Animal monitoring based on IoT technologies

Luís Nóbrega
DETI/IT
Universidade de Aveiro
Aveiro, Portugal
lnobrega@ua.pt

André Tavares
DETI/IT
Universidade de Aveiro
Aveiro, Portugal
andrefctavares@ua.pt

António Cardoso
DETI/IT
Universidade de Aveiro
Aveiro, Portugal
antoniocardoso@ua.pt

Pedro Gonçalves
ESTGA/IT
Universidade de Aveiro
Aveiro, Portugal
pasg@ua.pt

***Abstract*— The placement of grazing animals in vineyards requires additional support to the animal husbandry activities. Such support must include the monitoring and the conditioning of animal's location and behavior, specially their feeding posture. With such a system, it is possible to allow sheep to graze in cultivated areas (*e.g.* vineyards, orchards) without endangering them.**

This paper proposes an animal behavior monitoring platform, based on IoT technologies. It includes an IoT local network to gather data from animals and a cloud platform, with processing and storage capabilities, to autonomously shepherd ovine within vineyard areas. The cloud platform also incorporates machine learning features, allowing the extraction of relevant information from the data gathered by the IoT network. Thus, besides the platform description, some results are presented regarding the machine learning platform. Namely, this platform was evaluated for detecting and defining conditions respecting animal's posture, with preliminary promising results. Since several algorithms were tested, this paper includes a comparison of those algorithms.

Keywords—Animal Monitoring, Machine Learning, IoT, Posture Control, Big Data

I. NTRODUCTION

Viticulture is an agricultural practice with enormous implantation throughout the Mediterranean basin. One relevant example is the Portuguese one, where the total area of Portuguese vineyards was estimated to be around 191 thousand hectares, the fourth in Europe [1]. Moreover, the increasing exporting numbers of Mediterranean wine are transforming wine industry, namely due to its important role in countries' economy.

This activity is known to be tremendously labor-intensive, especially throughout the spring, a season of the year when it is necessary to continuously remove the wild species (weeds) so they do not compete for the sun and nutrients with the vines. The removal of weeds is an operation that started to be carried out manually and, when possible, using grazing sheep. As animals tend to feed on the lower branches of the vines and fruits, they become hazardous for cultures. Thus, as they couldn't be used throughout all the year, and sheep were taken exclusively to the meadows.

During the process of winemaking activity specialization that have been occurring during the last 50 years, animal weeding was abandoned. Instead, this process began to be performed mechanically in the areas between rows, and chemically in the areas under the rows, using herbicides. Such techniques are onerous, and while mechanical processes cause soil erosion and are a source of greenhouse gases, chemicals may contaminate fruits and watercourses in soils.

In the context of SheepIT project [2], solutions are being developed to enable ovine to weed vineyards (and other similar cultures), ensuring that do they not compromise the wine production, while contributing to the soil fertilization and to the sustainability of the livestock sector.

In order to reduce animal impact within agro companies specialized in dealing with plants, the project integrates an IoT sensor network, responsible for monitoring and conditioning animal's posture and location within vineyards. Additionally, it includes a cloud platform that aims at analyzing all the collected data and produce meaningful information to the end user. The processing power of today's cloud services, allows the incorporation of Data Mining (DM) and Machine Learning (ML) techniques, that can be used for extracting additional and relevant information for whom manages vineyards and/or shepherds.

In the scope of this paper it is explored the SheepIT cloud platform that receives animal sensor streamed data, performs data analysis (*e.g.* rule management), allows farm managers to access animal information and trigger alarms in real-time when certain situations are detected (*e.g.* panic attacks, attacks from predators, abnormal number of infractions, etc.). The paper also describes some preliminary results of an animal posture monitoring use case, for which several machine learning algorithms were tested.

The remain of the paper is organized as it follows. Section II describes related work on farm big data and section III describes the overall platform architecture, focusing in the cloud's one and its connection to the local IoT network. Section IV presents an example of the application of ML learning algorithms to predict the animal behavior. Section V concludes the paper.

II. RELATED WORK ANIMAL MONITORING

There is a huge diversity of published work on animal monitoring, with diverse objectives. Some examples are the study of the migratory behavior of wild animals [3], the behavior analysis of grazing animals [4][5][6], the study of grazing site profiles, the animal posture behavior [7] or the animal estrus [8][9]. Most of the work consists of the recording of animal

978-1-4673-9944-9/18/$31.00 ©2018 IEEE

978-1-5386-6930-3/18/$31.00 ©2018 IEEE

behavior sensing for later analysis, with few studies that perform streaming analysis.

