From Cloud Down to Things: An Overview of Machine Learning in Internet of Things

Farzad Samie, Lars Bauer, and Jörg Henkel, *Fellow, IEEE*

***Abstract*—With the numerous Internet of Things (IoT) devices, the cloud-centric data processing fails to meet the requirement of all IoT applications. The limited computation and communication capacity of the cloud necessitate the edge computing, i.e., starting the IoT data processing at the edge and transforming the *connected devices* to *intelligent devices*. Machine learning (ML) the key means for information inference, should extend to the cloud-to-things continuum too. This paper reviews the role of ML in IoT from the cloud down to embedded devices. Different usages of ML for application data processing and management tasks are studied. The state-of-the-art usages of ML in IoT are categorized according to their application domain, input data type, exploited ML techniques, and where they belong in the cloud-to-things continuum. The challenges and research trends toward efficient ML on the IoT edge are discussed. Moreover, the publications on the "ML in IoT" are retrieved and analyzed systematically using ML classification techniques. Then, the growing topics and application domains are identified.**

***Index Terms*—Edge computing, embedded intelligence, embedded systems, Internet of Things (IoT), machine learning (ML).**

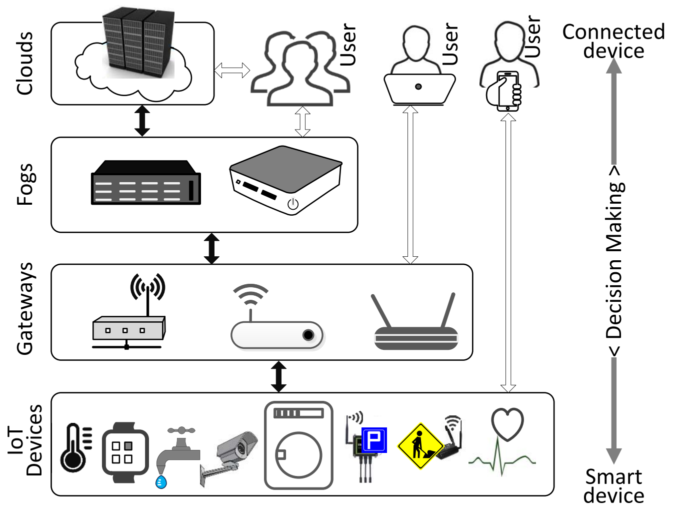


Fig. 1. Hierarchical architecture of IoT processing layers.

I. Introduction

With the rapid advancement in different technologies, such as semiconductor, wireless, and sensor, we have witnessed the interconnected devices permeating our daily lives. These embedded devices with sensors and communication means are the key components of the Internet of Things (IoT) [1], [2]. IoT offers advanced control and monitoring services in order to provide new applications or to improve the efficiency of existing ones [1].

The embedded IoT devices interact with the physical world using sensors and/or actuators. Each IoT device belongs to a wide continuum from *connected devices* to *smart devices* depending on where its data inference and decision making take place. As shown in Fig. 1, the hierarchical architecture of IoT consists of several processing layers [3]. The collected data can be acted upon at any of these layers or even by the *user*. The *connected devices* have no intelligence to process the input data and make decisions, instead, they rely on the user or other processing entities such as cloud to make the decision and control them remotely. On the contrary, *smart devices* are able to process the sensed data, assess the situation, and act independently. For instance, using a smartphone application to turn on the heating system remotely depends on the user's decision while a smart thermostat can autonomously tune the heating according to the home's occupancy or user comfort.

In the traditional cloud computing paradigm, the whole processing is performed on the cloud, making the IoT devices remote and connected sensors/actuators. However, the number of IoT devices is expanding rapidly and the massive amount of collected data is hard to manage by central clouds. The reasons are the huge workload on the IoT network and the unreliable and long latency [1], [2]. The edge computing paradigm suggests pushing the data processing to the edge of IoT (comprising gateways and embedded devices) close to where the data is collected [4], [5]. In many IoT applications, the computation can be distributed on different layers. For instance, the IoT device may perform the preprocessing on the data and transmits the intermediate results to the fog where the rest of the processing is performed [6].

Machine learning (ML) is a key enabling means for IoT which provides information inference, data processing and intelligence for IoT devices. From big-data processing on the cloud to the embedded intelligence, ML is an effective promising solution in different IoT application domains. Several surveys on ML and IoT were presented recently each of which covers a different aspect of ML in IoT as summarized in Table I.

Manuscript received July 24, 2018; revised October 2, 2018, November 23, 2018, and December 24, 2018; accepted January 7, 2019. Date of publication January 17, 2019; date of current version June 19, 2019. *(Corresponding author: Farzad Samie.)*

The authors are with the Chair for Embedded Systems, Karlsruhe Institute for Technology, 76131 Karlsruhe, Germany (e-mail: farzad.samie@kit.edu).

Digital Object Identifier 10.1109/JIOT.2019.2893866

TABLE I
EXISTING SURVEYS ON IOT AND ML

Ref.	Brief description
[7]	context aware computing, ambient intelligence, using sensor data to understand the changes in the environment, and respond accordingly, big data analytics.
[8]	basic concepts, introduction to deep learning methodologies
[9], [10]	reliable, secure, energy efficient and scalable machine learning architectures for edge devices
[11]	deep learning algorithms, IoT analytics for big data and streaming data
[12]	hardware and algorithmic techniques for embedded deep learning
[13]	hardware design, architecture, hardware-friendly algorithms, mixed-signal circuits, and advanced technologies for machine learning in image classification

In this paper, we review the usage of ML in different IoT domains, the efforts to bring the intelligence to the edge, applications of ML in both data processing and management tasks of IoT, and new trends, research directions, and challenges. We also analyze all the publications on ML in IoT using a systematic approach based on ML classification techniques.

The main contributions of this paper are as follows.

