic representations of information such as those defined in the Semantic Sensor Ontology [10] can provide interlinked information obtained by the abstraction process to the user and be used to query the status of the real world. Transferring the abstractions into a machine understandable format can also raise the interoperability of data.

B. Motivation for Information Abstraction

There is a huge demand for new data processing techniques and concepts to cope with the issues of the big data problem. We endorse that information abstraction can be used as a mean to reduce the deluge of data. Focusing on the abstracted information rather than the numerical data, can bring two main advantages: network traffic reduction and the enhancement of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2015.2411227, IEEE Internet of Things Journal

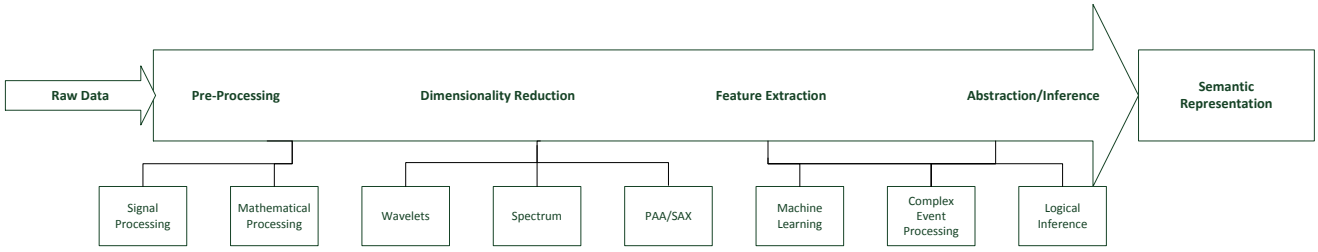

Fig. 1: Common Information Abstraction processes

comprehensiveness for the end-user. Instead of transmitting the raw data to the user, abstracted data is less granular but focus on the information which is useful for the user. Compared to lossless compression techniques, abstraction does not focus on reconstructing the initial data but allows extracting the information that is interesting for the user. Data abstraction can be used as a fundamental base for existing approaches such as outlier detection, activity recognition and other emerging areas in the domain of sensor networks.

Information Abstraction exploits several techniques and methods from different research areas to provide comprehensible information from a large amount of raw data to the user that are introduced in the following.

III. Creating abstractions

In the following, we introduce a general workflow that has been defined by examining several different approaches for information abstraction in the domain of sensor data (details in Section IV). The approaches that have been examined either follow the workflow as shown in Figure 1 or implement certain parts of it. Therefore we extracted the following main steps that serve as a common ground for the workflow: Pre-processing to bring the data into shape for further processing, dimensionality reduction to either aggregate the data or reduce its feature vectors, feature extraction to find lower-level abstractions in local sensor data as defined in Section II, Abstraction from lower-level abstractions to higher-level abstractions and finally representation to make the abstracted data available for the end-user and/or machines that can interpret the abstracted data. We introduce the different steps and key techniques used in this domain. All methods that are demonstrated use a synthesized test data set. The synthesized data set consists of 2048 samples. The first 1024 samples are Gaussian random numbers between 0.0 and 100.0, the next 512 samples represent Gaussian random numbers between 0 and 300 and the last 512 random numbers are in between 0 and 100. This has been chosen to model some kind of activity in between two periods of no activity and also to represent dynamicity in the data. The dataset has been constructed in this way to help increasing the comprehensiveness of the presented methods and techniques and the dataset does not aim to be a general representative for IoT data. In fact the very nature of IoT prohibits finding such a representative dataset that can cover various types and features all in a limited dataset. To showcase the applicability to the real world, the data used in the use-case scenarios in section V have been collected from actual sensor measurements.

On the Figures 2, 3, 4, 5 and 7 the annotation of the axes have been omitted because the figures showcase how the pattern of the data becomes more visible after applying the techniques. Because some of the techniques reduce the dimensionality, the patterns would be distorted by using the same annotations without rescaling. On the other hand, rescaling and inserting the annotations for each of the subfigures would clutter the overall figure and distract from the main information.

A. Pre-processing

The raw sensory data passes through a pre-processing stage to prepare the data for further steps. Pre-processing can be done on the sensor node, to reduce transmission cost and filter unwanted data. This can include mathematical/statistical methods to smooth the data by applying moving average windows, or methods from signal processing such as band-, low-, high pass filter to focus on certain frequency spectra. Transmission cost can be reduced by only sending certain information of a current sampling window to the base station/gateway such as the minimum and/or maximum values or the mean value of the current window.

The pre-processing is not only limited to a single sensor node, certain approaches use in-network processing to aggregate the data before further processing by finding the minimum, mean or maximum value in a set of sensor nodes before transmitting the data to the base station. Apart f local aggregation, in-network techniques can also be used to improve the accuracy of the data by calculating correlation with data from neighbouring nodes. The survey of Figo *et al.* [17] describes pre-processing techniques in detail. The applied pre-processing techniques introduced in this section are shown in Figure 2 and described in the following in sections:

1) Signal Pre-Processing

A filter can either be a simple hardware circuit or simple algorithm that removes unwanted parts of a signal in frequency domain by cutting the signal after/before a certain frequency. This leads to the advantages that less data has to be submitted and further processing steps have a focused dataset without background noise. However the trade-off for filtering the data is that outliers or other interesting data can be missing.

Low/High-Pass Filter: A low/high-pass Filter cuts off the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2015.2411227, IEEE Internet of Things Journal

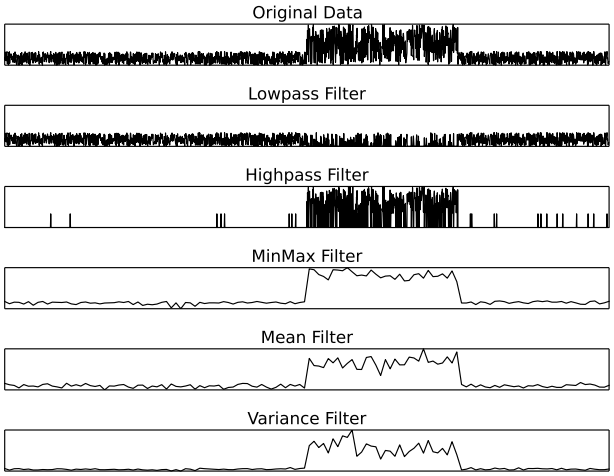

Fig. 2: Pre-processing techniques

current signal in frequency domain after/before a certain threshold: called the cut-off frequency. Arora *et al.* [4] use a low-pass filter to smooth the signal to prevent a split of activities in the later processing. Eriksson *et al.* [15] use high-pass filter to remove low-frequency components in a road-anomaly detection scenario where sensors are deployed on a car. The filter removes subtle changes in the acceleration signal and passes only high-frequency signal that are most probably caused by holes and cracks in the road.

Bandpass Filter: A Bandpass Filter has two cut-off frequencies, the lower and the upper frequencies and will only pass the signal in between. Stocker *et al.* [44] use bandpass filter to pre-process signals from a vibration sensor deployed at a road pavement to retrieve only data that is created by passing cars. Wang *et al.* [48] use bandpass filters for bird observation, where it is known that the birds produce a sound only in a certain frequency range. Olfati-Saber [39] introduces an approach for a distributed filter that includes several high and low-pass filters deployed over a sensor network to minimise the overall background noise and increase the accuracy of the observations by combining data from several sensor nodes.

2) Mathematical/Statistical Pre-Processing

In contrast to signal processing, mathematical pre-processing techniques do not utilise the signal and frequency but work on the produced output instead. Data windows are used to aggregate the data over a time window and transmit it either directly to the base station (e.g. a gateway) for further processing or disseminate the aggregated data over the network for in-networking processing before further processing.

Min, Max: The difference between the minimum and maximum inside a sample window can be used as a pre-processing step for further feature detection. Farringdon *et al.* [16] use the range of the min/max difference in combination with the averages to detect the orientation of a sensor badge attached to a person. Based on the values they detect if the person is standing, sitting or lying .

Mean, Median: The Mean or Median is usually used to smooth the data by removing peaks and noise from the signal. To use the mean or the median on streaming data, the moving average (median) can be applied by taking only the last n values into consideration and then subsequently shifting forward the sliding window. Ghasemzadeh *et al.* [21] use the moving average as a pre-processing step in a body sensor network to detect patterns in the neuromuscular system based on EEG signals. In their application scenario the moving average is used to cancel high frequency noise.

Variance, Standard Deviation: Both variance and standard deviation are used to represent the volatility of the data. Golding and Lesh [22] calculate the variance and standard deviation of the raw data to track people with cheap sensor devices.

Correlation, Integration: Especially with multi dimensional data from accelerometers, correlation and integration are used to get velocity and and position. By calculating the derivation of the speed, the distance can be approximated.

B. Dimensionality Reduction

To cope with the large amount of data that has to be processed and stored, dimensionality reduction techniques can be applied to reduce the size and length of the data by applying different methods on the data while keeping the key features and patterns.

The goal of dimensionality reduction is to reduce the length of an input Vector X_n with length n to a reduced vector of size M where $M << n$. Different methods have been introduced that either aggregate the data or filter certain samples of the original data to reduce the length of the initial data. This section gives an overview of some of the frequently used techniques.

