Cognitive Determination of Policies for Data Management in IoT Systems

Aly Megahed*, Samir Tata*, Ahmed Nazeem**

IBM Research - Almaden,
650 Harry Rd, San Jose, CA 95120, USA
*{aly.megahed,stata}@us.ibm.com
** ahmed.nazeem@ibm.com

Abstract. Internet of Things (IoT) has emerged as a very hot area in the past few years. Managing data in IoT systems is still a challenging research topic, particularly when one aims at determining the correct set of decisions to take given some trigger events in an IoT system. The state of the art in determining such actions and corresponding policies is ad-hoc, based on significant human intervention. In this work, we propose a cognitive automated approach for policy and action determination in IoT systems that uses historical data to learn the best set of actions to take and involves an mathematical optimization module that chooses the optimal set of actions to pursue given the limited resource capacity in the system. Our system requires minimal human intervention and thus could be very beneficial in today's IoT frameworks.

1 Introduction

Internet of Things (IoT) refers to networks of objects, machines, vehicles, and other physical systems with embedded sensing, computing, and communication capabilities [18]. These devices sense and then share real-time information about the physical world. IoT is growing at a highly increasing rate. For example, according to a study by Gartner ([33]), it is believed that, by 2020, there would be 25 Billion connected things. Such connected things produce massive amounts of monitored data produced by sensors and devices. That is why data management has been evolved as one of the recent challenges for IoT systems. Data management policies are one of the main techniques to ensure collecting, managing and disseminating data is performed in a systematic, planned and managed way [38]. Examples of such policies include:

- Applying face detection algorithms to the recorded videos in places where a robbery happened,
- Applying algorithms to focus on car plate recognition when an accident happens in a smart city,
- Calling the fire department when a fire is detected with a detailed request of the number of trucks, appropriate number of fire fighters and required tools.

Depending on the context of the IoT system, data management policies are added to/removed from the IoT system. A naive approach would rely on human intervention

expertise to add/remove policies. Doing so is inefficient and error-prone in many applications. Thus, there is a need for a cognitive system that determines which policies to adopt whenever changes are triggered in the system with as minimal human input as possible. This is what we develop in this paper. That is, we present a cognitive method that automatically determines the appropriate policies in an IoT system, by analytically manage the input data and use historical policies and data to learn appropriate actions.

This paper is organized as follows. First, we present the state of the art in Section 2. Then, in Section 3, we present our methodology for action recommendation for policy-based data management in IoT systems. We lastly present our conclusions and directions for future work in Section 4.

2 State of the Art

Data management in IoT systems typically begins with monitoring the data. There has been multiple works in the literature on data monitoring [2]. Four main monitoring approaches exist, namely sampling and filtering-based monitoring (see, for example, [40]), probing-based monitoring (see, for example, [20], [21]), diagnosis-based monitoring (see, for example, [26], [28]), and performance-based monitoring (see, for example, [10, 14]).

In [40], the authors presented an adaptive monitoring framework for IoT devices. The system adapts the frequency of the monitoring as well as the amount of data going through the system. Their system increases the monitoring frequency when the value of monitored metrics gets to a certain pre-specified threshold and decreases it when a quality of service violation is unlikely to happen. The authors developed a monitoring approach that uses probing and is passive in nature.

In [26], the authors adopted a diagnosis-based monitoring approach based, where system faults are identified and then the metric monitoring is adapted accordingly. In [14], the authors developed a method for integration, validation, and description of metrics of software project performance. G. Katsaros et al. [22] proposed an architectural approach that combines virtualization and physical levels for monitoring data collection, where many open source solutions were combined (e.g. Lattice [11], Ganglia [24], Nagios [15]) so as to get one holistic application covering different layers. They used collectors for data extraction from different virtual and physical layers, and then data externalization to the upper layer using an external data collector. They also proposed a monitoring manager that acts as an orchestrator of the whole monitoring system. That manager controls and provides the required interfaces to add or consume monitoring information.