1) We investigate the state-of-the-art on ML in IoT, categorize them according to their application domain, where they stand in the cloud-to-things continuum, and what they infer from data.
2) We present a comprehensive view on different ML techniques, the characteristics, the role of different processing layers from cloud to fog down to embedded devices in inference, suitability of ML techniques for each layer and opportunities to integrate IoT devices with embedded intelligence.
3) Using an ML approach, we analyze the publications on ML in IoT by training accurate classifiers which label the publications based on the key phrases in their title and abstract. The presented details allow other researchers to reproduce the analysis with updated publications at a later time.
4) Based on a thorough analysis of both IoT and ML literature, we present challenges and potential research trends on efficient ML for IoT with a focus on edge computing paradigm.

The remainder of this paper is organized as follows. Section II presents relevant background on ML techniques and their applications. Section III reviews the application of ML in different IoT domains, categorize their role, and identify their place in the cloud-to-edge continuum. In Section IV, the open challenges and research opportunities for efficient ML in IoT are discussed. Then in Section V, we quantitatively analyze the publications on ML and IoT and use an ML approach to classify and label the publications. Finally, Section VI concludes this paper.

II. BACKGROUND

ML aims at creating models based on observations and experiences, and then use the model to predict future data or discover patterns. Depending on the type of data to create the model, ML techniques can be divided into three categories.

1) *Supervised learning* uses training datasets which are labeled with *correct* output.
2) *Unsupervised learning* uses a collection of unlabeled data to find underlying patterns.
3) In *Reinforcement learning*, the *correct* output is not available *a priori* but in a trial-and-error fashion, the predicted output can be evaluated by a positive or negative *reward* which indicates how good or bad the predicted output was.

The usage of ML can be divided into four main categories.

A. Classification

The output of classification belongs to a finite set of predefined discrete values or *classes*. Depending on the number of classes, the classification problems fall into one of these two categories: 1) binary classification: it involves with only two labels such as false/true or 0/1 and 2) multiclass classification: it involves with more than two classes. For instance, monitoring the data traffic of an IoT device and detecting whether it is participating in a DDoS attack or not is a binary classification. But human activity recognition by wearable devices (to detect "walking", "running," "sitting," "standing," etc.) is a multiclass classification.

There are several classification models which include—but not limited to—logistic regression (LoR), support vector machine (SVM), decision tree (DT), naive Bayes, *k*-nearest neighbors (KNN), artificial neural networks (ANNs), and linear discriminant analysis (LDA). Note that LoR—despite its name—is a classification and not regression model. The internal operations of some classification models are schematically described in Fig. 2.

Deep neural networks (DNNs), unlike the ANN, do not rely on the selected features by the designer. Instead, they learn the features automatically and directly from the data. To do so, DNN usually requires more hidden layers in order to hierarchically detect the features from different layers of abstraction. The higher number of hidden layers make these networks "deep." Convolutional neural networks (CNNs) are a class of DNNs which are the main classification model for image classification and computer vision. CNN preserves the spatial structure of subfigures in a hierarchical fashion and uses convolution filters to detect feature maps. In recurrent neural networks (RNNs), the output of some neurons can flow back and feed the inputs for the neurons of the same layer or previous layers (see Fig. 3). These feedback loops in the network structure help the RNN to persist time series information and temporal patterns. RNNs perform very well for speech recognition, video classification, and other applications whose inputs or outputs are sequences.

Ensemble models consist of several complementary classifiers that might even be weak individually but can be combined to form a strong and accurate classifier (see Fig. 2). For instance, random forest (RF) ensembles several DTs each trained with different independent subsets of data. Boosted trees (e.g., gradient boosting trees) also ensembles

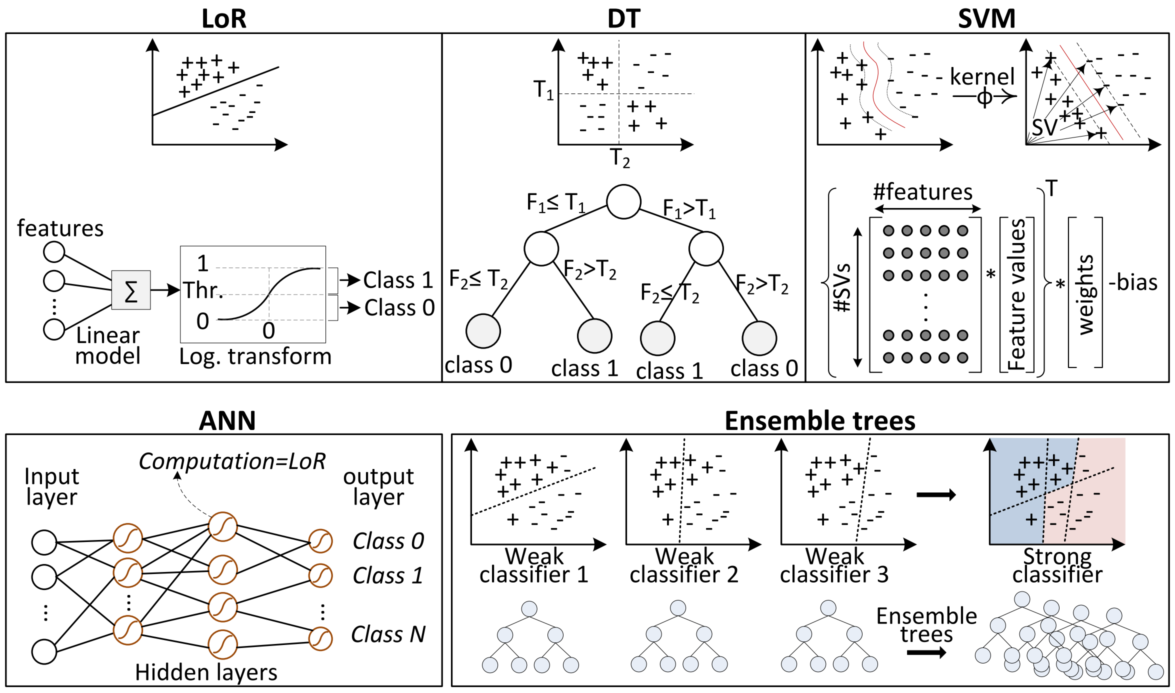


Fig. 2. Internal operations of some classification models.