Discrete Fourier Transformation: The Discrete Fast Fourier Transformation (DFT) transforms a signal from the time domain to a frequency domain. The signal is aligned along the frequency axis, resulting in an output vector of frequencies ranging from low-frequency to high-frequency coefficients. To reduce the dimensionality of the original time-series data, the data is transformed via DFT into the Fourier coefficients. Then only the first few coefficients are used to represent the original sequence. The shortened transformed vector is subsequently used in the inverse DFT to reconstruct the original data. The formula for transformation and inverse transformation (=reconstruction) are shown in Equation 1. In Figure 3 the original data and the transformed data with only n coefficients is depicted. The value n also represents the length of the output, the smaller the reduced vector, the lesser its resolution.

$$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-i\ 2\pi\ k\ n\ /\ N}$$
$$x_n = \frac{1}{N} \sum_{k=0}^{N-1} X_k e^{i\ 2\pi\ k\ n\ /\ N} \quad (1)$$

Wavelet Transformation: In comparison to the Fourier transformation that loses the time information of the data and transforms the data globally, discrete wavelet transformation (DWT) preserves the time dimension and transforms the data

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2015.2411227, IEEE Internet of Things Journal

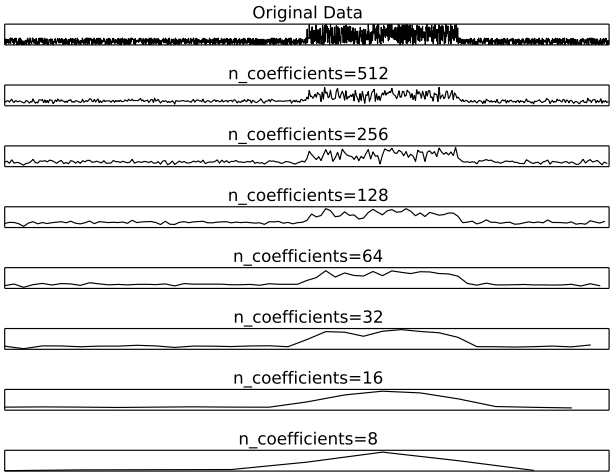

Fig. 3: Original Data, and reconstructed Fourier transformation with less coefficients

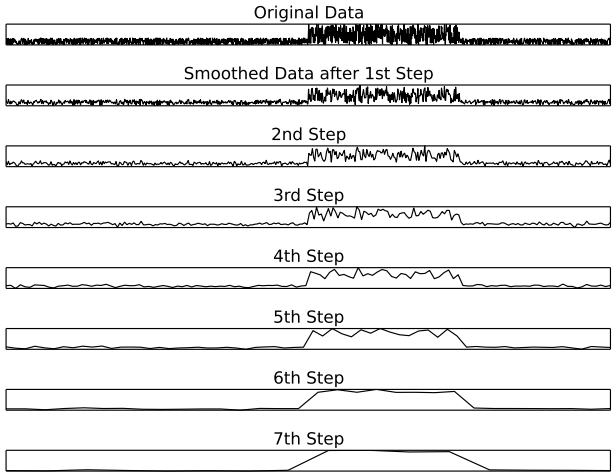

Fig. 4: Data transformed via DWT at different iteration Steps

locally which leads to a faster calculation. The Haar wavelet transformation originated in 1910 by Alfred Haar [24] is still frequently used in the domain of time-series analysis [45]. The transformation takes a one-dimensional input vector $X_S = [s_1, s_2, .., s_n]$ of length n and transforms it into two sets: a set of averages referred to as the smoothed values and the differences referred to as wavelet coefficients. Similar to the Fourier Transformation, the wavelet transformation works with input vectors with a length of a number in the power of two (2, 4, 8, 16 ...). The transformation is a recursive algorithm that in each step i calculates the average of the input for any 2 values and the difference between the values to the average by the formula in Equation 2.

$$\begin{aligned} coefficient_i &= \frac{s_i - s_i + 1}{2} \\ smooth_i &= \frac{s_i + s_i + 1}{2} \\ i = 2k + 1, &\text{where k is an integer} \end{aligned} \tag{2}$$

The following example demonstrates how the algorithm works:

Let us assume that the input vector for the transformation is: $X_8 = [2, 2, 3, 1, 5, 9, 1, 3]$. During the first Recursion step, averages and differences for X_8 are generated. Afterwards, the averages serve as input for the next recursion step. The Differences are stored separately and kept in a different vector. The result after step one is: $smooth = [2, 2, 7, 2]$ and $coefficient = [0, 1, 3, 1]$ The differences are attached to the previous differences, after the second recursion step the result is: $smooth = [2, 4.5]$ $coefficient = [0, 2.5, [0, 1, 3, 1]]$ The recursion ends when only a single averaged value remains leading to the result: $smooth = 3.25$ $coefficient = [1.25, [0, 2.5, [0, 1, 3, 1]]]$ In Figure 4 the process is visualised by applying the DWT over the sample data set. On the top left the original data is shown.

Piecewise Aggregation Approximation: The Piecewise Aggregation Approximation (PAA) [33] transformation is similar to the DWT smooth coefficient. However, PAA takes an output window length as a parameter to calculate the averages of the original data. The computation of the PAA involves the process of reducing the dimensionality of a time-series by averaging the data; this is shown in the following: PAA transforms a time-series vector X of length n into a reduced vector $\bar{X} = [\bar{x}_1, \bar{x}_2, ..., \bar{x}_m]$ with length m. Each element $\bar{x}_i$ is calculated with the formula shown in equation 1. Figure 5 visualises the process of applying PAA on the sample data set.

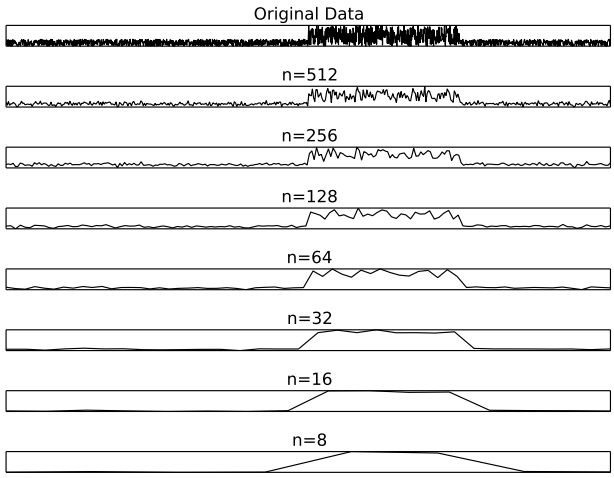

Fig. 5: Original Data and PAA transformation with different window lengths

$$\bar{x}_i = \frac{m}{n} \sum_{j=n/m(i-1)+1}^{(n/m)i} x_j \tag{3}$$

By applying the process to the series $X = [4, 8, 3, 2, 1, 1, 1, 1, 1, 1, 10, 5]$ with length $n = 12$ and an aimed reduced vector $\bar{X}$ of length $m = 6$ we get the result $\bar{X} = [6, 2.5, 1, 1, 1, 7.5]$.

Variable PAA: PAA has the drawback that it works with a fixed window length. In times of low data activity, the same event is aggregated over and over. In contrast if there is a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2015.2411227, IEEE Internet of Things Journal

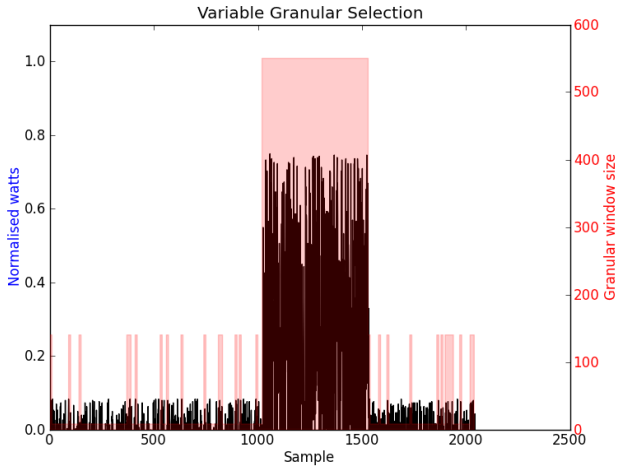

Fig. 6: Variable PAA and the adapted window sizes

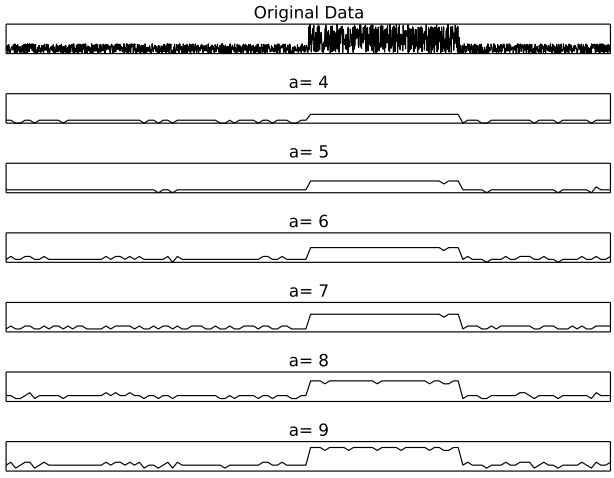

Fig. 7: Original Data and reconstructed SAX transformations with different alphabet sizes

lot of activity, aggregation can lead to information loss. As an extension we introduced an adaptive PAA approach in [20] that adapts the length according to the data activity for ultimately less data communication and better reconstruction of the original data. To select the different levels of granularity a method has to be introduced that based on the data activity chooses the right length m of the reduced data. The variability measure defines how far values are spread out. This can be used to create a higher granularity in values that are more distant to the mean of the data. The variable PAA approach assumes that the values away from the mean are more interesting and those values should be represented with a higher granularity then data that is close to the mean. To select m, we introduce functions for each statistical method that lead to a higher granularity based on the distribution of the data. In the case that the variance is in the first quartile of the distribution a smaller m is selected. If the variance is within the range of the second quartile, then a medium m is selected. In Figure 6 the variable PAA is applied to data. In times of more activity in the data, a higher window resolution is chosen.