Williams et al. [5] applied computer learning techniques over the GPS trace captured, during 4 months and from 40 cows, in order to classify their grazing behavior. Their experiments used WEKA data mining suit and comprised the use of four ML algorithms. They were able to classify three states, namely, grazing, resting and walking. The results accuracy is relatively low, especially for grazing and resting states, because of the use of a single sensor that records animal position. Despite confirming problems associated with high energy consumption of the system due to the use of GPS receivers, the paper identifies the suitability of the animal monitoring mechanism for other use cases related to activity monitoring and prediction of animal diseases.

Estrus detection [8] is one of the most documented animal monitoring use cases, possibly due to the economic gain obtained by an efficient management of inseminations. These studies typically report the use of accelerometer sensors coupled to the cows' necks or legs, monitoring animal activity. The gathered data is later analyzed and allows the detection of peaks of activity, which indicate estrus. In addition to the academic work there are some commercial products [9] that send monitoring information over a wireless network, allowing remote data analysis about feeding, rumination and levels of activity, easing cattle management.

In the work developed by Dutta et al. [6], monitoring collars are placed on cows, retrieving tri-axial accelerometry and magnetometry measurements. These are analyzed by a set of ML algorithms to determine thresholds that are later used to differentiate activities, using individual classifiers like "*Binary Tree*", "*Naive-Bayes*" or ensembles of those classifiers. Precision and sensitivity rates are over 90% for some of the classified activities, even for the simpler individual classifiers like the "*Binary Tree*".

Umstätter et al. [7] applied supervised behavioral classification to differentiate active (*e.g.* grazing, walking) and inactive (standing, recumbent) behaviors in 10 sheep. They carry GPS tracker collars equipped with pitch and roll tilt sensors, and different location conditions were experienced (*e.g.* hill, flat fields and shed). The datasets used in classification included minimum and maximum pitch and roll tilts over 30 second periods. Three classification methods were used: a linear discriminant analysis, which was successful in classifying active and inactive behaviors with over 92% accuracy; a classification tree, which resorted only to maximum pitch tilt and produced accuracies similar to those of the previous method, although it proved to be sensitive to the location (outside conditions reduced overall accuracy by 5% and the method required pre-calibration for each location); a manually developed decision tree, which relied on frequency analysis and customized datasets (measurement of stability of behavior, moving sum of differences from minimum to maximum tilt values). This final method reduced false classifications to 2.0% for the outdoor dataset, proving to be the most reliable.

III. AN ANIMAL MONITORING PLATFORM

The SheepIT project aims at offering an IoT-based solution for monitoring, controlling and managing herds of sheep weeding vineyards. To reach such goal, the system comprises several distinct blocks (and respective interactions), each of them being responsible for specific tasks. Among these tasks, the data gathering, data aggregation, data processing and data representation may be highlighted.

Fig. 1 depicts the overall architecture of the implemented platform. In the left part, the devices that compound the architecture of the IoT network implemented to run local tasks are illustrated. Such local infrastructure is in turn interconnected through a wide-band connection (*e.g* 3G, 4G, LTE) to a cloud platform, represented in the right part of Fig. 1. These two big blocks are described in the following sections.

A. IoT network

Collars, are the main data gathering interface, collecting data from sensors, being as well responsible for the supervision of the animals' posture, behavior and location. As these devices own processing abilities and because it is not suitable to wait for a decision to be handled and transmitted by a central node with more processing power, due to the delay associated, the posture control algorithm runs locally, analyzing sensors' data and applying corrective stimuli (*e.g.* electrostatic and auditory cues). Thereafter, the relevant data for the user is transmitted to an infrastructural network composed by fixed beacons. These devices are installed accordingly to the intended grazing areas and besides being responsible for collecting collar's data, they implement a periodical and synchronized beaconing signal emission all over the network that allow collars to evaluate their location through the use of RSSI-based localization techniques, and the network to trace back animal location. To maintain the power consumption as low as possible, the system follows an Time Division Multiple Access mechanism, being the details of the solution proposed detailed in [10].

