Improve IoT/M2M data organization based on stream patterns

Mário Antunes*, Ricardo Jesus†, Diogo Gomes*, and Rui L. Aguiar*
*Instituto de Telecomunicações
Universidade de Aveiro
Aveiro, Portugal
Email: mario.antunes,dgomes,ruilaa@av.it.pt
†DETI
Universidade de Aveiro
Aveiro, Portugal
Email: ricardojesus@ua.pt

***Abstract*—The increasing number of small, cheap devices full of sensing capabilities lead to an untapped source of information that can be explored to improve and optimize several systems. Yet, as this number grows it becomes increasingly difficult to manage and organize all this new information. The lack of a standard context representation scheme is one of the main difficulties in this research area. With this in mind we propose a tailored generative stream model, with two main uses: stream similarity and generation. Sensor data can be organized based on pattern similarity, that can be estimated using the proposed model. The proposed stream model will be used in conjunction with our context organization model, in which we aim to provide an automatic organizational model without enforcing specific representations. Moreover, the model can be used to generate streams in a controlled environment. Useful for validating, evaluating and testing any platform that deals with IoT/M2M devices.**

***Index Terms*—Stream Mining, Machine learning, IoT, M2M, Context awareness**

I. Introduction

Over the last years the Internet of Things (IoT) [1] has gained significant attention from both industry and academia. IoT has made it possible for everyday devices to acquire and store contextual data, in order to use it at a later stage. This allows devices to share data with one another, and even services on the Internet in order to cooperate and accomplish a given objective. A cornerstone to this connectivity landscape is machine-to-machine (M2M) [2]. M2M generally refers to information and communication technologies able to measure, deliver, digest and react upon information autonomously, i.e. with none or minimal human interaction.

Context-awareness is an intrinsic property of IoT and M2M scenarios [3]. Context-aware communications and computing have played a critical role in understanding sensor data. As discussed in [4] an entity's context can be used to provide added value: improve efficiency, optimize resources and detect anomalies. However, recent projects follow a vertical approach [5]–[7], devices/manufacturers can not share context information because each one uses a different structure, leading to information silos. This has hindered interoperability and the realisation of even more powerful IoT and M2M scenarios.

Context information is an enabler for further data analysis, potentially exploring the integration of an increasing number of information sources. The common definitions of context information [8], [9] are so broad that any data related to an entity can be considered context information. These definitions also do not provide any insight about the structure of context information. Currently there is no uniform way to share/understand vast amounts of IoT/ M2M data. It is unlikely that in the future a context representation standard will be widely adopted. First, the diversity of context representations, each one of them was designed for a specific usage and/or data types. Second, a widely adopted context representation does not completely solve the issue of knowledge extraction. Due to the vast amount of data it is extremely difficult to define *a priori* all the relations between information sources, patterns, and even possible optimizations.

Another important issue is the need for a new way to manage, store and process such diverse machine data: unconstrained, without limiting structures and with minimal human interaction. With this in mind we proposed a data organization model optimized for unstructured data [4], [10] that organizes context data based on semantic and stream similarity. Our model uses tailored features and unsupervised learning algorithms to automatically organize data.

In this paper we propose a stream model tailored specifically for stream similarity and to tackle the issue of propagating classification tags. We propose a generative model for stream characterization, that can be either used for stream similarity or generation. The objective is to use the previously mentioned model to organize sensor streams based on their patterns and improve the efficiency of our context representation model.

The remainder of this paper is organized as follows. In

Section II we discuss semantic similarity and present the most relevant methods. We discuss our generative model for stream characterization in Section III. The results of our evaluation are in Section IV. Finally, the discussion and conclusions are presented in Section V.