Symbolic Aggregate Approximation: The Symbolic Aggregate Approximation (SAX) [34] transforms a time-series into a discretised series of letters referred to as a word. SAX transforms the data into a reduced set by initially applying PAA first. Afterwards, the data gets discretised into letters by applying breakpoints according to a Gaussian distribution to the PAA output vector. The breakpoints β are generated according to an alphabet size a, which later represents letters from an alphabet. The PAA transformed vector is then discretised so that each point is between the interval $[\beta_{i-1}, \beta_i]$ with $\beta_0 = -\inf$ and $\beta_{inf} = \inf$. Figure 7 shows a data series and the reconstructed time-series after the SAX transformation with different alphabet sizes a.

C. Feature Extraction, Abstraction/Inference

After pre-processing of the raw data and the dimensionality reduction, features (e.g. interesting events) have to be extracted. Feature extraction describes the process of extracting representative features from the sensor data [25]. Feature extraction is an ambiguous terminology and used sometimes synonymously with dimensionality reduction, cluster analysis and feature selection. Originally based in the domain of pattern-recognition in images, feature extraction reduces the image to certain regions or characteristics to lower the amount of data that has to be processed to find similar images or differences between similar images.

In time-series data, feature extraction can be used to detect outliers by finding a reduced feature set that separates between regular values and outliers. A more detailed evaluation can be found in [30].

Abstraction and inference describe methods which use the extracted features to gain more information about the data and infer knowledge from that. In this work we group the two steps of extraction and inference from features into one, a process that abstracts from the pre-processed data to information that is machine and/or human interpretable. In the following some abstraction methods from the pre-processed/dimensionality reduced data are presented that are commonly found in the literature.

Clustering: Clustering algorithms group samples with similar or close attributes into the same group. Similarity measures can be defined beforehand, for example as the Euclidean distance. In time-series analysis the similarity can be computed by comparing the observed values but also by comparing metainformation such as observation time or observation type. A common technique to cluster data is the k-means algorithm [27] that calculates the similarity based on the euclidean distance between data samples. k-means requires the expected number of groups k as an initial parameter. The first step usually chooses centroids randomly from the samples. Afterwards the distance to the centroids of the other samples is calculated and based on the distance grouped to the closest centroid. In a recursive process, the average of each cluster is calculated and if required shifted until the centroids converge

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2015.2411227, IEEE Internet of Things Journal

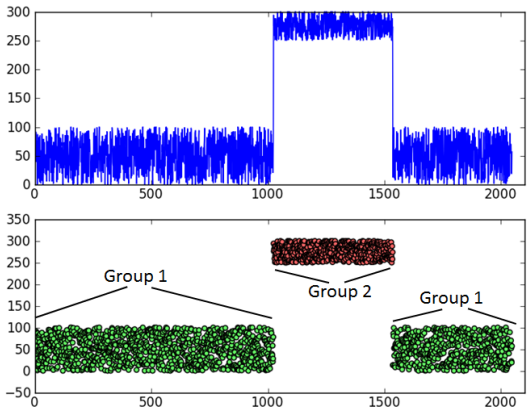

Fig. 8: k-means with $k = 2$ applied to the data; different colours state different cluster types

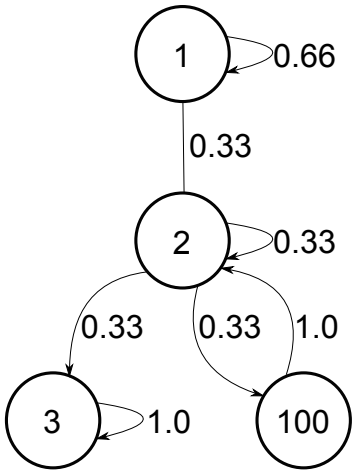

Fig. 9: Markov chain created by the frequency of the values in $[1, 1, 1, 2, 100, 2, 2, 3, 3, 3]$

to a certain point. Variations of that algorithm include non-random centroid starting points (Global k-means) or using the median or the medoids to shift the centroid (k-median, k-medoid). In Figure 8, k-means clustering is applied to the data with $k = 2$. The algorithm is applied on the data values, and it can be seen by the color coding that two groups of "lower" values and "higher" values are grouped together. Typical applications are the detection of outliers or grouping the data into non-temporal related groups.

Markov Chains: The frequency of samples or groups and their temporal occurrence can be used to construct Markov chains that represent the likelihood of temporal relations. The model is able to represent relations between values through temporal properties such as "Occurs After" and "Occurs Before". To visualize a simple Markov chain, we use the following measurements $[1, 1, 1, 2, 100, 2, 2, 3, 3, 3]$. It can easily be seen that the value 100 is an outlier and its likelihood to appear in a sensor stream is low. Also it can be seen that the chain terminates with the value 3 and therefore there is only a leading edge to itself with the probability 1. A graphical representation of the chain with the samples as vertexes and the probabilities as directed edges is shown in Figure 9.
Hidden Markov Model: Hidden Markov Models (HMM) add the temporal dimension into account and can be used for classification purposes similar to the clustering approach above. However, instead of looking solely at the attributes of the data also their temporal occurrence is considered. A HMM consists of several hidden states. The hidden states are formed by several input factors (emissions), where each emission leads to a state with a certain probability. In Figure 10, a hmm classification is applied to the same data as used in the k-means example. As a starting parameter we set the number of hidden states to 3. Based on their values and their temporal occurrence the data is coloured according to its state. This leads to 3 different groups: two "lower" value groups in different time epochs and one "higher" value group. To stress the difference on temporal clustering between the HMM and k-means approach we compare both in Figure 11. In the top

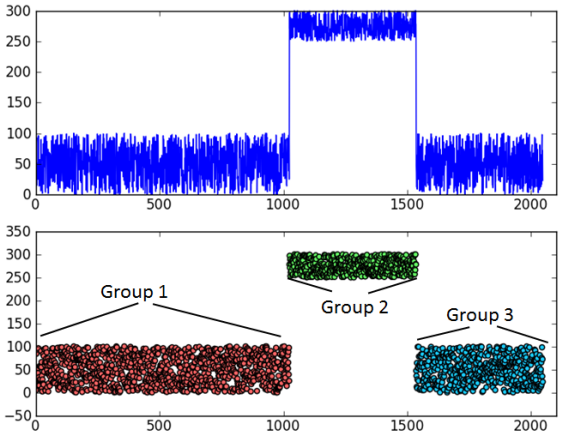

Fig. 10: HMM with three states applied to the data; different colours state different cluster types

diagram, the k-means approach with $k = 3$ is shown, grouping the data only based on their values into a "lower", "medium" and "higher" value group. In the bottom diagram we see the HMM classifier with 3 states grouping the data into groups also according to their temporal occurrence.

D. Semantic Reasoning & Representation

Semantic models allow to represent data, its metadata and the related context information in a linked graph model. For instance, the groups and events that have been learned through clustering and classification techniques can be represented. And also their relation to each other and the raw data can be modelled. The interlinked representation of events and observations in a semantic ontology allows to reason from simple events to more abstracted events e.g from simple tasks such as walking, or running to complex group activities through semantic rules. Common semantic representations relay on graph models where vertexes represent classes or instances of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2015.2411227, IEEE Internet of Things Journal