In [7], the authors proposed a tool, for distributed systems, for monitoring resources that enhances high-performance computing. The tool extracts both the application and resource state, and assigns new resources and/or shut down unused ones based on these states. This happens during the runtime of the application or in future usages. Data collectors are used to collect the data from resources and then the data is stored in a distributed database. Later, a statistical analysis is performed to take decisions on the resource assignment in the application, i.e., whether to keep it as the latest assignment or manually reconfigure it.

As for more specific data management literature on IoT, there has been some recent literature. For example, in [41], the authors formulate some deign principles for IoT data management. They developed optimization algorithms aimed at producing coherent IoT ecosystems, by means of publish/subscribe middleware and linked data that span over cloud infrastructures and mobile networks. In [13], the authors proposed a distributed data service for data collection and processing in IoT systems. They mentioned that their main goal is to enable multiple and different IoT middleware systems to share common data services that is coming from a loosely-coupled provider. To that end, they specified the corresponding techniques for data collection, filteration, storage, and aggregation to allow for efficient real-time data querying.

Padiya et al. [30] addressed the challenge of handling massive sensor data in an interactive fashion via Resource Description Frameworks (RDF). They compared multiple RDF storage mechanisms such as triple store, vertically partitioned table, horizontally partitioned table, column store, among others. They also represented a set of metrics that were designed to take decision for choosing the appropriate RDF data storage technique a priori for IoT systems. In [1], the authors present a survey for data management solutions proposed for IoT or subsystems of IoT. They discussed the different design primitives that they claim to be most important to address in an IoT management solution and showed how they were approached by the proposed solutions. Additionally, they proposed a data management framework for IoT that incorporates the aforementioned design elements and acts as a seed for a comprehensive IoT data management solution.

Gubbi et al. [19] present a cloud centric vision for the world-wide implementation of IoT. They proposed a cloud implementation based on the interaction of private and public couds and concluded that there is a need for expanding on the convergence of wireless sensor networks. In a previous work [39], we proposed an adaptive monitoring approach in IoT systems, where a metric value-based change or an environmental one triggers the need to choose which metrics to monitor and at what frequency. While an approach for doing this metric monitoring management was presented, no mention of action policies was discussed. A slightly similar work with an application in healthcare can be found in [4].

As one can see, all the previous works dealing with data management in IoT systems do not consider the issue of the determination of policies governing the IoT system. To the best of our knowledge, the only related literature is that concerned with security and control policies, rather than action policies, which is the focus of our work here. For examples on the latter policy focus, we refer the reader to the papers in [34] and [37]. It seems that action policies has been done in an ad-hoc and/or manual manner in the current state of the art. Thus, there is a need to fill this gap in the literature, as well as practical systems, via providing a methodology that handles automated cognitive action recommendation for different events that are monitored within an IoT system. This is what we provide next in this work.

3 Methodology

Figure 1 illustrates the overview of our methodology, where the first step (defined in details in Section 3.1) constitutes a recommendation model that gets trained on historical pre-defined/hypothetical events-conditions-actionss data. Such a model can then be used for new events-conditions to recommend the actions for such events-conditions. Only in case the recommended actions are not incorporated with a high confidence, they might need some expert validation. The expert validation step is detailed in Section 3.2 along with the update of the historical data as per the expert input. We then prioritize the actions recommended by the system in the step detailed in Section 3.3 using a model that learns such prioritization from historical data. Lastly, since the system is typically resource-constrained, an optimization mode that chooses the optimal final set of feasible actions to execute, trying to maximize the system utility by giving more resources for the actions with higher priority. We detail this optimization model in Section 3.4.

3.1 Recommendation of Action Given An Event-Condition

We assume that we are given a set of event-condition-action triplets. That is, for each given historical or hypothetical event, we are given a condition and a corresponding action. An example of elements of that set in our IoT context would be a robbery (event) that happens at a bank (condition) with the corresponding actions being calling the police and focusing the cameras of the smart cities on plates of cars surrounding the bank. Now, what we want to achieve in this step of our methodology is that whenever a new event-condition happen, the system is able to automatically recommend the best action to take.