II. Background and Related Work

Context information is an enabler for further data analysis, potentially exploring the integration of an increasing number of information sources. As previously mentioned, common definitions of context information [8], [9], [11] do not provide any insight about its structure. In fact, each device can share context information with a different structure. For example, sensory and location information can be used to characterize an entity context, yet the two can have different structures. One important objective of context representation is to standardize the process of sharing and understanding context information. However, nowadays no widely accepted context representation scheme exists; instead there are several approaches to deal with context information. These can be divided into three categories: i) adopt/create a new context representation, ii) normalize the storing process through ontologies, iii) accept the diversity of context representations.

In [12] the authors analize two different projects related with context-awareness. One of the projects uses a single context representation scheme. The authors concluded that using a single context representation limits the relations that exist between all the data sources. As a consequence it becomes increasily difficult to detect and react to complex events. Furthermore, it limits the quantity of data that can be shared with other projects.

The second possibility would be employing ontologies to normalize the organization process. Each context representation is mapped into the internal data model through an ontology [13]. This type of platform supports several context representations, yet it is necessary to define a new ontology (mapping) for each new representation. Defining a new ontology is a tedious task that requires human intervention. Due to the diversity and scale associated with IoT/M2M scenarios it is extremely difficult to maintain this strategy. As an example, we can consider the lexical database WordNet [14]. WordNet is a manually-created hierarchical network of nodes (taxonomy), that due to funding and staffing issues is no longer accepting comments and suggestions[1]. It is extremely difficult to maintaining large databases of relations (of any type) if they depend on human input.

As an alternative, we accepted the diversity of context representation as a consequence of economic pressures, and devised a bottom-up model [4], [10], [15] to organize context information without enforcing a specific representation. Our organization model is divided into four main parts, as depicted in Figure 1.

[1]http://wordnet.princeton.edu/wordnet/

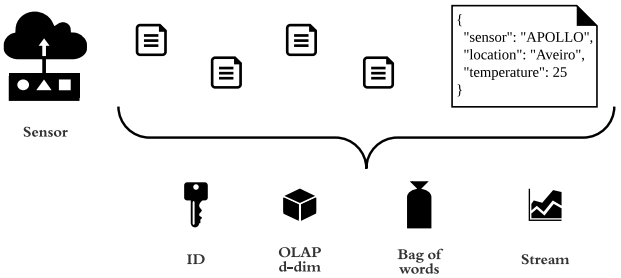


Fig. 1: Context organization model based on semantic and stream similarity.

The first two represent the structured component of our model and account for the source ID and fixed d-dimensions respectively. These d-dimensions allow human users to select information based on time, location or even other dimensions, and can be understood as an OLAP cube helping in the process of filtering information. The remaining parts of our model extract information from the content itself and organize it based on semantic and stream similarity. Our work on semantic similarity can be found in the following publications [4], [10]. The first steps towards a stream similarity model are given in this paper.

While there are several academic works based on stream prediction and mining [16], the same can not be said about stream similarity. Most methods are based on longest common sub-sequence algorithm [17], [18]. However, these methods are not ideal for IoT/M2M data for two main reasons. First, data acquired from IoT devices tend to be noisy, can be shifted in time and have different scales. Second, the vast number of IoT devices imply that there are several streams for the same phenomenon. Our objective is to learn a representation of the phenomenon, combining all the streams in a single model. Due to these reasons we devised our own generative model.

III. Generative model for stream characterization

Before discussing the inner works of our stream characterization model, let us discuss its origin. With the advent of IoT/M2M devices, context-aware platforms require novel organizational models, learning algorithms and proper testing. However, it is rather difficult to evaluate the accuracy of these systems when the environment is as dynamic and vast as the IoT/M2M environment. In order to properly test these platforms we require a controlled environment, in this context we require tools to control the input data.

There are some possibilities, one of the most common ones is to use several datasets gathered from actual sensors. Gathering, pre-processing, classifying and maintaining these datasets requires human intervention and are time-consuming. Furthermore, in order to guarantee that the tests cover all (almost) the possible inputs, large amounts of data are required. One alternative to this, is developing

a model that captures the information about a determined phenomenon and is able to generate several phenomena that are statistically similar. This was the drive to develop our stream characterization model. Apart from stream generation, our model structure makes it ideal to develop similarity metrics. We intend to explore the full capabilities of this model, as a tailored feature for IoT/M2M organization, in future publications.