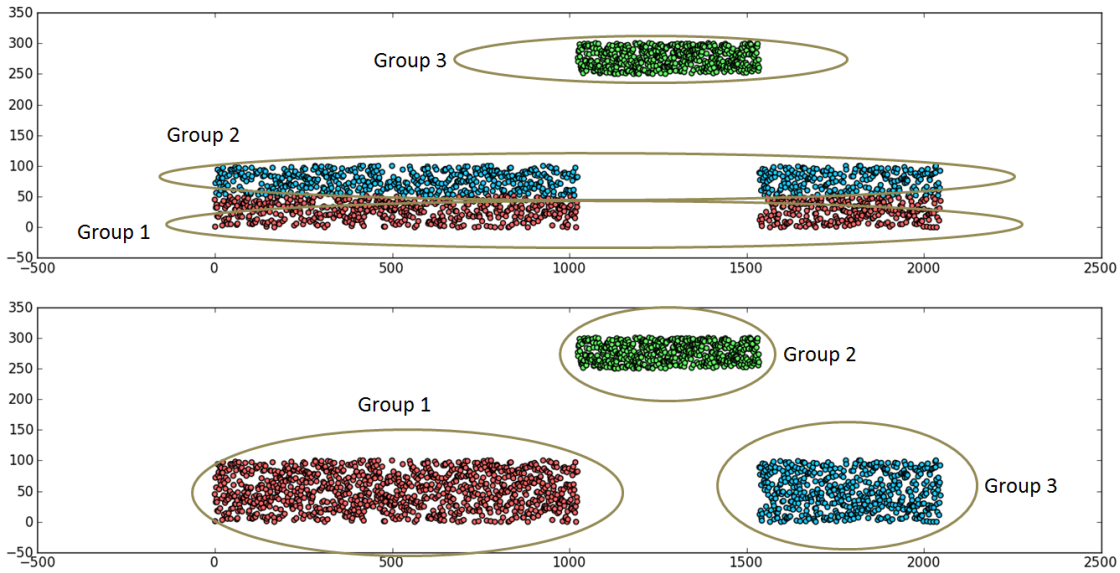

Fig. 11: Comparison between HMM with three states and k-means with $k = 3$, it should be noted that HMM takes the temporal dimension into account while grouping. Top: HMM with 3 states in temporal sequence: Bottom: Cluster with three groups

classes similar to the object oriented programming paradigm. Relationships between the concepts, class and instance are represented by connecting edges. Edges can be uni- or bi-directional and also allow transitive transition between the concepts. This transitivity enables to reason over the graph. In Figure 12 a simple semantic model is shown, where classes are coloured yellow and instances blue. With the help of query and reasoning languages it can be deducted that the higher-level abstraction of the concept ”Storm” has to be created by lower-level abstractions, in this Figure this is modelled by the class instances ”Cold” and ”Windy”.
The Data that is represented in a semantic representation usually follows some schema or meta-models from a certain domain. A common schema in the domain of sensor networks is the semantic sensor network ontology [10]. The usage of domain ontologies increase the interoperability of data from different sources by applying a common model.

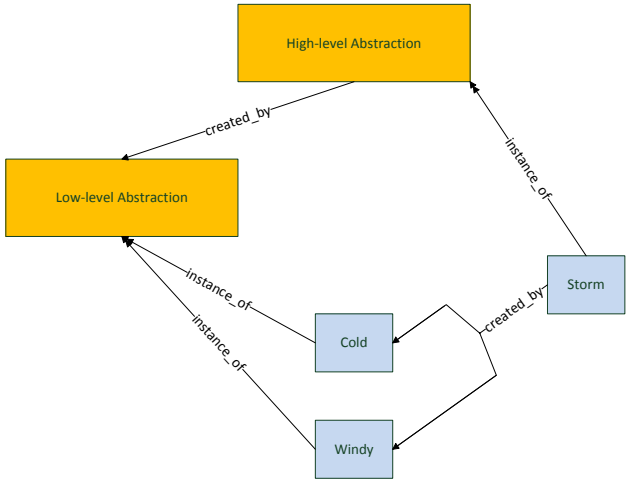

Fig. 12: A simple linked graph containing classes and instances that are linked together via properties.

IV. State of the Art in Information Abstraction

In this section current approaches for Information Abstraction from sensor data are presented and discussed. This discussions are divided to technical solutions and research approaches. Technical solutions are usually software and/or libraries that can be downloaded and used by the end-user. We focus our selection of approaches on software that are mainly used in the scientific community or are developed by scientific research groups. The technical solutions provide the methods and techniques that have been introduced and can be composed for special purposes. The research approaches that are also presented in this section have the goal to abstract from raw sensor data to higher-level abstraction.

A. Technical Solutions

RapidMiner is a software tool that provides methods for machine-learning, data-mining and statistical analysis. The tool follows the ETL (extract, transform, load) paradigm where data importers, operators and visualisation tools are represented as building blocks that can be stacked together. RapidMiner was developed for easy and rapid prototyping of data analytic chains. It enables the orchestration of the blocks using an interactive user interface. Therefore no programming skills are required to perform mining and processing tasks.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2015.2411227, IEEE Internet of Things Journal

The free open-source version of RapidMiner is a workstation application for rapid prototyping, but lacks features for scalability and real-time stream processing. Some approaches are introduced [41] to extend RapidMiner to support big data analytics. Others try to enable real-time streaming support [7]; however the algorithms used in RapidMiner might only be applicable in batch processing scenarios.

WEKA [26] is a similar toolbox with a strong focus on data mining tasks that can analyse static data. However, WEKA misses features for real-time stream handling. The MOA project [6], an advancement of WEKA for streaming data is able to handle streaming data including data from social media. However, MOA follows a centralized approach and lacks scalability for large-scale applications.

SAMOA [12] is a project that merges streaming data analysis techniques from MOA with distributed processing engines such as Apache Storm and Apache S4.

Orange [13] also follows a visual programming approach but additionally allows programming scripts in Python language. The focus of the orange toolbox lies more on the data visualisation rather than large-scale or real-time analytics.

B. Research Approaches

In the following we present research approaches that are used to transform raw sensor data to higher-level abstractions. The selection describes the different approaches from different domains and also discusses a broad application usage of information abstraction. This includes very domain-specific approaches as in vehicle detection and classification to very higher-level architectures.

GeoSensor Data Abstraction for Environmental Monitoring Application: Jung and Nittel [32] focus on providing data abstraction for environmental observation applications. Monitoring applications usually produce large volumes of heterogeneous data gathered from sensors distributed over large spatial areas. The authors state that the query distribution and processing over raw sensor data is too slow for real-time applications and therefore abstraction methods are required to make the data available for interpretation.

The authors introduce ”Slope Grid for Sensor Data Abstraction (SGSA)” abstraction method using several techniques to represent the gathered data on a map divided by a grid, where the grid represents the abstracted data as a slope that contains further information such as minimum, maximum and direction of natural phenomena such as wild fire.

Making Sense of Sensor Data Using Ontology: A Discussion for Road Vehicle Classification: Stocker *et al.* [44] detect and classify different types of road vehicles passing a street using vibration sensors and machine-learning algorithms. The objectives of the work attempts to acquire knowledge represented in an ontology by creating abstractions from the physical sensor layer and the sensor data layer.

At first, the data is pre-processed by applying a bandpass filter to the raw vibration sensor signal, filtering out the relevant frequencies triggered by cars passing the road. The bandpass filter is realised using fast Fourier transformation and summarising the values of a time window to provide input for the detection and classification using machine-learning methods.

Stocker *et al.* use a Multi Layer Perception (MLP) neural network classifier to detect and classify the different patterns gained after the pre-processing step to class vehicles based on their weight and length into the classes light vehicle and heavy vehicle. Due to the nature of MLP, a significant amount of training data is required. Training data has to be also manually annotated. The authors used the video data from a camera, mounted near to the vibration sensor to validate and classify a sample data set that is used as training data.

The outcome of the classification process is then transferred into an ontology representation. The authors use rule based inference to map the outcome of the classifier to the ontology. The ontology consists of concepts such as feature of interest (vehicle type) and observation result time. For each classified car, an individual is created in the ontology with the relevant context information.

Pattern-based event detection in sensor networks: Xue *et al.* [51] create abstractions from the raw sensor data using generic patterns that are utilised to report interesting events. In contrary to threshold based frameworks where events or abstractions are generated based on a certain threshold, the proposed work stores and communicates only the shape of the signal (data). The patterns capture the semantics of events and are more reliable than transmitting and processing raw data. The authors represent many events in real-world applications such as surveillance and pervasive applications in five basic patterns: horizon, slope, oscillation, jump and spike pattern, that depending on the context are sufficient to abstract from the real world data to represent any occurring event. The pre-processing phase of the approach can include distributed (over the underlying sensor network) mathematical computations to filter out noise before matching the raw data to the basic patterns by applying average- and/or min/max-computations. The mapping from raw data to pattern representations use in-network processes that run on the sensor nodes. To lower the size that is needed to store and communicate the patterns over constrained devices such as sensor nodes, the compression and dimensionality reduction techniques such as piecewise constant approximation and piecewise linear regression are used in this work.

An Experiment in Hierarchical Recognition: Gordon *et al.* [23] present an experiment to recognize group activities such as meetings, presentations and coffee breaks. The authors differentiate their work on different levels of abstractions, from lower-level abstractions e.g sensor measurements and medium-level abstractions e.g activities such as walking to higher-level abstractions such as meeting or coffee break.

Their approach utilises a hierarchical model where sensor nodes are on the bottom of the hierarchy and more powerful nodes e.g smart phones are on the top hierarchy levels. The higher the data is processed through the information hierarchy more context is considered and higher-level abstractions are created. On the sensor node level, data pre-processing techniques are applied on smart phones. For this purpose, feature extraction and classification are used. However, the work does not present any evaluation which they plan to carry out in

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2015.2411227, IEEE Internet of Things Journal

TABLE I: Overview of the research approaches and their selected methods and algorithms to abstract the data

Approach	Scenario	Pre-Processing	Dimensionality Reduction	Feature Extraction	Abstraction / Inference	Representation
GeoSensor	Geo-Spatial Data	Min-Max	X	Contour Map	X	Slope Grid (Graphical)
Road Vehicle Classification	Vehicle Detection	Bandpass	FFT	Supervised Learning (MLP Classification)	Inference	SSN
Pattern based event detection	Generic Event Detection	In-network aggregation	PCA	Shapes of Patterns	X	X
Hierarchical Recognition	Group Activities	Average / Variance	X	Local classification: kNN, DT	Global classification	X
Octopus	Smart Building	X	X	X	Solvers	Hierarchical Model
Envision	Environmental Decision Support	X	X	X	Semantic Rules	Semantic Representation
Information Abstraction	Environmental Data Abstraction	X	SensorSAX	k-means	Extended PCT	X

future research work.