In order to achieve this, we develop a classification model that classifies the events-conditions to the corresponding actions. We refer the reader to [6, 17, 27] for extensive analysis of machine learning classification. We note that the events-conditions are typically text data and thus, in order to do such classification, we need to first perform some text mining in order to retrieve the features of these events-conditions. This can be done as follows: We convert each event-condition to a bag of words. This text is transformed in order to get label indexers. The label indexers fit on the whole dataset. Additionally, we develop feature indexers that automatically identify categorical features and have them indexed. Then, we are ready to build the classification model that gets trained on such historical data to predict the recommended action. Several machine learning techniques can be used for each of the two steps. For the text mining, we refer the reader to recent review in [36]. An example of a traditional, but yet very popular, algorithm for the kind of text mining that could fit our purpose here is Term Frequency-Inverse Document Frequency (TF-IDF) [23]. The idea behind TF-IDF is that a numerical statistic is calculated to reflect how important a word is in a corpus. Its value increases proportionally to the number of occurrences of a word in the document, but is often offsetted by the frequency of the word in the corpus. The purpose of this is to adjust for the fact that some words appear very frequently in general. We refer the reader to the reference in [32] for details on how TF-IDF works.

Examples of machine learning classifiers along with their references are Decision Trees [31], Random Forests [8], K-Nearest Neighbor Classification [12], Support Vector