The Gateway works as an aggregator element, interconnecting the local network to the Internet. It implements a beacon (with greater processing power) that communicates with the remaining beacons, and connects the local network to the Internet through a wide band connection. Moreover, the Gateway acts as a local network manager, coordinating local nodes, aggregating sensory information and implementing a local alarm generator for critical situations (*e.g.* fence violation, panic detection).

To allow a smooth integration between the non-IP network and the IP-based Internet, the Gateway also incorporates a module capable of mapping the gathered information into JavaScript Object Notation (JSON) data structures that can easily be parsed by upper layer applications.

Such infrastructure may be replicated in different properties from different owners, being all supported by a cloud platform described thereupon.

B. Cloud Platform

The Cloud platform, as illustrated in Figure 1, is composed by five different interconnected modules, responsible for the aggregation, analyze and processing of stream data. The

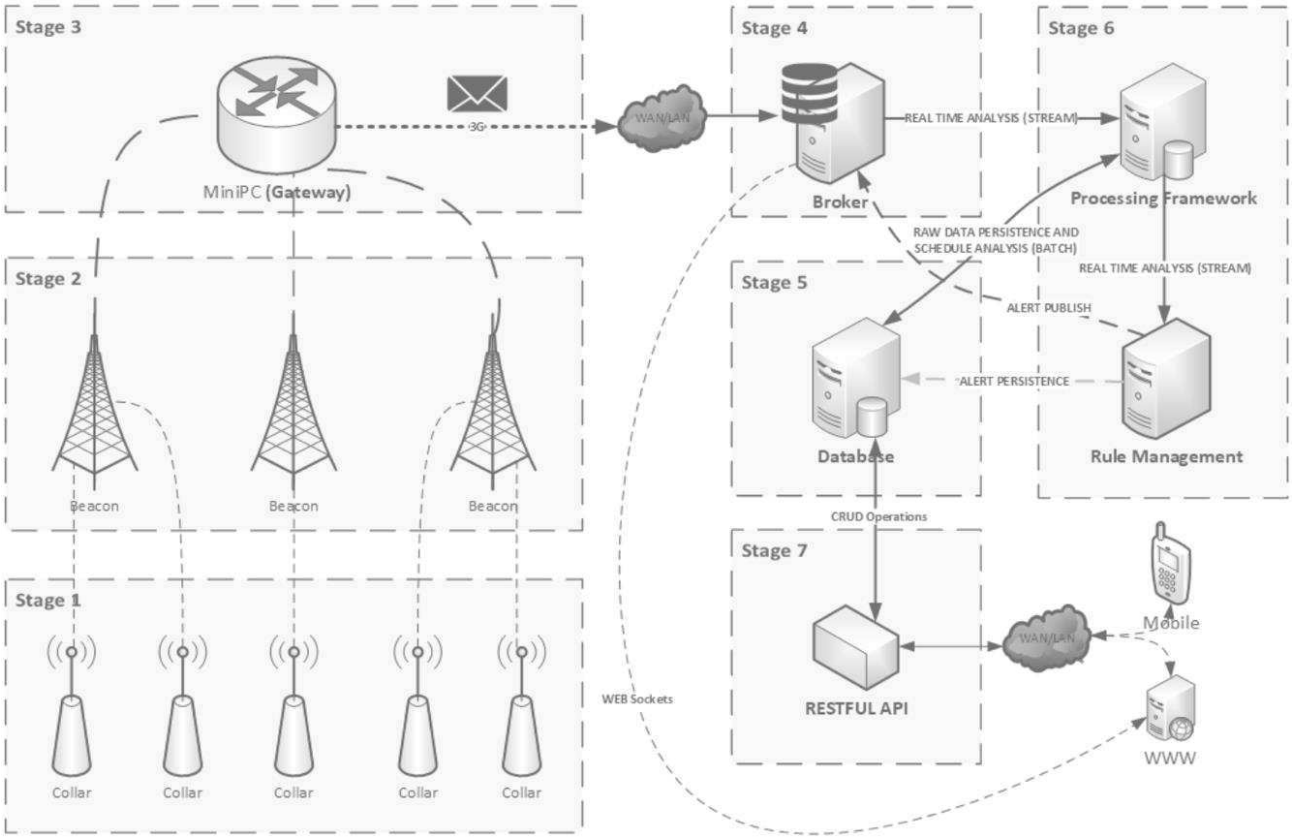

Fig. 1: Overall architecture

Message Oriented Middleware (RabbitMQ) [11] is one of the first stages, allowing message routing through producers and consumers. It receives JSON messages from the Gateway and make them available to the data processing engine. The broker supports several messaging protocols such as AMQP [12] or MQTT [13], both based on asynchronous publish/subscribe architectures. Henceforth, RabbitMQ works as an intermediary between the Gateway and the remaining platform, managing all the received messages prompted by the Gateway in a First-In-First-Out (FIFO) queue. Also, the RabbitMQ allows security mechanisms to be employed, such as SSL/TLS certificates.