Octopus: Smart Buildings, Sensor Networks and the Internet of Things Octopus [18] attempts to bridge the gap between the data level requirements and the Internet of Things. The authors introduce a system that creates data abstractions from sensor measurements and links it with physical objects and phenomena that are represented in a model. The model is divided into different layers, similar to the work by Gordon *et al.* [23]. Higher layers represent more-abstracted information and include the context of the physical object. The Octopus platform introduces solvers that abstract and link from lower layers to higher layers. Solvers represent the operators that are used on certain sensors to achieve the information extraction. A sample solver for "Talk Attendance" includes models to aggregate the information from sensors in a meeting room measuring the usage of seats and also modules to integrate calendar events. The paper describes a higher-level architecture of the approach, however does not go into detail how the abstraction is achieved in an automated manner.

Semantic Event Processing in Envision: The Envision framework [35] combines semantic models and complex event processing via rules to infer events in real-time. The approach introduces event processing services (EPS) that translate the raw data into semantic events. The semantic ontologies of Envision represent the instantiated events inferred by the EPS but also the patterns and rules that led to them. The system is semi-automatic; rules and patterns have to be designed via an interface with the Event Pattern Language (EPL). Similar to the Octopus framework, Envision describes an architectural view, but does not go into detail of aspects how the system can be autonomous. Especially in cases where there are large numbers of different sensors, the manual annotation of event processing services is not feasible.

Semantic Perception: Converting Sensory Observations to Abstractions: Henson *et al.* [29] use abduction reasoning to infer abstractions from current sensor observations. They utilise the parsimonious covering theory (PCT) that is predominantly used in the medical domain to find the best explanation of a disease based on a set of observations made by a physician.

A PCT-based model is represented by a uni-directional graph that connects diseases with observations that are likely to lead to the particular disease. Henson *et al.* introduce an ontology that is used for reasoning from observations that are made from particular abstractions. The reasoning process follows abductive reasoning method, where the abstraction that has the most measured observations is chosen. The ontology and the concept of an abductive reasoner are described and and examples are made. However, the connection between abstraction and observation is created and maintained in the system.

Information Abstraction for Heterogeneous Real World Internet Data In our recent work [5] and [19], we extend Henson *et al.* work with a method to model the graph in an automated manner using probabilistic graph modelling techniques and machine-learning methods. Our proposed method finds the significant measurement data and autonomously generate a PCT graph linking observations and abstractions. The ontology is divided into different levels of abstraction, namely lower-level and higher-level abstractions. Lower-level abstractions represent single events measured by a particular sensor, higher-level abstractions are aggregated inferred abstractions incorporating several data sources and processing steps based on the model by [23].

The model uses a clustering algorithm to find similar events to generate the first unlabelled lower-level abstractions. A hidden-Markov model is used to include the time dimension to infer relationships between abstractions over time. A rule-based engine is used to label the abstractions. The approach still requires a priori knowledge such as the labelling rules, however aims to provide autonomous extensions in the future.

C. Discussion

As shown in Table 1, there is currently no approach that implements the complete proposed workflow for converting raw data into machine interpretable abstractions. However, the technical approaches that have been introduced allow to develop the required algorithms and methods to implement all required steps. Nevertheless, the current research approaches only pick certain components to fulfill the goal in their application domain. This leads to the issue of having only few domain-independent approaches to process the IoT [19, 35]. With respect to the Big Data issues and the large heterogeneous volume of data that has to be processed, the domain-dependent approaches are not suitable solutions for the problem. There is a need for approaches that are able to automatically select algorithms and tune the required parameters based on the characteristics of the data. It might be not possible to choose methods that will lead to 100% certain results; however, a pre-selection of methods that can be provided to a data analyst to support rectification of the algorithms which can lead to a semi-automated information abstraction. There are also issues related to high-performance

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2015.2411227, IEEE Internet of Things Journal

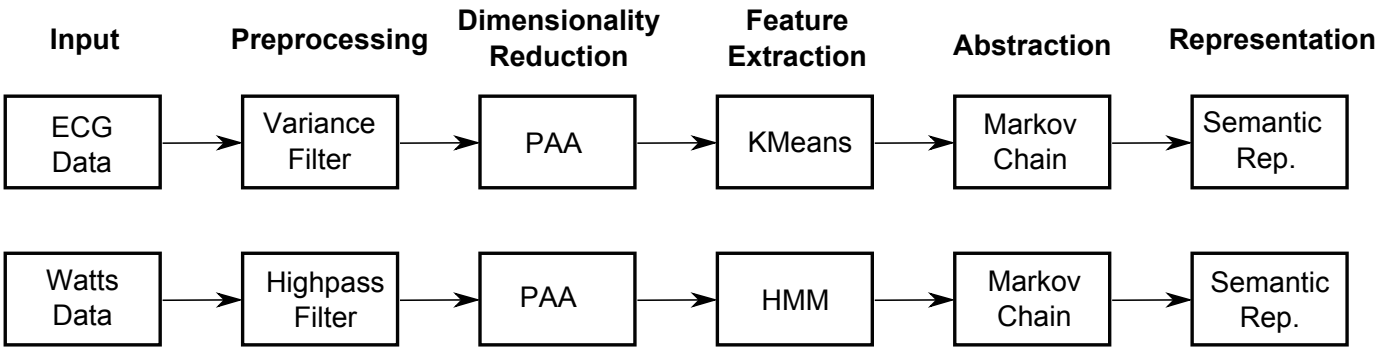

Fig. 13: The workflow of the data abstraction and the selected algorithms for both use cases

computing and efficient processing of very large amounts of data. In constrained IoT environments, energy efficiency of data collection, communication and in-networking processing are also important issues. We have discussed some of these issues in our previous work reported in [19, 20]. In this paper, our main focus has been on data analytics for IoT and extracting meaningful machine/human interpretable knowledge from the data. In this regard, we define the following three requirements for developing data analytics tools for processing IoT data.

- (Automatic or semi-automatic) algorithm selection and parameter tuning for various application scenarios.
- Decision support for data analysts to cope with the large volume of size, diversity and velocity of the data.
- User interfaces that allow the examination of several data streams and the application of various (pre-learned) methods to cope with the data

In our current research, we have developed a knowledge acquisition toolkit that aims to fulfill the three mentioned requirements. The tools integrates the common methods that are shown in the state-of-the-art section and provide an all-in-one customisable tool. At the current stage, the tool allows a selection of several different data sources and applying various algorithms. Nevertheless, the current version of the tool does not incorporate automated mechanisms to find the best methods based on the input data. Focusing on adaptive methods for data processing is investigated in our ongoing work and additional features will be included in the next version of the tool. The next section describes our data analytics tool and presents a use-case demonstration.

V. Use-Case Scenario

We have chosen two real world data sets, one from a smart office scenario and the other from the medical domain. The smart office scenario exhibits the usage of the IoT to get insights from power consumption patterns of people in the office that can be leveraged to reduce energy consumption by turning of computer workstations and lights. The medical scenario shows how data that is captured by a pacemaker sensor can be used to help medical advisors to find outliers and possible distortions of a patient's heart activity.

To get a better understanding and to motivate analysts to apply the mentioned algorithms on their data, we introduce the knowledge acquisition toolkit (KAT). KAT provides algorithms for numerical and textual data analysis that can help to extract meaningful information and represent it in a human-readable or machine interpretable format. More technical details can be found on the official website: *http://kat.ee.surrey.ac.uk*.

We apply the introduced algorithms on two data sets from different domains and explain the selection of the applied algorithms to give an overview where and how the particular algorithms can be applied on real world data. In Figure 13 we show the workflow chain of the algorithms that are going to be applied on the data.

The first data set is from our own sensor test bed deployed at the University of Surrey [38]. The data comes from a sensor node in front of one of the authors desk monitoring the power consumption of a workstation connected to the power meter. The raw data set was captured over a month and contains 274960 samples.

The second data set is from the machine-learning data set repository maintained by the University of California, Irvine and represents an electrocardiogram (ECG) dataset with 3751 samples, published by Olszewski [40] and found at the UCR time-series Classification/Clustering page[3].

In Figure 14, the file data loading screen of KAT is shown, on the left window the ECG is presented, on the right the power consumption data is displayed. The tool supports different input sources and formats such as CSV, EXCEL, SQL, CKAN API [49]. In the case that several categories inside a data source are available, the user can select the categories on which the algorithms should be applied on. Our aim in this example use-case is to find the outlier in the ECG dataset happening at around sample 2400-2600 and to cluster the repetitive "work day" behaviour in the watts dataset and find a semantic representation for it.