This section will address two different but related ideas. First, we will present our generative model for stream characterization based on Markov Chains and detail its inner workings. Second, we will elaborate on a stream generator which uses this previously mentioned model.

A. Stream Characterization

Our approach is to model a stream's behaviour using a first order Markov Chain. Considering a perfect scenario where there are no noise or errors, most events would thus happen in a very predictable manner (i.e. without major variances). We could formulate our model as Equation 1, by knowing how probable it is for, at a given time instant x_{i-1} with a value of y_j, a stream at the time x_i has a value of y_k. In other words, the probability of having value y_k at a time instant x_i knowing its immediate predecessor.

$$P_i(y_k|y_j) \tag{1}$$

For the remainder of this paper we will call the succession of a value to the one following it (along the x axis) a jump or transition.

We could then argue that using the method above and knowing all the probabilities of all the jumps along the period of the event, we could represent it with quite high confidence. For the sake of argument, consider that we had at our disposal such a probability function as expressed above, and we were given a sequence of values representing an event. We would like to compute the similarity (S) between the sequence of values and the probability function. This can be achieved by verifying all the values of P_i for all transitions within a sequence's period, and either averaging them or using some other statistical indicator to get a representative, normalized value of the overall resulting probabilities (see Equation 2, where n represents the number of samples in the stream).

$$S = \frac{1}{n}\sum_{i=1}^{n} P_i \tag{2}$$

The probability function assigns high or low values to each jump of the sequence based on how well it relates to the events expressed by the probability function itself. If the sequence's values were off the event's, then the overall probability would be low. On the other hand if it was high, then we could be confident that this sequence is similar to the event represented by the function.

The problem arises as we notice that this perfect scenario is not possible in practical cases, and thus if we intend to use such a function as the one described above to represent a stream, we need to overcome three issues and make a few changes to its definition:

1) Streams representing the same events more commonly than not vary widely, for several reasons, such as noise, location, time of day, etc.
2) It is impractical, due to time and space constraints, to have a function mapping every set of points $((x_i, y_j), (x_{i+1}, y_k))$ that might appear in a stream;
3) Along the lines of the previous item, it is not reasonable to consider the continuous and/or infinite domain associated with most events (which would imply considering infinite values).

Our proposal solves these issues by overlaying a grid-like structure over the different values a stream takes along its period, effectively turning each (x_i, y_j) in the preceding discussion into a gap (as depicted in Figure 2). This gap gives rise to two other values that are now to be considered, Δx and Δy, each representing the resolution of their corresponding axis.

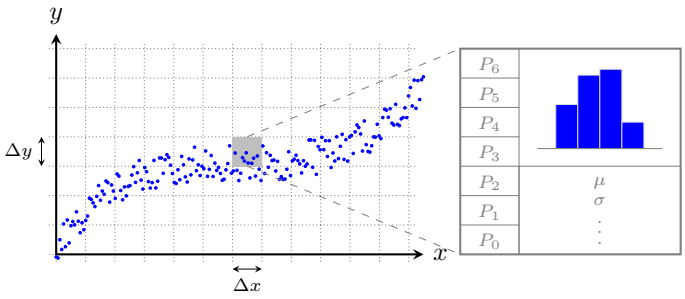


Fig. 2: Structure proposed to model stream information. A grid is overlayed over the streams, in order to build a matrix like structure where each slot contains a probability vector, an histogram of values, and other relevant statistical values (currently the mean and standard deviation of the values inside the slot).