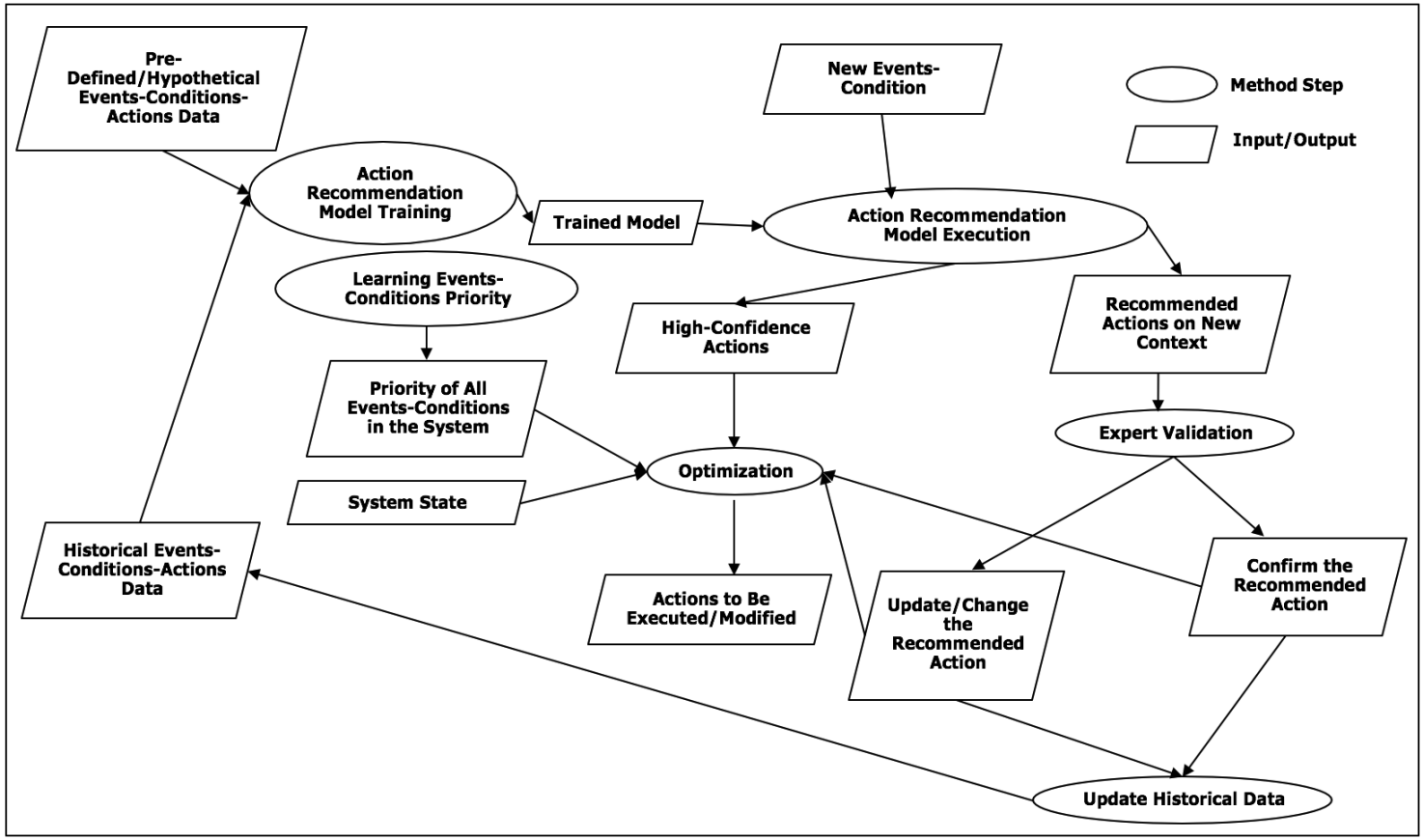

Fig. 1: Overview of Our Overall Methodology

Machines [35]), Logistic Regression [25], Naive Bayes [29], and Gradient Boosting Machine (GBM) classifier [9]. We also refer the user to recent similar applications of data mining and classification in [3] and [5]. Figure 2 illustrates that whole methodology of action recommendation for a given event-condition.

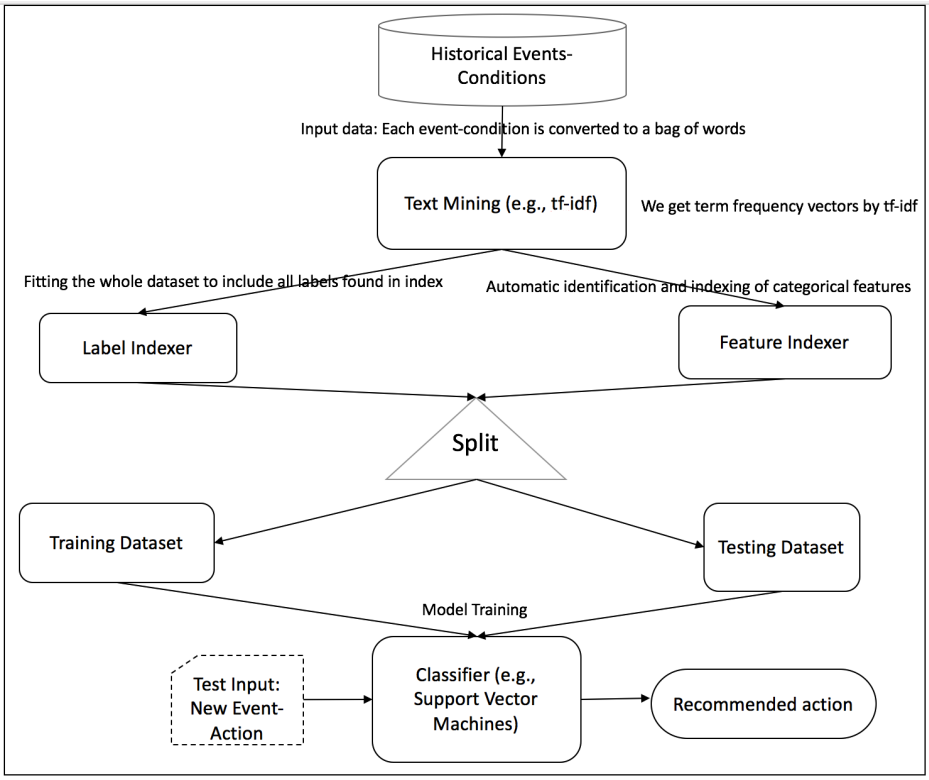

Fig. 2: Illustration of Our Action Recommendation Methodology

3.2 Validating the Learned Actions and Updating the Historical Data

Machine learning classifiers, like the ones we proposed to use in the previous subsection, typically output a score of confidence in the classification assignment. Assuming that the user of our system determines a threshold for the minimum accepted confidence (e.g., 50%), we program the following in our system: If the threshold of assigning an action to an event-condition is equal or above that threshold, then we just apply it. If it is lower, then we let an expert validate that recommended actions of low confidence. This would typically happen when the model encounters new classes of events and conditions pairs.