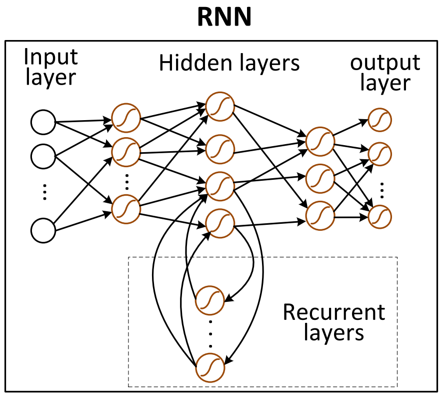


Fig. 3. RNN architecture.

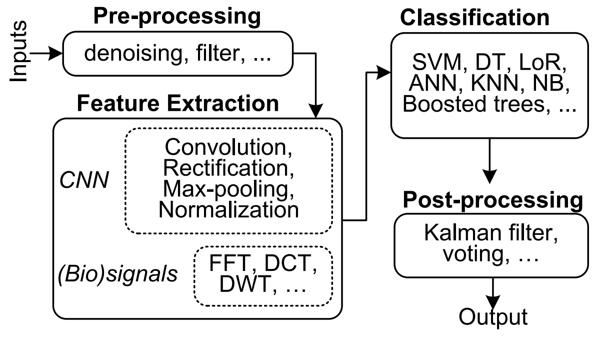


Fig. 4. Different stages of classification task at runtime.

trees but the trees are not trained independently in parallel. The new tree is trained in a way that mispredicted inputs of the last tree are prioritized. Ensemble models are more widely used for classification but can be applied for regression too.

Some classification models, such as RF, ANN, or kNN are intrinsically able to deal with multiple classes. Some models, such as SVM and LoR shall use multiple binary classifications in order to handle a multiclass problem. Two main approaches are as follows.

1) *One-Versus-All (OvA):* In this approach, one binary classifier is used to separate each label from all other labels. Therefore, this approach requires N classifier to handle an N-class problem.
2) *One-Versus-One:* In this approach, one binary classifier is used to separate each pair of classes. Hence, it results in $N \times (N-1)/2$ classifiers in an N-class problem.

The typical pipeline for ML classification includes multiple stages as shown in Fig. 4: preprocessing (i.e., filtering, denoising, etc.), feature extraction (e.g., fast Fourier transform, discrete wavelet transform, discrete cosine transform, max pooling, rectification, convolution, normalization, etc.), classification, and finally post-processing to smooth out the output (e.g., Kalman filter, majority voting, etc.). Each stage contains different computation operations each of which could benefit from efficient hardware implementation or software optimization.

B. Regression

Regression models the relationship between input variables and output to provide a predictive model. The output of regression belongs to a continuous range. For instance, in a smart grid application, a regression model can predict energy usage based on past information. Linear regression (LiR) is the most widely used model for regression which formulates a linear equation of input variables (e.g., $p = \sum f_i \cdot w_i$ where f_i is the value of ith feature, w_i is the weight of that feature and p is the predicted value). Some of the models such as DT and KNN can also be used for regression even though their common usage is for classification. Polynomial regression is a similar model with the difference that the power of features (i.e., f_i) can be more than 1. This model has the risk of over-fitting which makes the model perform very good for the test data but it is generalized for the new data.

LiR model may suffer from multicollinearity (i.e., high correlation between features). It causes high variance in the value of features' weight when performing the training phase multiple times. To avoid this issue, two similar regression models are typically used, e.g., Ridge regression and least absolute shrinkage selector operator (LASSO) regression. While these models also use linear predictors, they constrain the value of features' weights in the training phase. Consequently, the large weights and variance between them are avoided. An extra advantage of LASSO is that it performs the *feature selection* by removing the unimportant features as it makes the small weights to become 0. Hence, the difference between Ridge, LASSO, and LiR is the weights of the features in their linear model. When the number of features is very large, LASSO regression is more suitable.

C. Clustering

Clustering is an unsupervised learning method that extracts the structure and hidden patterns from data. It clusters the data into few groups which have similar features and common structure and the data points in different clusters are dissimilar. The main applications of clustering include marketing (e.g., discover customer segments to target lookalike customers with similar advertisements), recommender systems (e.g., retail, music, and video-on-demand), outliers detection (e.g., detecting fraudulent patterns in insurance, smart grid etc.), content management (e.g., clustering the articles, books and other contents), etc.

K-means is one of the most popular models for this task which clusters the input data into k clusters.

Given the number of clusters, k, the k-means algorithm follows these steps.

1) Select k centroids (e.g., randomly) for the clusters.
2) Find the closest centroid for each data point and assign the data to this cluster. The metric to find the closest centroid is often the Euclidean distance.
3) Update the centroids by calculating the average position of data points in the clusters.
4) Repeat steps 2 and 3 until the position of centroids converge.

The complexity of k-mean clustering is $O(n)$, but the disadvantage is that it requires prior knowledge of k.

Hierarchical clustering is another technique which has more flexibility for the number of created cluster. The hierarchical clustering follows several steps.

1) Start with N clusters (e.g., one for each data point if it is small).
2) Merge the two closest clusters.
3) Update the calculated distance between clusters.
4) Repeat steps 2 and 3 until there is only one cluster.

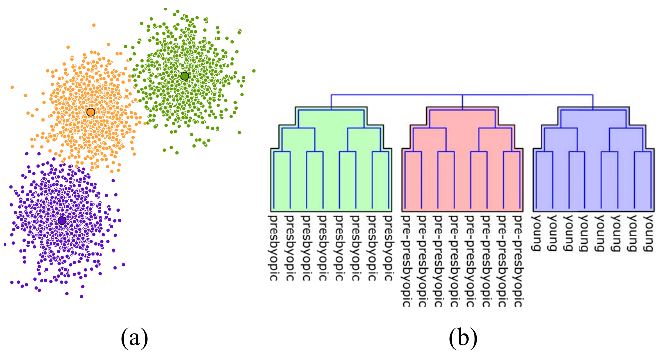


Fig. 5. Examples of (a) k-means clustering with $k = 3$. (b) Hierarchical clustering.