Issue 1 can be solved by overlaying multiple streams representing a same event, and computing the probabilities that arise from their transitions. Issues 2 and partially 3 are solved by now considering jumps' areas instead of single values, in a sense discretizing both a stream's domain and codomain. By the law of large numbers and assuming that those streams do follow a pattern (even if with noise and/or erratic behaviour), one can be sure that eventually the probabilities will converge. Issue 3 can be further improved in the case of periodic streams. Given that most real scenarios are periodic to some extent, this assumption can be made without major disadvantages. Even if the sensor data has some seasonality property, we can generate a model based on the last k periods, or generating a stream model for each seasonality period. Furthermore, the period of a phenomenon can help to slip a stream according to its period and using it as the domain of the grid, it is possible to work even with infinite domains. Each stream's period is taken as a 1-period stream by itself.

This way we are capable of characterizing the underlying behaviour of some event, based on the behavioural patterns of some related streams. We say this method is Markov chains' based since it assumes that there is little to no knowledge lost by only considering direct transitions along the x axis. This means that we do not use all the previous values a stream took before a given x_i when computing the probability of being in some other area in the time slot following (with $x_{i+1} \equiv x_i + \Delta x$). This is done to minimize the computational complexity that would arise from doing so.

The representation mentioned above can still have a problem: the notion of "area" itself. If it is too wide or too narrow, the model fails to capture the relevant pattern of the event. If any of Δx or Δy are too broad, information about the event will be lost. On the other hand, if these values are too narrow, the computation's complexity of the probabilities will start to degrade. Even worse, it can make the whole representation too specific (resulting in overfitting).

In order to minimize this issue we propose to keep the following values associated to each slot, as shown in Figure 2:

Probability vector This is the function which makes possible representing the nature of the stream using probabilities. Each P_i maps to the probability of jumping to the y_i following along the x axis (the transition).

Histogram of values Each slot maintains a histogram of values, allowing the model to identify which values are more commonly found within that slot. In a sense this adds another dimension to the model.

Other statistical values Other statistical values may be kept for further improvements. For example, keeping the average and the standard deviation of the values within the slot. They are both cheap computationally wise and may be of significance when evaluating how well a given point fits within the slot.

B. Stream Generation

Apart from stream characterization and similarity estimation, our model can also be used to generate streams. Context-aware platforms, in fact any platform that deals with context information (IoT/M2M data included), benefits from a realistic stream generator. As these platforms become smarter it becomes imperative to validate and evaluate the platform in a controlled environment. In our specific case, initial work demanded the use of large datasets to carry on tests and to evaluate the capability of representation of our organization model. This lead to the development of a stream generator general enough to be used in a wide class of streams. We want to use it to easily build synthetic datasets from real ones we had, but which were not as big as needed.

Such generator would have to output plausible streams, and not just a stream which would for instance minimize

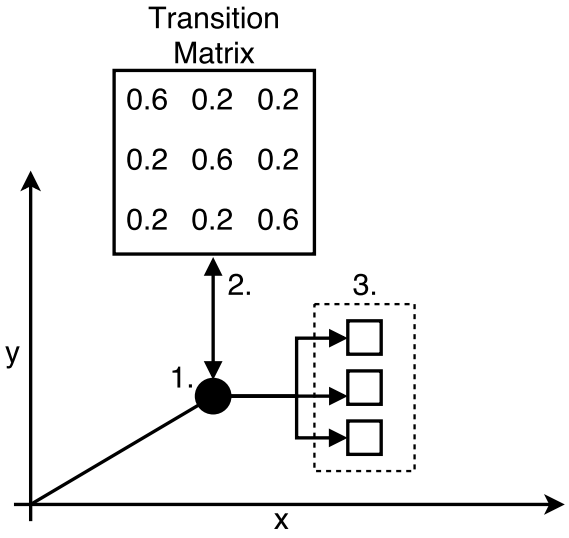


Fig. 3: Depiction of the generation process. 1) at each gap, generate a random value that represents the transition probability; 2) use the transition matrix to identify the next gap; 3) at the destination gap, generate a new value (based on the gap histogram).

the errors between itself and the set of streams given as examples. This constituted an opportunity to test our proposed representation. The internal structure of the generator is, thus, a matrix of slots, each with the values as described in Section III-A. This matrix is built for each type of pattern we want to learn, from a set of streams representative of the pattern (e.g. temperature or humidity). After having the matrix built, we can traverse it (along its x axis) to generate streams similar to the underlying pattern of the ones which were previously presented (as depicted in item 3).