At this moment, Apache Spark [14] is the only subscriber of a single queue created within the RabbitMQ. It is the main processing framework responsible for orchestrating all the operations of the platform. Among them we can highlight the JSON data translation, alarms generation, data processing including DM and ML techniques, and data persistence into the database (DB). It combines two main processes, a stream process that handles real-time traffic and a batch process that handles non-periodic traffic, especially as a result of processing tasks performed within the platform. Both processes are able to persist into the DB, feeding it regularly with new data. Complementarily to this processing framework, a rule management module (Drools) [15] was added, allowing the definition of complex event processes (CEP) and event stream processes (ESP). Briefly, they allow the generation of predictions, detection of patterns or other relevant relations that may trigger actions as alarms, such as real time notifications to the end user.

The DB represents another important module of the platform. The data gathered directly by collar's sensors as well as data retrieved by them, static information added by the user or other information about the system operation, need to be stored conveniently in order to be easily accessed by upper layer applications. Having the SheepIT network several entities, a relational database model looks more promising instead of a non-relational model. Moreover, besides all the advantages of a NoSQL model, the payload of the messages expected does not justify the use of this model on this architecture. Among the several available solutions, PostgreSQL has been chosen since it is suitable for environments with a large amount of data besides the security and integrity mechanisms that it presents.

Finally, the platform includes a REST API framework to allow the WEB development. With this, it is possible to interact directly with the user but also with other relevant platforms. For instance, information about animals could be directly integrated within legal databases for animal registry.

IV. ANIMAL BEHAVIOR MONITORING

Within the presented platform, collars are the main source of information. They are tasked of collecting information regarding the sheep's posture, location and infractions. The gathered sensor data can be converted into useful information to the

farmer, such as hours of activity, travel times, preferred pasture areas and timings, anomalous situations (*e.g.* panic, illness), number of fence and posture infractions, among other.

However, much of this information cannot be retrieved directly from sensors, being necessary more computational demanding mechanisms. One interesting possibility is the use of DM and ML techniques, which popularity and power have been increasing. Considering the ML module incorporated within the platform, we focused on a single but crucial use case, namely the detection of sheep' posture.

A. *ML use case – detecting sheep's posture infractions*

Detecting if a sheep is feeding on the vines or on the weeds is not straightforward, being necessary to evaluate more than one sensor to avoid bad decisions. Thus, ML was chosen to help on the process, namely resorting on supervised algorithms, which means that the learning algorithm learns from a training set and then applies the learning model to a test set to be evaluated. In this particular application, the data evaluated is gathered from a 3-axis accelerometer and an ultrasound transceiver incorporated on collars, which yield measurements of neck pitch and distance to the ground. Sheep were released onto a plain field and their activity recorded on video for about 3 hours. At the same time, collars continuously retrieved time stamped raw sensor data and sent it into the network in order to be manually classified.

The recorded videos were analyzed and each measurement entry was classified into one of the following categories: resting (*i.e.* the animal is standing, with its head facing forward – on the left of Fig. 2); the animal is eating off the ground, with its head lowered (on the middle of Fig. 2); standing and reaching up for food (on the right of Fig. 2); walking; and running. Only the cases previously defined as "*standing and reaching up for food*" were considered as infractions. All the remaining cases were re--classified as "*not infraction*". Hence, we faced a binary classification problem. Although not used in the scope of this paper, the wider classification made initially will allow more advanced activity classifications re-using the same dataset.

The ML techniques were implemented using R. Some pre-processing was performed, specially to remove redundant and duplicated data (through the Sequence Number and Time Stamp of the entry). The resulting dataset of 20555 entries was then shuffled and split into 2 subsets in a ratio of 75%-25%: a *Training set* (15416 entries) and *Test set* (5139 entries). The former is used by algorithms to define learning models, *i.e.*, algorithms learn from a data set correctly classified, while the latter is used to test the learned models.