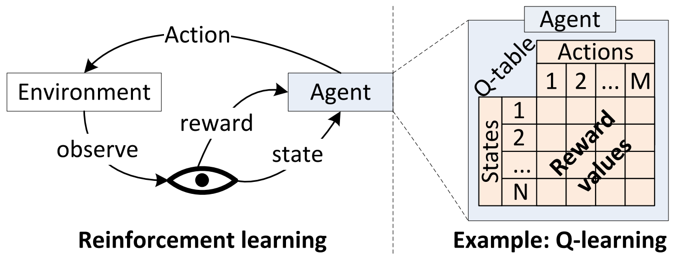


Fig. 6. Components of reinforcement learning and their interactions.

Fig. 5(a) and 5(b) show examples of k-means and hierarchical clustering, respectively.

D. Reinforcement Decision Making

Reinforcement learning is mainly used for decision making. The outcome of each decision will be considered as the reward or penalty to evaluate the decision and update the decision function. Reinforcement learning does not require offline training and hence is suitable for IoT systems with no prior knowledge about the relation between their input and correct output (e.g., decision). The learner (known as *Agent*) learns from experiences and trial-and-error. It takes actions, observes the reward of the action plus the change in the state of the environment, and then improves its decision making accordingly (see Fig. 6).

The reinforcement learning problems can be formalized by the Markov decision process which is described by the following.

1) A finite set of *states*.
2) A finite set of possible *actions* in each state.
3) The *transitions* between states.
4) The *rewards* associated with each transition.

The objective of the reinforcement learner is to maximize the sum of rewards in the long run

$$\sum_{t=0}^{t\to\infty} \gamma^t \times r(s(t), a(t)) \tag{1}$$

where $r\big(s(t), a(t)\big)$ denotes the reward associated with taking action a at state s. The γ is a discount factor between 0 and 1 which determine the importance of future rewards. To solve this Markov decision process problem, we need to find a policy to take the best action at each state such that the accumulated reward is maximized [14].

Q-Learning is the most popular reinforcement learning algorithm as it is easy to implement and proven to converge to the optimal solution (see [15] for more details). It creates an action-value function (called *Q*-function) which estimates the expected reward of taking a certain action while being in a certain state. The initial value of *Q*-function of all inputs is a fixed arbitrary value. At each timestamp, the function is updated according to the taken action and the received reward

$$Q(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \times \left(\underbrace{\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\substack{\text{discount}\\\text{factor}}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}}}_{\text{learned value}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \right) \quad (2)$$

where α—the learning rate—indicates the weight of learned value. When setting $\alpha = 1$, the old value is completely replaced with the learned value. The updating *Q*-function is basically an exponential moving average of learned values. Note that *Q*-learning considers the achievable reward for subsequent actions, too. The *Q*-function is usually implemented as a table (called *Q*-table) whose size is bounded by the number states multiplied by the maximum number of actions for a state (see Fig. 6).

The reinforcement learning, unlike other ML models, faces a tradeoff between *exploitation* and *exploration* at runtime: whether to *exploit* the actions that have been already tried and shown to be effective, or to *explore* new actions and discover more effective ones. For instance, in *Q*-learning, we can either choose the action with the highest expected reward or take a random action some percentage of the time. Some techniques use the "ϵ-greedy" method for exploration which implies that with the probability of ϵ they choose a random action with uniform distribution instead of exploiting the so-far best-known action.

Characteristics of Machine Learning Techniques: IoT devices at the edge are usually resource-constrained (i.e., limited energy budget, memory, and computation capability [16]). On the other hand, the resource usage of ML techniques highly depends on the application. However, some techniques are inherently resource-hungry or light-weight. Lane *et al.* [17] analyzed the deep learning algorithm on wearable IoT devices to process audio and image data. They investigate the performance, resource usage and the execution characteristics of the DNN and CNN models.

Table II summarizes some of the characteristics of ML techniques. These properties make some ML techniques more suitable for some processing layers and less suitable for others.

TABLE II
CHARACTERISTICS OF SOME ML TECHNIQUES

Clustering	pros.	The designer does not require deep understanding of the input data and their features
	cons.	not embedded-friendly, computation intensive, memory intensive, requires very large amount of data to perform well
Reinforcement	pros.	low computation and low memory requirement, hence it is embedded-friendly. It can be deployed with no prior knowledge about input data
	cons.	learns from runtime experiences, hence may make inefficient decisions
Boosted / Ensemble Trees	pros.	The accuracy can be further increased even after deployment. Traversing trees can be parallelized.
	cons.	Large memory requirement to keep the trees data structure

III. MACHINE LEARNING IN IOT

A. Machine Learning in Data Processing

ML plays an important role in providing the application service to the user. Most of IoT applications require to classify the input data as the output of their service. Many application domains benefit from ML advantages as discussed in the following.

1) Healthcare and Well-being: Wearable health monitoring devices are able to capture biomedical signals from the user's body. The collected data can be used to detect early diseases, activity recognition, monitoring lifestyle, assist patients to cope with daily difficulties, etc.

The heart electrical activity signal or electrocardiogram (ECG) can be used to detect stress, cardiovascular diseases, etc. These applications train supervised classifiers, such as Bayesian network, SVM, LoR, etc. Azariadi *et al.* [18] used SVM classifier to detect heartbeat abnormality from ECG signals on the IoT device. In [19], cardiovascular diseases are predicted from the ECG signal using classification techniques. The ECG signal is captured from wearable devices and transmitted to a remote processing unit. The classifier is based on the Bayesian network model with four classes: 1) normal; 2) premature atrial contraction; 3) premature ventricular contraction; and 4) myocardial infarction. In [20], the level of driver's stress is detected from his/her ECG signal. First, several features are extracted from the ECG, including heartbeat, heartbeat variation, etc. Then, several classifiers have been used to detect three levels of stress, among which naive Bayes achieves the highest accuracy. Another application of classification in stress detection is presented in [21]. The heartbeat rate is considered as the main feature and two classification models (LoR and SVM) are evaluated. The design is intended for IoT devices.