Preliminary tests show the good capability of the generator to learn the most relevant motifs of the streams and be capable of generating realistic streams from the representation built. This is further discussed in Section IV.

IV. Performance evaluation

This section will present our current results, explaining some algorithms developed and making a few considerations about them. First, we will deal with our generator and its capability to generate reasonable streams. Afterwards, we will present our current results regarding period finding.

A. Generator

The generation process is carried by iterating over each column of the matrix representing the event being considered. The initial row the algorithm occupies is given as a parameter.

At each column, the algorithm generates a random value given the probabilities held in the histogram of the slot. Then, the algorithm computes the next row which will

TABLE I: MSE values computed for the streams generated.

	Real		
	Mean	Median	Stdev
Temperature	10.5	9.3	3.9
Humidity	51.4	37.3	27.9
Light	217360	175633	100361
	Generated		
	Mean	Median	Stdev
Temperature	10.0	9.3	3.0
Humidity	48.3	49.1	11.8
Light	221271	222265	39933

be occupied (for the iteration following), by generating a random value and comparing it with the transition probabilities from the Markov chain it holds.

1) Single Period: We worked with three kinds of natural phenomena: environment temperature, humidity and light intensity. Each set was composed of approximately one hundred streams.

During development, we worked with environment temperature only. With this in mind our goal was to extend our results by including different streams, and ensuring that our methods would still hold.

Since we have not yet finished a similarity metric for our stream characterization model, we will use MSE (mean-square error) and visual representations to evaluate the performance of our model. This evaluation was carried by k-cross validation, and is displayed at Table I. We obtained these values by selecting one real stream and comparing it with all the others, and doing the same for a generated stream (and picking with different streams). We would like to highlight that the differences between the real and generated results are not too far away.

Meanwhile, Figure 4 enables a more visual evaluation of our results, plotting real *vs* generated streams.

We would like to highlight that not only are the curves similar, but the standard deviation at each point is also approximate. This suggests that our model does not seem to be over fitting — the set of learning curves was composed of heterogeneous samples, which is indeed propagated to the generated streams.

The MSE values also validate that our generated curves are not too far off the real ones. Even regarding "Light", which scored a much bigger MSE than the other sets, our model agrees with the results from real streams.

2) Multiple Periods: Our model also supports the generation of multiple continuum periods. We modelled this behaviour by computing the probabilities of wrapping around the matrix representation (i.e. going from the last column to the first).

This way, with the same Markov simplification made throughout the document, we gave the model the knowledge to generate continuous stream (in a true infinite stream scenario). We have called this the "continuum" mode, which Figure 5 presents a plot of.

We find it relevant to say that the transitions between periods are smooth and that, without the colouring to tell them apart, the transition points would probably be unnoticeable. This property will become rather important when considering time shifts in a similarity metric.

B. Period Finding

In order to automate the usage of the generator (and later the similarity component), we have developed a module for automatic period detection. It works by first computing a periodogram of a stream, and selecting the k strongest frequencies from it — commonly named candidate frequencies. Then, for each of these candidates, an autocorrelation factor (ACF) with lag of the inverse of the frequency (the period) is computed. The period which gives the best ACF is selected.

Since the sample points of a stream may not be equally spaced, we drop portions of the stream that do not meet a minimum percentage of points given the period being considered. Furthermore, we linearly interpolate the stream in order to get equal spacing between samples (which is import for the autocorrelation, otherwise the point-wise operations are not matched), and evaluate the linear-interpolation every $2\Delta t$, where Δt is the mean time differences along each original sample. Initially we did not use interpolation, making the autocorrelation fail due to the mismatch between samples' spacing. With linear interpolation the problem was solved. Despite this, we are still evaluating other methods to improve period detection on data streams.