First we apply pre-processing filters to the data. On the ECG data, we choose the variance filter to reduce the dataset to samples with a high volatility in windows. The windows size can be defined in KAT. On the watts data, we choose to filter the noise at the bottom of the data, to minimize the "background power consumption" and focus on power peaks

[3]http://www.cs.ucr.edu/ eamonn/timeseriesdata

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2015.2411227, IEEE Internet of Things Journal

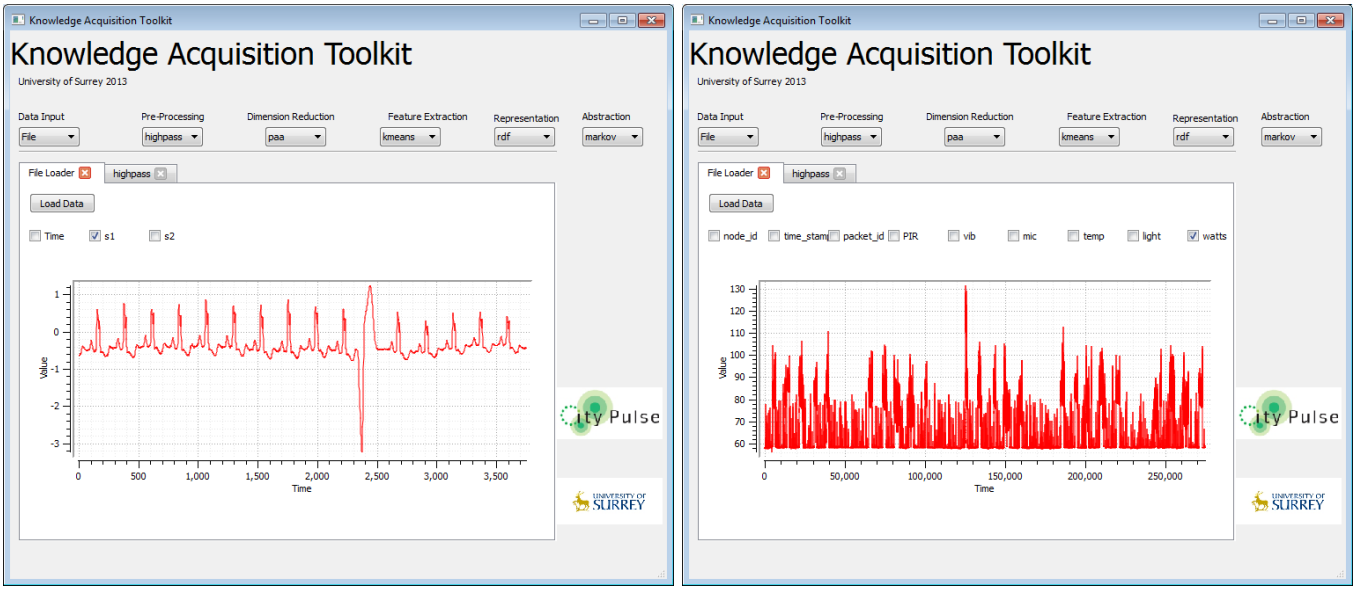

Fig. 14: ECG data (left) and power consumption data (right) loading screen

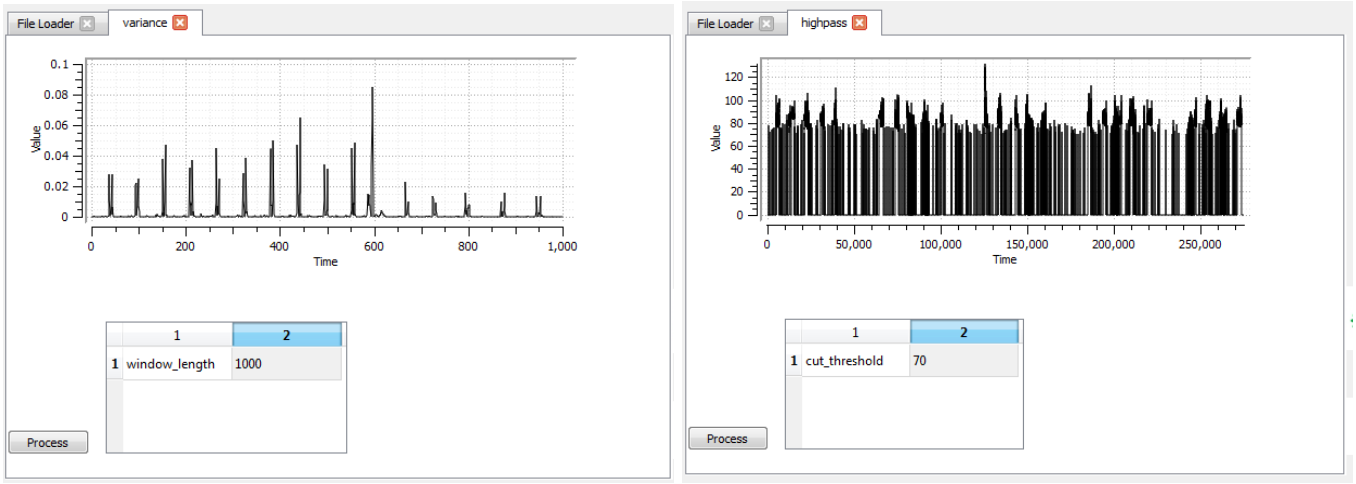

Fig. 15: Left: ECG data after applying variance filter. Right: Watts data after applying highpass filter.

(=possible presence in an office) with the help of a high-pass filter. The processed data can be seen in Figure 15

To eliminate rigorousness and redundancy we reduce the dimensionality of the data. For both data sets we use the Piecewise Aggregate Approximation technique (PAA). The interesting patterns of the data now becomes more visible, as shown in Figure 16. In the ECG dataset it is noticeable that there is a peak that stands out from the others. In the watts dataset it can be seen that there is some regularity behind the data. The reader can easily infer, that the power consumption is high during a work day at office hours and low between the work days (between peaks) and on the weekend (long gaps). In both cases the amount of data samples has been reduced significantly, 100 out of 274960 samples for the watts data and 50 out of 3751 for ECG data. This will ease the processing of the following processing steps. The more processing intensive cluster algorithm can now operate on less data samples to provide the first level of lower-level abstractions. We run a k-means algorithm on both datasets. On the ECG dataset we run k-means with $k = 3$, representing low activity (called the PR-Interval) in group 0, peaks (called the QT Interval) in group 1 and outliers in group 2. On the watts dataset, we use HMM to group it into two temporal groups, a group representing a work day and a group representing the weekend(probably no presence in the office). The clustering of the data is represented in Figure 17. After the clustering step, we discover temporal relations between the clustered data. For temporal relation discovery, we use a Markov chain approach to calculate the probabilities of the occurrences of the groups. To ease the understanding, we labeled the groups in KAT. The results are shown in Figure 18. A possible representation of the abstractions that have been acquired through the overall process can be seen in Figure 19. KAT allows to define parameters how granular the data should be

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2015.2411227, IEEE Internet of Things Journal

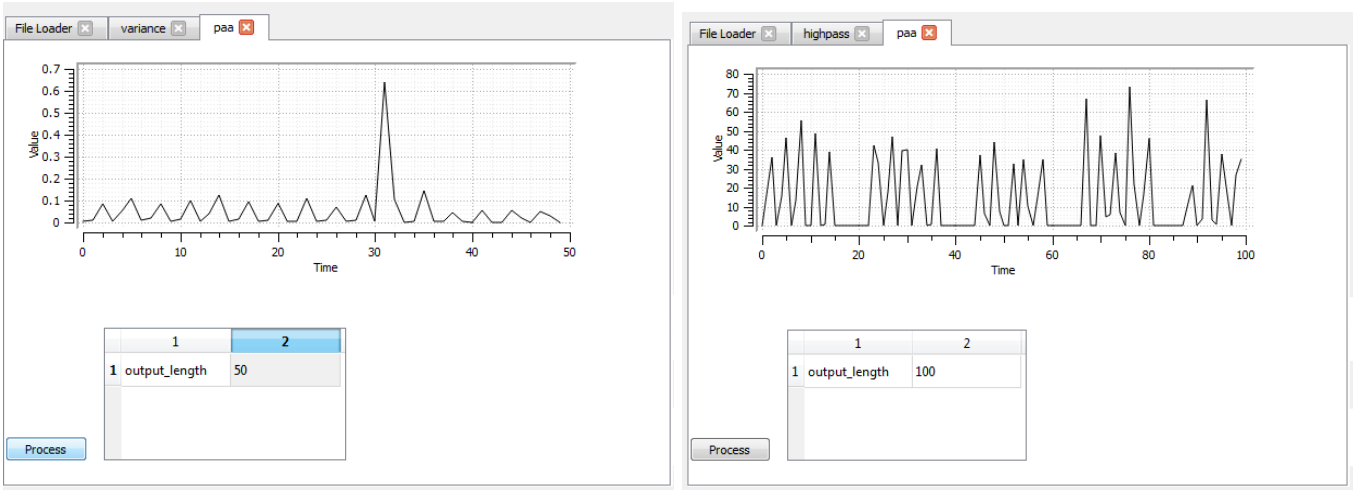

Fig. 16: Left: ECG data after applying PAA, revealing the outlier and suppressing the background noise. Right: Watts data after applying PAA, revealing the regular pattern of a workday