The brain electrical activity signal or electroencephalogram (EEG) can be used in several applications for health or well-being purposes. In [22], a light-weight supervised classification model is proposed to predict epileptic seizures by classifying the EEG input data using LoR and gradient boosting trees. Several features form time-domain and frequency-domain of EEG signal are used for classification. The LoR classifier, which is low-overhead, is implemented on the IoT devices, while the XGB classifier is offloaded to the gateway. In [23], the drowsiness level of the driver is classified on a wearable IoT devices which uses EEG signal. The supervised classifier is based on SVM.

In [24], the *k*-means clustering is used to analyze the physiological data and discover the patterns in the collected data from the wearable health devices. The algorithms are implemented on low resource fog nodes and gateways, close to the data source.

2) Smart Home: A self-learning home management system is presented in [25] which uses different ML techniques for price clustering and price prediction. Detection of human presence, activity recognition, self-organizing home appliances, controlling air-conditioning based on user comfort, etc., are enabled by ML. Particularly, the activity recognition has received considerable attention [26] due to its wide application for assisted living, home automation, etc.

For an activity recognition application, [27] uses embedded classification to process the accelerometer data on the IoT device instead of transmitting raw data to the cloud server. They consider six activities and use an SVM model for classification. To deal with the multiclass problem, they use an OvA approach. The activity recognition application may fall into assisted living and healthcare domains as well.

Idowu *et al.* [28] used multiple regression models to forecast the heat load for a multifamily smart building. This forecast can help to optimize the energy consumption of building in the smart grid era.

3) Smart Agriculture: IoT solutions have been proposed to improve productivity and reduce the cost of maintenance in agricultural systems and farming.

In [29], the image classification based on CNN is used to detect pests and diseases of plants in a farm. In a similar application, Mwebaze and Owomugisha [30] used linear support vector classifier, kNN and ExtaTree models to detect the crop diseases from plant images. The images can be taken with a smartphone and then uploaded to a server where the classification takes place. The application involves a multiclass classification where four diseases and the health status will be detected.

Patil and Thorat [31] monitored the environmental parameters for a grape plant including temperature, humidity, leaf wetness, etc. The captured data is transmitted to a server where an application identifies the chances of grape diseases in its early stages and notifies the farmer. In [32], CNN is used to detect the flowers and seedpods from the images taken and sent from the plant in the farm to monitor the growth rate and environmental parameters.

In summary, the aims of these solutions can be categorized into two main goals: 1) detection of diseases and pests [29]–[31] and 2) maintaining the required environment of plants such as temperature, moisture, soil's condition, etc. [32]. ML techniques have been utilized to assess the environment or plant using either surveillance means, such as image and video [29], [30] or environmental sensors, such as temperature, humidity, etc. [31].

One of the drawbacks in these existing solutions is the dependency on the cloud servers. The complexity of data processing in these applications calls for efficient and powerful hardware to enable ML on embedded IoT devices.

4) Smart Industry: Smart industry can benefit from its available data and ML techniques to improve its maintenance management, quality assurance, scheduling, etc. Mangal and Kumar [33] used ML classification in a *Bosch* assembly line to predict failure in the components and therefore improve the production yield. One challenge is that their training dataset contains more than 4000 features. They divide the dataset into two subsets and trained a model for each. Then each model is used to calculate the prediction probability for the other subset. This value is added as a new feature in the final classification model which is based on extreme gradient boosting trees (XGBoost).

The application of ML in forecasting the supply chain demand is studied in [34]. ANN, RNN, SVM, and LiR models are investigated in this paper. Note that the problem is to forecast the demand (a continuous variable) and hence it is a regression problem. As mentioned before, the SVM and neural network models can be used for regression too.

In [35], ML classifiers are used to predict the maintenance issues in the manufacturing process and hence reduce the downtime and associated costs. They used SVM and kNN models to detect the failures that are caused by accumulative wear and tear effects. Two challenges in this problem are: 1) unbalanced data as the number of failures in the dataset is low and 2) to predict and prevent the failure, we need to detect before it happens when the observations show no fault. The proposed solution in [35], is to use multiple classifiers for each of which the training dataset is manipulated as follows: a specific number of observations before the actual failure are labeled as "failure." Increasing this number results in a more conservative maintenance recommender system. In [36], unsupervised clustering is proposed for discovering degradation in cyber-physical systems (CPSs). Afterward, the failures and root-cause can be predicted and reasoned.

5) Smart Grid: Using the connected meters, smart grid aims to improve the efficiency of the power grid. The main concepts in the smart grid domain (dynamic pricing, integration of renewable energies, etc.) are shown to benefit from different ML techniques including classification, regression, reinforcement learning, etc.

Wei *et al.* [37] presented a *Q*-Learning method to manage the battery charge/discharge from renewable sources to minimize the cost of power from the grid. O'Neill *et al.* [38] took advantage of *Q*-learning to estimate the future energy price in smart grid (with energy real-time price), and then schedule the home appliances to minimize the energy cost. Thapa *et al.* [39] used reinforcement learning principle to schedule the loads in the smart grid where the consumers must decide whether to request for power supply in the current time slot or not. Sharma *et al.* [40] studied the correlation between weather parameters and solar intensity. Then, they use ML regression models to predict the solar intensity based on the weather forecasts parameters. Chen *et al.* [41] exploited multiple regression models based on the ensemble of extreme gradient boosting trees (XGboost) and Ridge regression to predict the annual energy consumption of households.

In [42], four regression models based on ANN, SVM, RF, and XGBoost are evaluated for the electric load forecasting in the smart grid. Results show that the time scale for forecasting affects the accuracy.

Moreover, energy fraud detection—one of the issues in the conventional grid—can benefit from ML in the smart grid realm. For instance, Ford *et al.* [43] used ML to model the consumption behavior of customers and then detect the abnormal behaviors with fraudulent activities.

In some application domains, such as smart grid, multiple similar learners may exist in the IoT network in the same vicinity (e.g., each consumer may predict the real-time price). This opportunity enables collaborative and distributed learning which can improve the accuracy and reduce the overhead (see Section IV).

B. Machine Learning in Management and Organization

Besides the IoT services, ML can be used in management and organizing IoT systems/devices. The management tasks, such as resource allocation, power management, scheduling, security state assessment, etc., can benefit from efficient ML approaches.