To validate the period detection method we used three sets of streams (akin to the generator method): environment temperature, humidity and light intensity.

We expected a periodicity of approximately 1 day (± 86400 s) for them, which is further suggested by visual inspection of the plots of the streams we used.

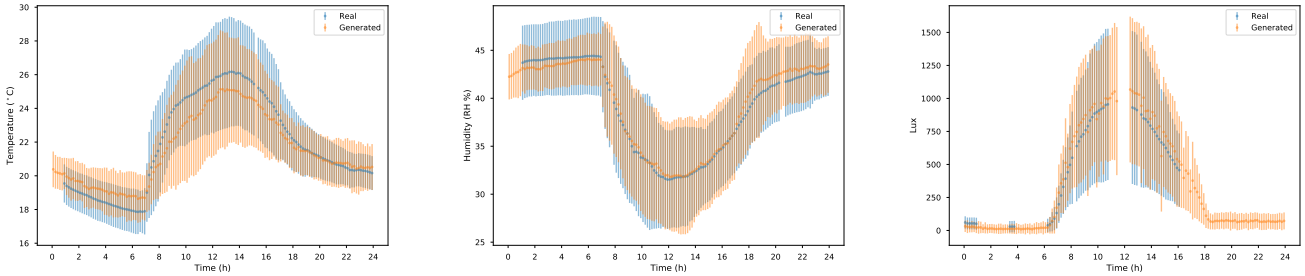
Figure 6 shows three histograms, each depicting the computed periods over approximately fifty streams of the respective phenomena.

Although including some outliers, we highlight that the maximums of each plot stand well above the rest of the values and are indeed close (within a 5% margin) to the expected 1 day period.

V. Discussion and Conclusions

The number of sensing devices is increasing at a steady step. Each one of them generates massive amounts of information. However, each device/manufactures shares context information with different structure, hindering interoperability in IoT/M2M scenarios.

We tackled this issue by developing an organization model agnostic to context representation. Our organization model uses tailored features to automatically organize data and improve its accuracy. By using our generative stream model as a tailored feature to describe stream patterns we believe that our organization model will be further



(a) Temperature (b) Humidity (c) Light

Fig. 4: The three kinds of generated streams: temperature, humidity and light. The vertical bars represent the standard deviation (at each point) of 20 different streams.