In the latter case, after the expert review, the expert would either confirm the recommended action or they would update/change it. Note that such expert validation is minimal compared to ad-hoc prior systems where the expert defines all the policies manually. Lastly, the expert recommend actions are used to update the historical events/conditions/actions data to improve the learning model in the future recommendations.

3.3 Learning the Priority of Events-Conditions

The objective of this step is to prioritize the actions recommended by the recommender discussed in Section 3.1. We need to do such prioritization because there are typically limited resources in the systems and thus, we might need to do a subset of the recommended actions. Therefore, we want to prioritize these actions so that the ones that the system finally chooses to perform have the highest possible collective priority.

We assume that we are given sets of previous actions and their importance. For example, an action of calling the fire department when a fire occurs is obviously more important than the action of turning off the air conditioning in a building given a certain temperature. That importance could either be a score/weight of importance of each action or a binning classification (e.g., classifying each action in one of five bins, where the first bin corresponds to most important actions, the second one is less important,. . ., etc).

Now, the way our prioritization works is similar to our recommendation. That is, we perform some text mining on the actions and the classify them into the importance classes. The classification could be a multi-label classification in case the historical data are labeled into bins, or if could be a binary classification in case the prioritization is in the form of a weight for each action. In the latter case, the score that the model gives for each action corresponds to the predicted weight of such action.

After we prioritized the actions, we note that we cannot just simply rank them in order of the predicted priority and keep fulfilling the capacity of the resources that we have with the ones with the highest weight until that capacity is fulfilled. We cannot do that because, first the resources might be multi-dimensional (e.g., CPU, RAM, network bandwidth, and storage of the devices used for the data flow in our IoT system), and secondly, even if it were one-dimensional (i.e., just one resource type), using such greedy algorithm might not make an optimal use of the available resource capacity. Thus, we formulate a mathematical optimization model in the next section in order to optimally choose the final set of actions that can be feasibly performed.

3.4 An Optimization Module for the Final Action Recommendations

Let the set of resources be R and the set of recommended actions be A. We denote the utilization of each action $a \in A$ from each resource $r \in R$ by u_{ar}, and the capacity of resource $r \in R$ by c_r. We also denote the priority/weight of action $a \in A$ by w_a. Lastly, our only set of variables here are the ones related with whether we will choose action $a \in A$ to be executed or not. We let X_a be such variables, where X_a is 1 if action $a \in A$ is chosen, and 0 otherwise. Now, we formulate our mathematical optimization model as follows:

$$\max \quad \sum_{a \in A} w_a . X_a \tag{1}$$

$$\text{s.t.} \quad \sum_{a \in A} u_{ar} . X_r \leq c_r, \ \ \forall r \in R \tag{2}$$

$$X_r \in \{0, 1\} \tag{3}$$

Where objective function 1 maximizes the sum of the weights of the actions that are to be chosen. Note that such weight is the score of the action, in case the output of the priority learning step is the weight, and it is the reciprocal of the priority/bin case the given priority is the corresponding bin. Constraint 2 ensures that the total utilization of all chosen actions from each resource is less than or equal to the total capacity of that resource. Lastly, constraint 3 ensures that each of our variables is either zero or one.

Model (1-3) is an integer programming model, which might not be easy to solve. However, we note that its structure is a multidimensional knapsack problem, which is well studied in the operations research literature. We refer the reader to the review article in [16] for efficient ways of solving such model.

4 Conclusions and Directions for Future Work

In this paper, we proposed a cognitive system for action recommendation action recommendation for policy-based data management in IoT systems. That is, whenever an event-condition happens in an IoT system, we showed how our system can be used to recommend the optimal set of actions to execute, given the resource constraints of the system, and with minimum human intervention.