Fig. 2: resting (left), grazing (middle) and standing recaching for food examples

B. *Machine Learning Algorithms comparison*

Taking advantage of the ML module integrated within the SheepIT's computational platform, different ML algorithms were evaluated to assess with which accuracy sheep's posture infractions could be detected. Different algorithms were tested, particularly the most popular in classification problems: Random Forest, Decision Trees (DT) using C50 and rpart packages, XGBoost, K-Nearest Neighbors (KNN), Support Vector Machine (SVM) and Naïve Bayes. The results were evaluated in terms of the following metrics, often used in classification problems; *i*) Accuracy (ACC) that represent the correctness of the prediction within the entire population; *ii*) True Positive Rate (TPR) or sensitivity, that identifies how often the model predicts true when it is actually true; *iii*) True Negative Rate (TNR) or Specificity, that identifies how often it is false when it is actually false; *iv*) Precision or Positive Predictive Value (PPV) that identifies, when it predicts true, how often it is correct; and *v*) The Receiver Operating characteristic Curve (ROC) and the Area Under de Curve (AUC).

Table 1 - MACHINE LEARNING ALGORITHMS COMPARISON

Algorithm	**Metric**				
	ACC	***TPR***	***TNR***	***PPV***	***AUC***
Random Forest	0.9696	0.8267	0.9861	0.8728	0.987
DT (C50)	0.9693	0.8475	0.9833	0.8539	0.986
XGBoost	0.9685	0.82674	0.9848	0.8625	0.988
KNN	0.9622	0.7702	0.9844	0.85.03	0.977
SVM	0.9642	0.7590	0.9879	0.8778	0.972
DT (rpart)	0.9591	0.8211	0.9750	0.8728	0.970
Naïve Bayes	0.9527	0.8795	0.9612	0.7230	0.979

All these metrics are derived from the Confusion Matrix that can be built when testing a ML model in a dataset of test. Although, for simplicity, we omit the representation of all the Matrices, we summarize all the metrics evaluated on Table 1. It can be observed that the results do not differ much among all the

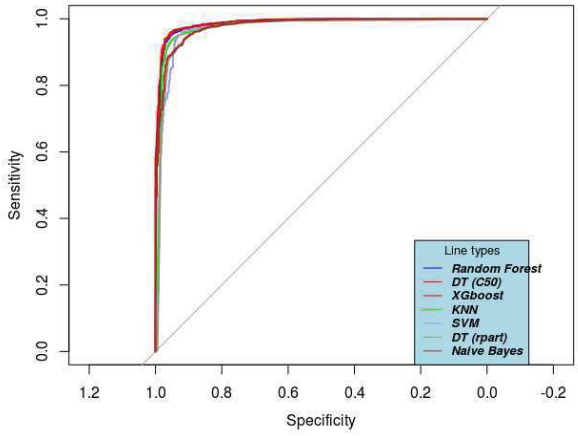

Fig. 3: ROC Curves for the ML algorithms evaluated

algorithms, although Random Forest, DT (using package C50) and XGBoost present the best results in terms of accuracy and AUC. This similarity in terms of results can also be verified athwart Fig. 3, on which we can witness that the curves are mostly overlapped.

Slightly different is the computational time necessary to generate the model as well as the easiness to interpret the relation between the attributes and the classifier label (in this case, infraction or not). In that scope, DT's exhibit the most appropriate features, since besides generating fast models they allow the representation of the model in terms of a set of if's and else's, forming a tree easy to interpret. Each terminal node (leaf) in a branch represents a classification label. Thus, each path since the root node to a terminal leaf, describes a classification rule that can be easily transposed to a set of if's and else's. This is quite relevant to the SheepIT project since it represents a very useful mechanism to obtain the proper conditions to implement a posture control mechanism in the sheep's collar, based on a simple microcontroller.

V. Conclusions

Weed control is a noteworthy problem in vineyards. It demands from winemakers significant economic and labor efforts. Moreover, the solutions currently used, either mechanical or chemical, are intended to be avoided by producers in order to increase the quality of their products. Thus, sheep, by their propensity to feed from weeds, are seen as an alternative and environment friendly solution. However, to protect cultures, the SheepIT project proposes a system to condition the posture and location of sheep while they graze in vineyards.