1) Security and Privacy: Due to the pervasive presence of IoT devices in people's lives, security, and privacy are the major concerns of users. To improve the privacy of user or assure the security state, ML techniques can come into play. The efforts can be divided into two categories: 1) detect and 2) prevent. Detecting device tampering, data injection, malware, abnormal behavior [44], as well as intrusion detection [45] are among the first category.

Wei *et al.* [46] used deep learning to detect false data injecting in the smart grid. It exploits the hidden correlation between data readings—due to the physical coherence—and evaluates the trustworthiness of data. Sharmeen *et al.* [47] consider the use of mobile devices in industrial IoT and analyze malware detection aspects on these mobile devices. They analyze datasets, feature extraction techniques, and classification models with regard to their benefits and limitations.

Sun *et al.* [48] considered ML to model the attack activities of botnets in IoT. They identify the activity pattern and the structure of botnets by clustering the attackers.

Other research works attempt to prevent security/privacy vulnerabilities. To improve the privacy of users, Jeong *et al.* [49] suggested not to transmit the raw data to the cloud. Instead, only partially processed feature data that are needed for NN are sent. Chauhan *et al.* [50] used RNNs to authenticate users based on breathing acoustics on medium-range IoT devices (e.g., smartphone or single-board computers). The goal is to detect if the sample belongs to predefined registered users.

2) Power/Battery Management: IoT embedded devices have various parameters that can be adapted at runtime to improve their energy efficiency. Their *sensing* task, *processor*, *processing* task, and *communication* task can benefit from ML techniques to determine the efficient operating parameters. In [51], the input data sampling rate is adapted at runtime based on reinforcement learning to save energy on sensor nodes. The algorithm is based on Q-learning and can be implemented on either cloud, gateway or IoT device. Golanbari *et al.* [52] used a regression model to find the optimal power supply voltage for chips to improve the energy efficiency of low-power IoT devices at runtime. Chafii *et al.* [53] used reinforcement learning to enable a dynamic spectrum access procedure in narrow band IoT (NB-IoT) wireless modules which enhances battery lifetime of devices. Kraemer *et al.* [54] evaluated ML techniques for predicting the solar energy for energy-harvester low-power IoT nodes based on public weather forecast.

3) Resource Allocation/Task Scheduling: Resource management/allocation problems are generally formulated as constrained optimization problems. Sometimes these problems are intractable or have unscalable solutions. Even though ML techniques cannot guarantee the exact (optimal) solution, they are low-overhead and scalable solutions.

Cui *et al.* [55] presented a framework that decides whether to process the data on the gateway, at the edge, or on the cloud server. He *et al.* [56] used a deep reinforcement learning (DRL) method for cache allocation on content-centric computing nodes and determining transmission rate between nodes. Iranfar *et al.* [57] used ML for allocating hardware accelerators and processing cores to the video streams on an IoT edge device. In [58] a reinforcement learning algorithm is presented for the scheduling problem in distributed systems. Min *et al.* [59] proposed a reinforcement learning technique in which IoT device—considering its battery level—determines "how much" to offload its computation.

Demirel *et al.* [60] developed a DRL method to schedule the sensor-to-controller communication in the networked control systems where there are multiple independent controlled subsystems that are connected through a shared network.

An unsupervised clustering method is suggested for grouping low power fog nodes in a 5G network [61]. Each group then is assigned to a high power fog node in order to reduce the latency of communication.

The state-of-the-art usage of ML in IoT can be categorized based on different aspects. Each of these aspects is important from the design exploration perspective. These aspects include the ML techniques and models, application domain, the processing layer, the type of input data to the model, etc. We summarize these categorizations in two tables as follows: Table III summarizes the state-of-the-art using ML in IoT, categorized according to the ML techniques, where they belong in the cloud-to-edge continuum, and what they use ML for. The reinforcement learning has been mainly used for the management on the constrained devices. In addition, the clustering technique is mainly used on the cloud server. Classification, on the other hand, has been used on the cloud, edge devices and distributed between multiple layers.

Table IV summarizes the state-of-the-art using ML in IoT, categorized by the exploited ML technique and the type of inputs. The type of inputs is divided into five categories: 1) *Multimedia* including image, video, speech, and acoustic data; 2) *(Bio-)signal* including ECG, EEG, and other signals captured from body; 3) *Env. Parameters* including temperature, humidity, etc.; 4) *Power Grid* information; and 5) *Other*. The type of input data usually affects the *preprocessing* stage as well as the feature extraction stage (see Fig. 4). For instance, when the input data is a signal (e.g., ECG and acoustic), the application involves filtering or transformations (e.g., FFT). These operations perform more efficiently on the hardware than software. Hence, having co-processors and accelerators on the IoT edge devices improves the energy efficiency significantly. Moreover, when the input data is *multimedia*, transmitting the whole data to the cloud server

TABLE III
STATE-OF-THE ART USAGE OF ML IN IoT FOR MANAGEMENT AND ORGANIZATION (MNG.) AND APPLICATION DATA PROCESSING (ADP)