There are several directions for future work. First, an implementation of our system with a real-world IoT use case would show the effectiveness of our approach. Second, investigation of different machine learning algorithms in the recommendation and priority determination steps would be helpful in determining which ones are most appropriate for our proposed system. Lastly, one direction for future research to our work is incorporating uncertainty in the observed events and conditions, and designing a system that would recommend actions before such certainty is realized.

References

1. Abu-Elkheir, M., Hayajneh, M., Ali, N.A.: Data management for the internet of things: Design primitives and solution. Sensors 13(11), 15582–15612 (2013)
2. Aceto, G., Botta, A., de Donato, W., Pescapé, A.: Cloud monitoring: A survey. Computer Networks 57(9), 2093 – 2115 (2013)
3. Asthana, S., Megahed, A., Becker, V., Nakamura, T., Gajananan, K.: A cognitive prioritization for reports generated in resource constrained applications. In: Services Computing (SCC), 2017 IEEE International Conference on. pp. 418–425. IEEE (2017)
4. Asthana, S., Megahed, A., Strong, R.: A recommendation system for proactive health monitoring using IoT and wearable technologies. In: Artificial Intelligence and Mobile Services (AIMS), 2017 IEEE International Conference on. pp. 14–21. IEEE (2017)

5. Asthana, S., Strong, R., Megahed, A.: Healthadvisor: Recommendation system for wearable technologies enabling proactive health monitoring. arXiv preprint arXiv:1612.00800 (2016)
6. Bishop, C.: Pattern recognition and machine learning. Springer, New York (2007)
7. Brandt, J., Gentile, A., Mayo, J., Pebay, P., Roe, D., Thompson, D., Wong, M.: Resource monitoring and management with OVIS to enable HPC in cloud computing environments. In: IEEE International Symposium on Parallel Distributed Processing (May 2009)
8. Breiman, L.: Random forests. Machine learning 45(1), 5–32 (2001)
9. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22^{nd} ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 785–794. ACM (2016)
10. Cheng, Y., Chen, W., Wang, Z., Yu, X.: Performance-monitoring-based traffic-aware virtual machine deployment on numa systems. IEEE Systems Journal PP(99) (2015)
11. Clayman, S., Galis, A., Mamatas, L.: Monitoring virtual networks with Lattice. In: Network Operations and Management Symposium Workshops (April 2010)
12. Cover, T., Hart, P.: Nearest neighbor pattern classification. IEEE transactions on information theory 13(1), 21–27 (1967)
13. Cruz Huacarpuma, R., de Sousa Junior, R.T., de Holanda, M.T., de Oliveira Albuquerque, R., García Villalba, L.J., Kim, T.H.: Distributed data service for data management in internet of things middleware. Sensors 17(5), 977 (2017)
14. Doraisamy, M., bin Ibrahim, S., Mahrin, M.N.: Metric based software project performance monitoring model. In: Proceedings of the IEEE International Conference on Open Systems (ICOS). IEEE (August 2015)
15. Entreprises, N.: Nagios Documentation. http://www.nagios.org/documentation (2014)
16. Fréville, A.: The multidimensional 0–1 knapsack problem: An overview. European Journal of Operational Research 155(1), 1–21 (2004)
17. Friedman, J., Hastie, T., Tibshirani, R.: The elements of statistical learning, vol. 1. Springer series in statistics New York (2001)
18. Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M.: Internet of things (IoT): A vision, architectural elements, and future directions. Future Gener. Comput. Syst. 29(7) (September 2013)
19. Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M.: Internet of Things (IoT): A vision, architectural elements, and future directions. Future generation computer systems 29(7), 1645–1660 (2013)
20. Jeswani, D., Natu, M., Ghosh, R.K.: Adaptive monitoring: A framework to adapt passive monitoring using probing. In: Proceedings of the 8th International Conference on Network and Service Management. pp. 350–356. CNSM '12, International Federation for Information Processing, Laxenburg, Austria, Austria (2013)
21. Jeswani, D., Natu, M., Ghosh, R.K.: Adaptive monitoring: Application of probing to adapt passive monitoring. Journal of Network and Systems Management 23(4), 950–977 (2015)
22. Katsaros, G., Gallizo, G., Kübert, R., Wang, T., Fitó, J.O., Henriksson, D.: A Multi-level Architecture for Collecting and Managing Monitoring Information in Cloud Environments. In: Leymann, F., Ivanov, I.I., van Sinderen, M., Shishkov, B. (eds.) Proceedings of the third International Conference on Cloud Computing and Services Science, CLOSER. SciTePress (2011)
23. Leskovec, J., Rajaraman, A., Ullman, J.D.: Mining of massive datasets. Cambridge university press (2014)
24. Massie, M.L., Chun, B.N., Culler, D.E.: The ganglia distributed monitoring system: design, implementation, and experience. Parallel Computing 30(7) (2004)
25. Montgomery, D.C., Runger, G.C.: Applied statistics and probability for engineers. John Wiley & Sons (2010)