Besides the local operation, the system comprises a computational platform running on the cloud that receives the data gathered locally and process it in order to retrieve additional information from them. One of the mechanisms that may be used is Machine Learning.

This paper presents the overall system architecture, from collars, the mobile nodes carried by sheep, up to the cloud platform with different tasks as data analyses, data processing or data storage.

The potential of this platform is proven by an important use case of the SheepIT project, namely the detection of moments where sheep present a posture that could put in risk the vines and grapes. A data set composed of collar's sensor data was built and stored taking advantage of the existing platform and each entry was manually classified. Different ML algorithms were then evaluated in order to assess the platform power. All algorithms showed similar accuracy but the results obtained using DT are especially relevant since, their easier interpretation, helped to the definition of posture control algorithm to be implemented on collars.

Future work shall include evaluating the platform behavior in terms of scalability and performance with larger amounts of data as well as evaluating other ML use cases, as detection of illness, panic attacks, patterns of movement, food preferences, just to cite a few.

Acknowledgment

This work is supported by the European Regional Development Fund (FEDER), through the Competitiveness and Internationalization Operational Programme (COMPETE 2020) of the Portugal 2020 framework [Project SheepIT with Nr. 017640 (POCI-01-0247-FEDER-017640)].

References

[1] The International Organisation of Vine and Wine, "State of the Vitiviniculture World Market," 2017.

[2] L. Nóbrega, P. Pedreiras, and P. Gonçalves, "SheepIT - An electronic shepherd for the vineyards," in *8th International Conference on Information and Communication Technologies in Agriculture, Food & Environment*, 2017.

[3] J. Hunter *et al.*, "OzTrack -- E-Infrastructure to Support the Management, Analysis and Sharing of Animal Tracking Data," *2013 IEEE 9th Int. Conf. e-Science*, pp. 140–147, 2013.

[4] L. A. González, G. J. Bishop-Hurley, R. N. Handcock, and C. Crossman, "Behavioral classification of data from collars containing motion sensors in grazing cattle," *Comput. Electron. Agric.*, no. 110, pp. 91–102, 2015.

[5] M. L. Williams, N. Mac Parthaláin, P. Brewer, W. P. J. James, and M. T. Rose, "A novel behavioral model of the pasture-based dairy cow from GPS data using data mining and machine learning techniques," *J. Dairy Sci.*, vol. Volume 99, no. Issue 3, pp. 2063–2075, 2016.

[6] R. Dutta *et al.*, "Dynamic cattle behavioural classification using supervised ensemble classifiers," *Comput. Electron. Agric.*, no. 111, pp. 18–28, 2014.

[7] C. Umstätter, A. Waterhouse, and J. P. Holland, "An automated sensor-based method of simple behavioural classification of sheep in extensive systems," *Comput. Electron. Agric.*, vol. 64, no. 1, pp. 19–26, Nov. 2008.

[8] M. S. Shahriar *et al.*, "Detecting heat events in dairy cows using accelerometers and unsupervised learning," *Comput. Electron. Agric.*, no. 128, pp. 20–26, 2016.

[9] "MooMonitor+ - Health & Fertility Monitoring," 2018. .

[10] L. Nóbrega, P. Gonçalves, P. Pedreiras, and S. Silva, "Energy efficient design of a pasture sensor network," in *The 5th International Conference on Future Internet of Things and Cloud- FiCloud 2017*, 2017.

[11] M. Rostanski, K. Grochla, and A. Seman, "Evaluation of highly available and fault-tolerant middleware clustered architectures using RabbitMQ," in *Computer Science and Information Systems (FedCSIS), 2014 Federated Conference on*, 2014, pp. 879–884.

[12] M. Ritchie *et al.*, "AMQP Advanced Message Queuing Protocol Protocol Specification License."

[13] OASIS, "MQTT Version 3.1.1," *OASIS Stand.*, no. October, p. 81, 2014.

[14] M. Zaharia *et al.*, "Apache spark: a unified engine for big data processing," *Commun. ACM*, vol. 59, no. 11, pp. 56–65, 2016.

[15] M. Proctor, "Drools: a rule engine for complex event processing," in *Proceedings of the 4th international conference on Applications of Graph Transformations with Industrial Relevance*, 2011, p. 2.