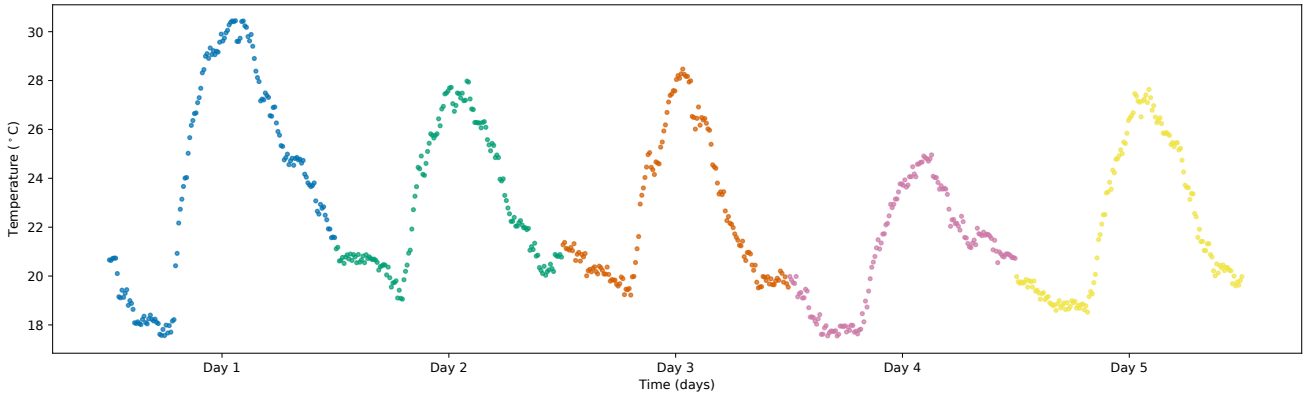
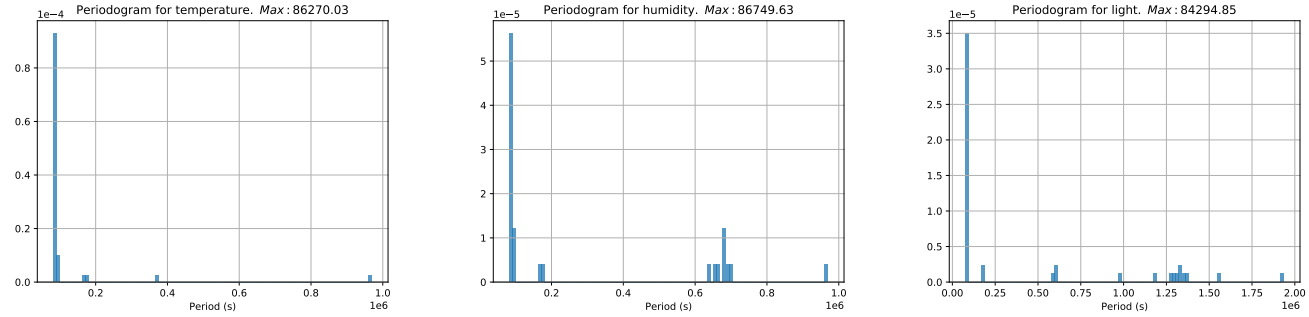


Fig. 5: 5 temperature streams generated as a continuum. The generator had been trained with around 100 streams prior to the generation.



(a) Temperature (b) Humidity (c) Light

Fig. 6: Periodograms for the three phenomena analysed.

improved. It is worthwhile to mention that there are several academic works based on stream prediction and mining [16], but the same cannot be said about stream similarity and stream characterization. Further work needs to be done to assert some ideas expressed in this paper, but our stream characterization model appears to be a viable option.

We are currently devising a similarity metric to estimate the similarity between a stream and our model. After, the proposed stream characteziarion model will be used in our context organization model as a tailored feature. This allow us to organize data based not only on semantic features [10], but also on stream patterns. It is our belief that

the integration of our fast labelling method with existing classification techniques will make organization across large stream-bases both possible, efficient and accurate. Our algorithm will serve as a strong filter, trimming the search space so that other techniques can proceed. For example, IoT/M2M platforms can use machine learning techniques over our context organization model to provide smart and proactive services, high level inference, amongst others.

There is room to further improve our stream characterization model. Specially to cope with the variability associated with IoT/M2M scenarios. Some questions which are yet to be answered include: Is scale (along the y axis) important? If yes, in which cases and how to work with it? How to cope with time and location differences across the different sensors? We will continue our research on these topics and hopefully answer these questions in future publications.

Meanwhile, the ability to generate streams resembling a given set of learning ones, can be useful in many situations. For instance, to generate large synthetic datasets where otherwise there is no specific generator available. Our general purpose generator has another big advantage, since it improves the repeatability and validity of IoT/M2M and context-aware platforms. Currently these platforms use advanced machine learning algorithms to improve and optimize several processes. Having the ability to test them for a long time in a controlled environment is extremely important.

Acknowledgement

This work was partially supported by European Regional Development Fund (ERDF) under grant agreement No. 7678 (Ref. POCI-01-0247-FEDER-007678) entitled "SGH - SMART GREEN HOME", and research grant SFRH/BD/94270/2013.

References

[1] F. Wortmann, K. Flüchter *et al.*, "Internet of things," *Business & Information Systems Engineering*, vol. 57, no. 3, pp. 221–224, 2015.