26. Munawar, M.A., Reidemeister, T., Jiang, M., George, A., Ward, P.A.S.: Adaptive Monitoring with Dynamic Differential Tracing-Based Diagnosis. Springer Berlin Heidelberg, Berlin, Heidelberg (2008)
27. Murphy, K.P.: Machine Learning: A Probabilistic Perspective (2012)
28. Natu, M., Sethi, A.S.: Probabilistic Fault Diagnosis Using Adaptive Probing. Springer Berlin Heidelberg, Berlin, Heidelberg (2007)
29. Ng, A.Y., Jordan, M.I.: On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. Advances in neural information processing systems 2, 841–848 (2002)
30. Padiya, T., Bhise, M., Rajkotiya, P.: Data management for internet of things. In: Region 10 Symposium (TENSYMP), 2015 IEEE. pp. 62–65. IEEE (2015)
31. Quinlan, J.R.: C4. 5: programs for machine learning. Elsevier (2014)
32. Ramos, J., et al.: Using tf-idf to determine word relevance in document queries. In: Proceedings of the first instructional conference on machine learning. vol. 242, pp. 133–142 (2003)
33. Rivera, J., van der Meulen", R.: Gartner says 4.9 billion connected 'things' will be in use in 2015, (2014), http://www.gartner.com/newsroom/id/2905717
34. Roman, R., Najera, P., Lopez, J.: Securing the internet of things. Computer 44(9), 51–58 (2011)
35. Schölkopf, B., Smola, A.J.: Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT press (2002)
36. Sharma, K., Sharma, A., Joshi, D., Vyas, N., Bapna, A.: A review of text mining techniques & applications. International Journal of Computer (IJC) 24(1), 170–176 (2017)
37. Sicari, S., Rizzardi, A., Grieco, L.A., Coen-Porisini, A.: Security, privacy and trust in internet of things: The road ahead. Computer Networks 76, 146–164 (2015)
38. Tata, S., Megahed, A., Mohamed, M., Nazeem, A., El Harouni, A.: Data collection and management in iot environments: Challenges and future directions. In: Big Data (BigData Congress), 2017 IEEE International Congress on. IEEE (2017)
39. Tata, S., Mohamed, M., Megahed, A.: An optimization approach for adaptive monitoring in IoT environments. In: Services Computing (SCC), 2017 IEEE International Conference on. pp. 378–385. IEEE (2017)
40. Trihinas, D., Pallis, G., Dikaiakos, M.D.: AdaM: An adaptive monitoring framework for sampling and filtering on IoT devices. In: IEEE International Conference on Big Data. IEEE Computer Society, Los Alamitos, CA, USA (2015)
41. Zarko, I.P., Pripuzic, K., Serrano, M., Hauswirth, M.: IoT data management methods and optimisation algorithms for mobile publish/subscribe services in cloud environments. In: Networks and Communications (EuCNC), 2014 European Conference on. pp. 1–5. IEEE (2014)