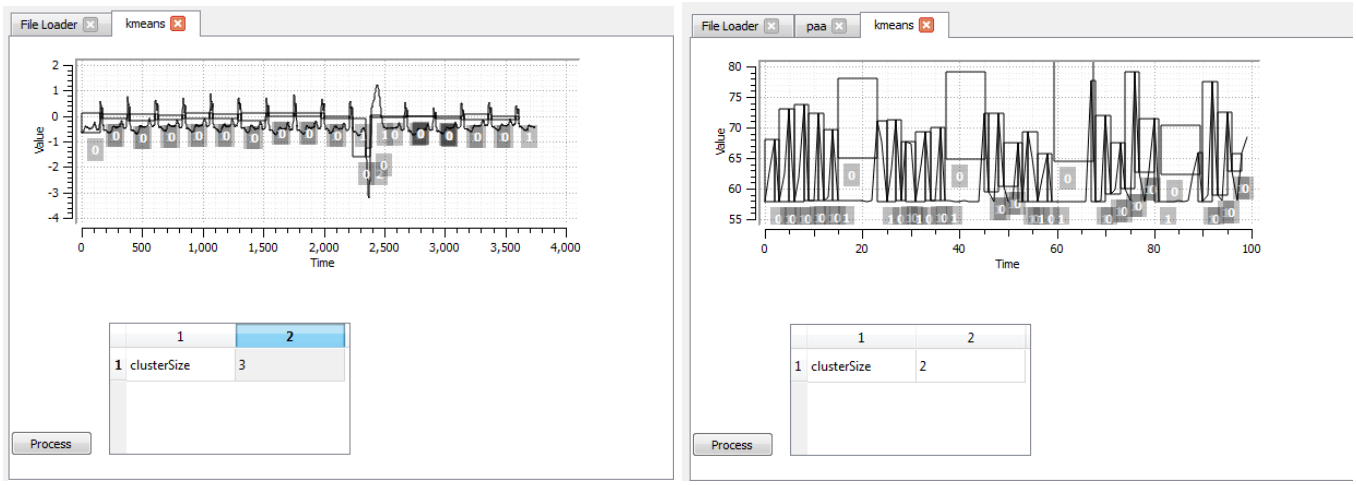

Fig. 17: Left: ECG data after applying k-means with $k = 3$, grouping the data into groups of data with low values(0), high-values(1) and outliers(2). Right: Watts data after applying HMM with 2 states, grouping the data into two groups of low power (0) and high power (1) consumption

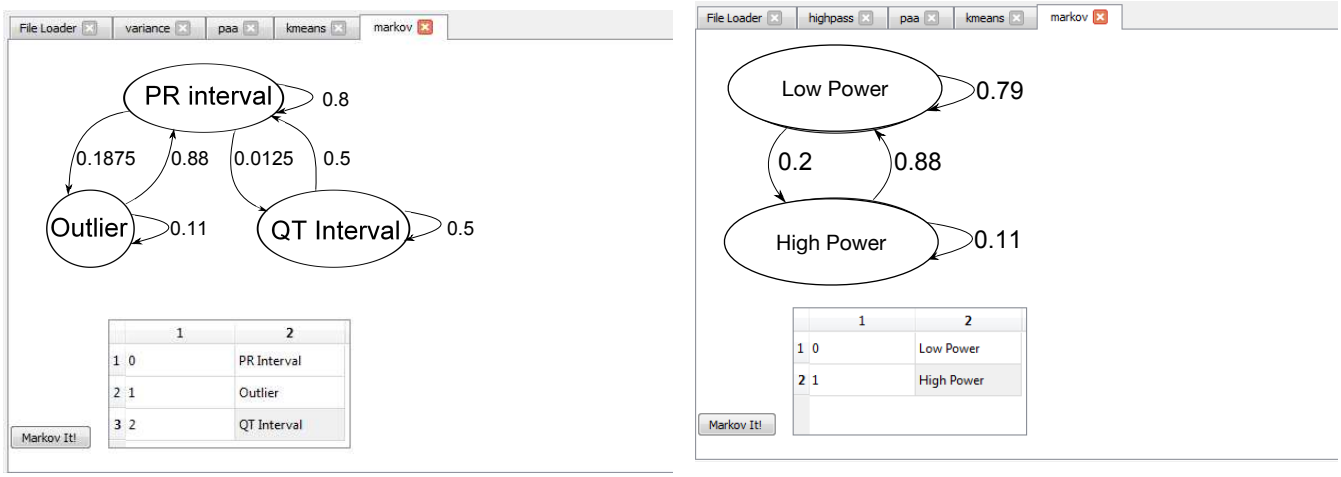

Fig. 18: Left: ECG data after applying PAA, revealing the outlier and suppressing the background noise. Right: Watts data after applying PAA, revealing the regular pattern of a workday

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2015.2411227, IEEE Internet of Things Journal

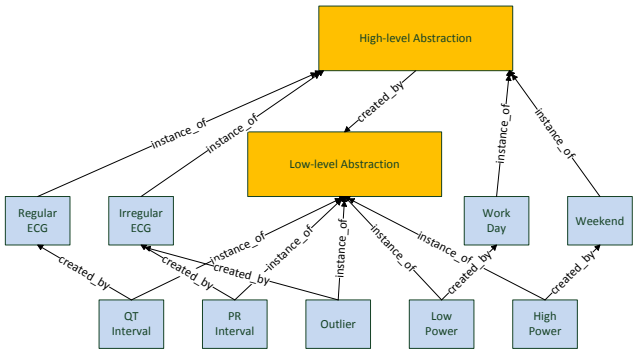

Fig. 19: Linked representation of the information acquired through the abstraction process

presented. For instance, it would also be possible to include the raw data coming from the sensors that lead to the lower-level abstractions, but for presentation reasons we only include the information from the lower-level abstractons and onwards. Despite the information that has been acquired it would also be possible to include the provenance information e.g parameters and operators that led to the different abstractions. An ongoing research project that captures the provenance parameters is the PROV-O ontology[4] and will be included in future work.

VI. Conclusions

In this paper we present a survey of techniques and methods to process and transform raw sensor data into higher-level abstractions that are human and/or machine-understandable. We explain a workflow for information abstraction and describe approaches that can be applied during different stages of the proposed workflow. The paper introduces different techniques from signal processing, machine-learning and the semantic web that can be utilised for sensor data processing. Then we describe existing software tools that can be used to implement the processes and examine current research work from different domains that can be used for information abstraction in the IoT domain. We discuss the current Big IoT Data challenges and describe requirements such as high scalability and real-time processing of the data. We highlight existing research approaches and examine their advantages and shortcomings. We have presented an integrated IoT data analytics tool called Knowledge Acquisition Toolkit (KAT). KAT can be used to import sensor data from various sources and enables processing the raw sensor data and creating abstractions using the common data analysis methods that are discussed in the state-of-the-art. Our future research will focus on extending KAT with analysis methods to work with high performance computing and Big Data analytics tools such as Hadoop. Besides the size and scalability extensions, extensions on dynamicity handling and automated selection will enhance the large-scale data analysis in this domain. We are also currently developing data processing and abstraction techniques that are adaptive to changes in the input data and have the capacity of handling multi-modal data without the need of domain knowledge. These will also be integrated into KAT once the new adaptive methods have been finalised, presented and peer reviewed.

[4]http://www.w3.org/TR/prov-o/

Acknowledgment

This paper describes work undertaken in the context of the EU FP7 CityPulse project contract number: 609035.

Frieder Ganz received his PhD from the Centre for Communication Systems Research at the University of Surrey. His research is focused on information abstraction and extracting machine-interpretable knowledge from large volumes of sensory data using stream processing and machine learning techniques.

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

Payam Barnaghi is a Lecturer (Assistant Professor) with the Institute for Communication Systems, University of Surrey, Surrey, U.K. His research interests include machine learning, Internet of Things, semantic Web, Web services, information centric networks and information search and retrieval.

Francois Carrez received a PhD in Theoretical Computer Science from the University of Nancy France in 1991. For 18 years he worked for the Alcatel Research centre in Paris in areas such as Security, Distributed Artificial Intelligence, AdHoc Networking and Semantics. He joined the University of Surrey in 2006 and is currently a Senior Research Fellow at the Centre for Communication Systems Research (CCSR). His research interests include: Semantic Web, Internet of Things, Activity Theory and Social and Behavioural Sciences.

REFERENCES

[1] Karl Aberer, Manfred Hauswirth, and Ali Salehi. "A middleware for fast and flexible sensor network deployment". In: *Proceedings of the 32nd international conference on Very large data bases*. VLDB Endowment. 2006, pp. 1199–1202.

[2] Charu C Aggarwal, Naveen Ashish, and Amit Sheth. "The Internet of Things: A Survey from the Data-Centric Perspective". In: *Managing and Mining Sensor Data*. Springer, 2013, pp. 383–428.

[3] Ian F Akyildiz et al. "Wireless sensor networks: a survey". In: *Computer networks* 38.4 (2002), pp. 393–422.

[4] Anish Arora et al. "Exscal: Elements of an extreme scale wireless sensor network". In: *Embedded and Real-Time Computing Systems and Applications, 2005. Proceedings. 11th IEEE International Conference on*. IEEE. 2005, pp. 102–108.