	Cloud	Things	Distributed	
Classification	[46]§ detects the false data injection in smart grid using deep learning, [44]§ detects abnormal behavior of IoT devices (when some data are modified), [45]§ detects intrusion in IoT network layer using expanded version of LoR, [43]† uses ANN to detect fraud in smart grid.	[47]§ $\star$ malware detection in industrial mobile IoT devices	-	**Mng.**
	[33]* uses gradient boosting tree classifier to detect the failure in the products in assembly line, [29]* uses deep CNN to detect pests and diseases of plants from images, [32]* uses CNN to detect and monitor the flowers and seedpods from images.	[18]‡ detects abnormally in heart from ECG signal using SVM, [23]‡ detects driver's drowsiness from EEG signal using SVM, [50]§ uses RNN to authenticate users based on breathing acoustics, [27]‡ $\P$ uses SVM to recognize the human daily activities.	[22]‡ uses two classifiers (one on IoT device and one on the gateway) to predict epileptic seizure from EEG signals, [49] performs some layers of ANN on IoT device and offload the rest to cloud, [62] a framework to decide which parts of DNN must be offloaded to the cloud, [63] distributes CNN layers between cloud and IoT devices, [64] distributes CNN layers between cloud and edge nodes.	**ADP**
Regression	[40]¶ † uses SVM and linear regression to predict solar intensity based on weather forecast for smart homes	[52] selects the optimal supply voltage for IoT chips at runtime, [54] predicts the solar energy for energy-harvester IoT devices based on public weather forecast.	-	**Mng.**
	[41]† uses ensemble to predict the annual energy consumption of households, [28]† $^{+}$ uses regression models to forecast heat load in multi-family buildings to optimize energy consumption in smart grid, [34]* forecasts the demand in supply chain of smart factory using SVM, ANN, RNN and linear regression, [42]† uses SVM, ANN, RF and XGBoost to forecast the load in smart grid	-	-	**ADP**
Reinforcement	[37]¶ † uses Q-Learning to manage the battery charge/discharge from renewable sources in smart grid, [38]¶ † uses Q-learning to estimate the price of power in smart grid and then schedule the home appliances, [65] uses cooperative reinforcement learning for power allocation in device-to-device communication.	[59] manages computation offloading for IoT devices based on their battery, [51] adapts input sampling rate on IoT device at runtime, [57] allocates hardware accelerators and cores to the video streams, [60] schedules the sensor-to-controller communication, [53] schedules wireless spectrum access in NB-IoT to enhances battery lifetime, [58] schedule tasks in heterogeneous distributed systems.	[56] uses deep reinforcement learning for cache allocation and transmission rate control between IoT computing nodes (e.g. fog nodes).	**Mng.**
	[66]$^{+}$ uses deep reinforcement learning for localization in smart buildings.	-	-	**ADP**
Clustering	[25]¶ self-learning home management system for price clustering and price prediction of power, [48]§ clusters the botnet attack activities, [61] clusters the fog nodes in 5G network to reduce the latency.	-	-	**Mng.**
	[36]* uses clustering to discover degradation in CPS.	-	[24]‡ uses k-means clustering for physiological data on fog and gateways.	**ADP**

¶ Smart Home, $^{+}$ Smart Building, † Smart Grid, ‡ Healthcare, * Industry, § Security, * Agriculture

Mng.: Management & Organization, **ADP**: Application Data Processing

is inefficient due to its overhead on the transmission energy of the IoT device and its load on the IoT network. It requires to enable the image classification and recognition application on the edge devices. In addition, and especially for multimedia data, approximate computing paradigm offers opportunities to improve the efficiency of ML techniques in terms of reducing the computational complexity, saving power consumption and improving the performance.

IV. New Trends and Open Challenges

A. Toward Machine Learning at IoT Edge

IoT systems face an increasing need for data processing and decision making at the IoT edge which requires pushing the computation, including ML algorithms, close to the embedded devices. However, the IoT devices at the edge are resource-constrained (i.e., limited energy budget, computation capability, and memory) [16]. This challenge calls for: 1) more efficient hardware platforms to carry on ML algorithms and 2) optimizing ML techniques to reduce their power consumption, memory requirement, and computation intensity.

Shafique *et al.* [9] presented an overview of the architectures for efficient deep learning on IoT devices. Venkataramani *et al.* [75] studied the computational requirements of deep learning techniques and capabilities of current embedded computing platforms. To bridge the computational gap, embedded devices shall benefit from approaches, such as hardware accelerators, approximate computing, and emerging post-CMOS technologies.

B. Efficient Hardware for Embedded ML

ML algorithms can be accelerated using application specific instruction processor (ASIP), application specified integrated circuit, FPGA, or co-processors.

An ASIP accelerator for ML is presented in [74] which also takes advantage of approximate computing to further improve the energy efficiency. Zhang *et al.* [67] presented design techniques for implementing DNNs on FPGAs to achieve energy efficiency for image and video processing.

While most of the hardware accelerators target computational primitives, Chen *et al.* [76] focused on the impact of memory on accelerators' efficiency and introduce a series of hardware accelerators for neural networks to minimize memory transfers.

C. Approximate Computing in ML

Approximate Computing has shown promising opportunities to improve performance and energy efficiency of both hardware and software implementation at the cost of quality degradation. The main constraints on the embedded IoT devices include computation capability and memory. In different ML techniques, either of feature extraction and classification stages might be computation or data intensive. For instance, boosted tree techniques are usually data intensive as they consist of many small trees whose information must be read during classification. Moreover, in some models, the feature extraction involves with FFT, DWT, convolution, etc., which are computation intensive.

Chippa *et al.* [77] demonstrated that the SVM classifiers can be approximated by ignoring less important support vectors and features. Consequently, the number of required dot products reduces at the cost of loss in classification accuracy. In [71] and [78], approximation methods for ANWs are presented which first determine the less critical neurons and then approximate their computation to achieve energy efficiency and memory access reduction under quality constraints. To reduce the number of multiplications in CNNs, Huan *et al.* [79] predicted near-zero multiplications and approximate them. Sheng *et al.* [80] exploited approximation in feature extraction of CNN for image classification by presenting a quantization-friendly separable convolution architecture. Gao *et al.* [70] carried out some basic arithmetic operations (e.g., DCT, FFT, SVM, *k*-means, and KNN) by approximate function units and show the potentials to save the power consumption.

Venkataramani *et al.* [81] introduced scalable-effort classifiers, cascaded classifiers with growing accuracy and complexity. Depending on the difficulty of input data, the number of stages used for classifying is adjusted at runtime.

Previous work, despite their accomplishments, mainly are ad hoc methods which target specific applications.

D. Distribute Computation Among Layers (Offloading)

As shown in Fig. 1, the IoT system has a hierarchical architecture of computational entities. From embedded devices to the gateways to the fog computer and finally cloud servers, each layer is able to perform a certain amount of computation (computational capability increases with the same order). Offloading the computation to a higher level introduces a tradeoff between the computation and communication costs as well as latency.

Castillo *et al.* [63] proposed to distribute the layers of deep CNN between embedded device and cloud. This paper shows that the task of image recognition can be distributed among multiple smart cameras to improve response time and energy efficiency. Li *et al.* [64] considered offloading parts of learning layers from IoT devices to the cloud in a deep learning application. The intermediate CNN layers are performed on the edge nodes and the rest of the layers are offloaded to the cloud servers. In [22], the feature extraction and LoR classification are performed on the embedded device while the other classifier—gradient boosting tree—is implemented on the gateway due to its memory consumption. Eshratifar and Pedram [62] broke down the computation of DNN, perform some parts on the edge device and offload the rest to the cloud server.