[2] K.-C. Chen and S.-Y. Lien, "Machine-to-machine communications: Technologies and challenges," *Ad Hoc Networks*, vol. 18, pp. 3–23, 2014.

[3] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context aware computing for the internet of things: A survey," *IEEE Communications Surveys Tutorials*, vol. 16, no. 1, pp. 414–454, 2014.

[4] M. Antunes, D. Gomes, and R. L. Aguiar, "Scalable semantic aware context storage," *Future Generation Computer Systems*, vol. 56, pp. 675–683, Mar. 2016.

[5] R. Fantacci, T. Pecorella, R. Viti, and C. Carlini, "Short paper: Overcoming iot fragmentation through standard gateway architecture," in *2014 IEEE World Forum on Internet of Things (WF-IoT)*, March 2014, pp. 181–182.

[6] J. Robert, S. Kubler, Y. L. Traon, and K. FrÃđmling, "O-mi/o-df standards as interoperability enablers for industrial internet: A performance analysis," in *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, October 2016, pp. 4908–4915.

[7] S. K. Datta, C. Bonnet, R. P. F. D. Costa, and J. HÃđrri, "Datatweet: An architecture enabling data-centric iot services," in *2016 IEEE Region 10 Symposium (TENSYMP)*, May 2016, pp. 343–348.

[8] G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith, and P. Steggles, "Towards a better understanding of context and context-awareness," in *Proc. of the 1st international symposium on Handheld and Ubiquitous Computing*, 1999, pp. 304–307.

[9] T. Winograd, "Architectures for context," *Hum.-Comput. Interact.*, vol. 16, no. 2, pp. 401–419, December 2001.

[10] M. Antunes, D. Gomes, and R. Aguiar, "Learning semantic features from web services," in *Future Internet of Things and Cloud (FiCloud), 2016 4rd International Conference on*. IEEE, 2016.

[11] A. K. Dey, "Understanding and using context," *Personal and Ubiquitous Computing*, vol. 5, no. 1, pp. 4–7, 2001.

[12] T. Mota, N. Baker, B. Moltchanov, R. Ioanna, and K. Frank, "Towards pervasive smart spaces: A tale of two projects," in *Future Network & Mobile Summit 2010. The Second International Workshop on Information Quality and Quality of Service for Pervasive Computing in Conjunction with IEEE PERCOM 2010*, 2010.

[13] P. Lopes and J. L. Oliveira, "Coeus: Semantic web in a box for biomedical applications," *Journal of Biomedical Semantics*, vol. 3, no. 1, p. 11, 2012.

[14] G. A. Miller, "Wordnet: A lexical database for english," *Commun. ACM*, vol. 38, no. 11, pp. 39–41, November 1995.

[15] M. Antunes, D. Gomes, and R. Aguiar, "Semantic features for context organization," in *Future Internet of Things and Cloud (FiCloud), 2015 3rd International Conference on*. IEEE, 2015, pp. 87–92.

[16] G. Krempl, I. Žliobaite, D. Brzeziński, E. Hüllermeier, M. Last, V. Lemaire, T. Noack, A. Shaker, S. Sievi, M. Spiliopoulou, and J. Stefanowski, "Open challenges for data stream mining research," *SIGKDD Explor. Newsl.*, vol. 16, no. 1, pp. 1–10, Sep. 2014.

[17] A. Marascu, S. A. Khan, and T. Palpanas, "Scalable similarity matching in streaming time series," in *Advances in Knowledge Discovery and Data Mining: 16th Pacific-Asia Conference, PAKDD 2012 Proceedings, Part II*. Springer Berlin Heidelberg, June 2012, pp. 218–230.

[18] A. Camerra, J. Shieh, T. Palpanas, T. Rakthanmanon, and E. Keogh, "Beyond one billion time series: indexing and mining very large time series collections with $$i$$ sax2+," *Knowledge and Information Systems*, vol. 39, no. 1, pp. 123–151, 2014.