[5] Payam Barnaghi et al. "Computing perception from sensor data". In: *Sensors, 2012 IEEE*. IEEE. 2012, pp. 1–4.

[6] Albert Bifet et al. "MOA: a real-time analytics open source framework". In: *Machine Learning and Knowledge Discovery in Databases*. Springer, 2011, pp. 617–620.

[7] Christian Bockermann and Hendrik Blom. "Processing data streams with the rapidminer streams-plugin". In: *Proceedings of the RapidMiner Community Meeting and Conference*. 2012.

[8] Guanling Chen and David Kotz. "Context aggregation and dissemination in ubiquitous computing systems". In: *Mobile Computing Systems and Applications, 2002. Proceedings Fourth IEEE Workshop on*. IEEE. 2002, pp. 105–114.

[9] Jim Cicconi At&T Cisco IBSG et al. *The Internet of Things Infographic*. Aug. 2012.

[10] Michael Compton et al. "The SSN ontology of the W3C semantic sensor network incubator group". In: *Web Semantics: Science, Services and Agents on the World Wide Web* 17 (2012), pp. 25–32.

[11] Joëlle Coutaz et al. "Context is key". In: *Communications of the ACM* 48.3 (2005), pp. 49–53.

[12] Gianmarco De Francisci Morales. "SAMOA: a platform for mining big data streams". In: *Proceedings of the 22nd international conference on World Wide Web companion*. International World Wide Web Conferences Steering Committee. 2013, pp. 777–778.

[13] Janez Demšar et al. "Orange: Data Mining Toolbox in Python". In: *Journal of Machine Learning Research* 14 (2013), pp. 2349–2353. URL: http://jmlr.org/papers/v14/demsar13a.html.

[14] Arnoldo Díaz-Ramírez et al. "Wireless sensor networks and fusion information methods for forest fire detection". In: *Procedia Technology* 3 (2012), pp. 69–79.

[15] Jakob Eriksson et al. "The pothole patrol: using a mobile sensor network for road surface monitoring". In: *Proceedings of the 6th international conference on Mobile systems, applications, and services*. ACM. 2008, pp. 29–39.

[16] Jonny Farringdon et al. "Wearable sensor badge and sensor jacket for context awareness". In: *Wearable Computers, 1999. Digest of Papers. The Third International Symposium on*. IEEE. 1999, pp. 107–113.

[17] Davide Figo et al. "Preprocessing techniques for context recognition from accelerometer data". In: *Personal and Ubiquitous Computing* 14.7 (2010), pp. 645–662.

[18] Bernhard Firner et al. "Poster: Smart Buildings, Sensor Networks, and the Internet of Things". In: *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*. SenSys '11. Seattle, Washington: ACM, 2011, pp. 337–338. ISBN: 978-1-4503-0718-5. DOI: 10.1145/2070942.2070978. URL: http://doi.acm.org/10.1145/2070942.2070978.

[19] Frieder Ganz, Payam Barnaghi, and Francois Carrez. "Information Abstraction for Heterogeneous Real World Internet Data". In: (2013).

[20] Frieder Ganz, Payam Barnaghi, and Francois Carrez. "Multi-resolution Data Communication in Wireless Sensor Networks". In: *Proceedings of the IEEE World Forum on Internet of Things WF-IoT*. IEEE. 2014.

[21] Hassan Ghasemzadeh et al. "Sport training using body sensor networks: A statistical approach to measure wrist rotation for golf swing". In: *Proceedings of the Fourth International Conference on Body Area Networks*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering). 2009, p. 2.

[22] Andrew R Golding and Neal Lesh. "Indoor navigation using a diverse set of cheap, wearable sensors". In: *Wearable Computers, 1999. Digest of Papers. The Third International Symposium on*. IEEE. 1999, pp. 29–36.

[23] Dawud Gordon et al. "An experiment in hierarchical recognition of group activities using wearable sensors". In: *Modeling and Using Context*. Springer, 2011, pp. 104–107.

[24] Alfred Haar. "Zur Theorie der orthogonalen Funktionensysteme". German. In: *Mathematische Annalen* 69.3 (1910), pp. 331–371. ISSN: 0025-5831. DOI: 10.1007/BF01456326. URL: http://dx.doi.org/10.1007/BF01456326.

[25] David L Hall and James Llinas. "An introduction to multisensor data fusion". In: *Proceedings of the IEEE* 85.1 (1997), pp. 6–23.

[26] Mark Hall et al. "The WEKA data mining software: an update". In: *ACM SIGKDD Explorations Newsletter* 11.1 (2009), pp. 10–18.

[27] John A Hartigan and Manchek A Wong. "Algorithm AS 136: A k-means clustering algorithm". In: *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 28.1 (1979), pp. 100–108.

[28] Wendi Rabiner Heinzelman, Anantha Chandrakasan, and Hari Balakrishnan. "Energy-efficient communication protocol for wireless microsensor networks". In: *System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on*. IEEE. 2000, 10–pp.

[29] Cory Henson, Amit Sheth, and Krishnaprasad Thirunarayan. "Semantic perception: Converting sensory observations to abstractions". In: *Internet Computing, IEEE* 16.2 (2012), pp. 26–34.

[30] Victoria J Hodge and Jim Austin. "A survey of outlier detection methodologies". In: *Artificial Intelligence Review* 22.2 (2004), pp. 85–126.

[31] Tim Tau Hsieh. "Using sensor networks for highway and traffic applications". In: *Potentials, IEEE* 23.2 (2004), pp. 13–16.

[32] Young Jin Jung and Silvia Nittel. "Geosensor Data Abstraction for Environmental Monitoring Application". In: *Geographic Information Science*. Springer, 2008, pp. 168–180.

[33] Eamonn J Keogh and Michael J Pazzani. "Scaling up dynamic time warping for datamining applications". In: *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2000, pp. 285–289.

[34] Jessica Lin et al. "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. 2003, pp. 2–11.

[35] Alejandro Llaves et al. "Semantic event processing in envision". In: *Proceedings of the 2nd International Conference on Web Intelligence, Mining and Semantics*. ACM. 2012, p. 25.

[36] J Mantyjarvi. "Sensor-based context recognition for mobile applications". In: *VTT PUBLICATIONS* (2003).

[37] Luca Mottola and Gian Pietro Picco. "Programming wireless sensor networks: Fundamental concepts and state of the art". In: *ACM Computing Surveys (CSUR)* 43.3 (2011), p. 19.

[38] Michele Nati et al. "A Framework for Resource Selection in Internet of Things Testbeds". In: *Testbeds and Research Infrastructure. Development of Networks and Communities*. Springer, 2012, pp. 224–239.

[39] Reza Olfati-Saber. "Distributed Kalman filter with embedded consensus filters". In: *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC'05. 44th IEEE Conference on*. IEEE. 2005, pp. 8179–8184.

[40] Robert T Olszewski. *Generalized feature extraction for structural pattern recognition in time-series data*. Tech. rep. DTIC Document, 2001.

[41] Zoltán Prekopcsák et al. "Radoop: Analyzing big data with rapidminer and hadoop". In: *Proceedings of the 2nd RapidMiner Community Meeting and Conference (RCOMM 2011)*. 2011.

[42] Amit Sheth, Cory Henson, and Satya S Sahoo. "Semantic sensor web". In: *Internet Computing, IEEE* 12.4 (2008), pp. 78–83.

[43] Stephan Sigg et al. "Investigation of context prediction accuracy for different context abstraction levels". In: *Mobile Computing, IEEE Transactions on* 11.6 (2012), pp. 1047–1059.

[44] M Stocker, M Rönkkö, and M Kolehmainen. "Making sense of sensor data using ontology: a discussion for road vehicle classification". In: *Proceedings of the (iEMSs) International Congress on, Environmental Modelling and Software*. 2012.

[45] Zbigniew R Struzik and Arno Siebes. "The Haar wavelet transform in the time series similarity paradigm". In: *Principles of Data Mining and Knowledge Discovery*. Springer, 1999, pp. 12–22.

[46] Harald Sundmaeker et al. "Vision and challenges for realising the Internet of Things". In: *Cluster of European Research Projects on the Internet of Things, European Commision* (2010).

[47] Rui Tan et al. "Quality-driven volcanic earthquake detection using wireless sensor networks". In: *Real-Time Systems Symposium (RTSS), 2010 IEEE 31st*. IEEE. 2010, pp. 271–280.

[48] Hanbiao Wang, Deborah Estrin, and Lewis Girod. "Preprocessing in a tiered sensor network for habitat monitoring". In: *EURASIP Journal on Advances in Signal Processing* 2003.4 (2003), pp. 392–401.

[49] Joss Winn et al. "Open data and the academy: an evaluation of CKAN for research data management". In: (2013).

[50] Christopher R Wren and Emmanuel Munguia Tapia. "Toward scalable activity recognition for sensor networks". In: *Location-and context-awareness*. Springer, 2006, pp. 168–185.

[51] Wenwei Xue, Qiong Luo, and Hejun Wu. "Pattern-based event detection in sensor networks". In: *Distributed and Parallel Databases* 30.1 (2012), pp. 27–62.