E. Distributed and Collaborative Machine Learning

Except the reinforcement learning, other ML models require a very large amount of data for training which is usually done on a central cloud at design time. A promising alternative is distributed ML where the learning model parameters are obtained on different edge nodes without sending the raw data to the central cloud. Wang *et al.* [82] presented a gradient-descent-based distributed learning in which local learning parameters are obtained from edge devices, then aggregated on another node, and sent back to the edge devices as global parameters. The frequency of global aggregation defines a tradeoff between computation cost on edge devices and communication cost on the network infrastructure. A control algorithm is proposed to determine this frequency such that the loss function of the learner is minimized under a resource budget constraint.

TABLE IV
ML TECHNIQUES EXPLOITED IN THE STATE-OF-THE-ART CATEGORIZED BY THEIR TYPE OF INPUTS

type of inputs	**SVM**	**DT**	**LoR**	**NB**	**kNN**	**Ensemble trees**	**ANN**	**DNN/ CNN/RNN**	**Reinforcement**	**LiR**	**k-Means**
Multimedia	[30]				[30]	[30]	[49]	[32], [50], [63], [12], [62], [64], [32], [29], [67]			[24]
(Bio-)Signal	[18], [20], [21], [23], [27]	[20]	[20], [22], [21]	[20]	[20]	[22], [20]					
Env. Parameters	[68], [40]									[40]	
Power Grid	[42]					[41], [42]	[41], [43], [42]		[37], [69], [38]	[28]	
Other	[34], [70], [54], [44], [35]		[33], [45]		[70], [54], [35], [45]	[33]	[34], [54], [71], [72]	[34], [25]	[56], [51], [60], [73], [53], [66], [57], [59], [58], [65]	[52], [74]	[70], [44], [36], [25], [74]

Portelli and Anagnostopoulos [83] introduced an online distributed learning model in which the edge devices train their local regression model independently, and then the parameters of models are sent to a central cloud. An adaptive method aggregates the regression models to improve accuracy.

As mentioned before, one drawback of reinforcement learning is the consequent penalty of decisions in some experiences. Reinforcement learners—without knowing it *a priori*—make inefficient decisions and then learn from them. In a collaborative learning approach, the embedded devices may share their experiences with other devices in order to avoid inefficient decisions by other devices. In an application-specific attempt, [73] uses distributed multiagent reinforcement learning for traffic light control. Marinescu *et al.* [69] introduced another case-study for distributed reinforcement learning in the smart grid where electric vehicles must determine when to charge their batteries based on the demand prediction. Due to the distributed nature of some IoT applications, online collaborative learning is one of the promising research direction in ML for IoT. In [65], a cooperative reinforcement learning algorithm is used for runtime power allocation in device-to-device communication in order to reduce the interference with the cellular users. The neighboring devices share their value functions to increase the *explored* space.

V. ANALYSIS OF PUBLISHED RESEARCHES

We aim to quantitatively analyze the number of research works that have been published as conference papers or journal articles that target the domain of ML and IoT. We use the available APIs from academic databases and search engines including *IEEE Xplore*, *ACM* digital libraries, *arXiv* repository, and *Elsevier*. We retrieve all the entries that matched the keywords including "machine learning" and "Internet of Things" or "IoT." We collect the following information from the result queries: *title*, *abstract*, *date*, and *key terms* (when provided).[1]

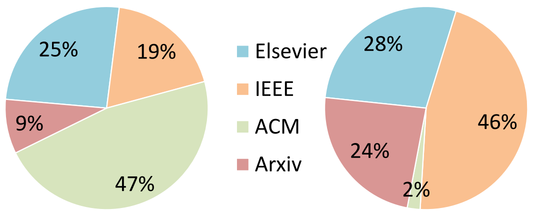


Fig. 7. Left: Total retrieved records from search. Right: Papers on both ML and IoT.

In total, we retrieved more than 19 000 records. Then, we processed the records and cross-checked the "title" and "abstract" with the keywords in ML and IoT domain. The number of records that have the keyword of both ML and IoT reduced to 605 records. Fig. 7 shows the share of each publication database in the total retrieved records (on the left), and the share of filtered publication with both IoT and ML content.[2]

Fig. 8(a) shows the publication year of the retrieved papers. The number of publications has grown exponentially in recent years. However, the growth rate for the publications on IoT is higher than ML. Note that the data is collected in early 2018, and thus the number of publications in 2018 is not available yet. Fig. 8(b) shows the publication year of the papers on both ML and IoT.

A. Classification of Papers Using Machine Learning

Since there are a large number of records retrieved from the inquiries, we exploit a systematic approach based on ML to process the records and classify/categorize them. Each paper may have multiple labels (e.g., privacy and smart home), hence, we must train a binary classifier for each label. We randomly select 250 papers from the collected data and manually inspect them by reading the title, abstract, and "terms" (if available) and labeling them. We associate each label with several keywords. For instance, the keywords for the security

[1]Key terms are only provided from *Elsevier* and *IEEE*. Moreover, *ACM* only provides title and date.

[2]The records from the ACM do not contain the abstract. Therefore, fewer papers could be categorized as ML and IoT.

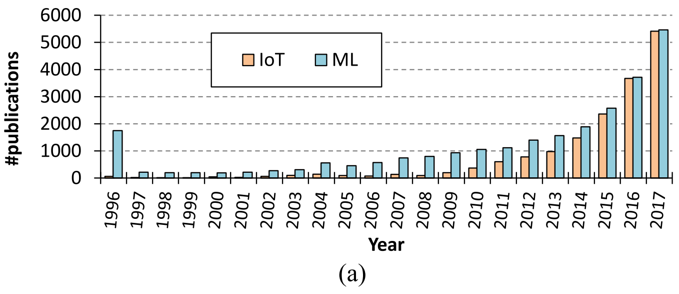
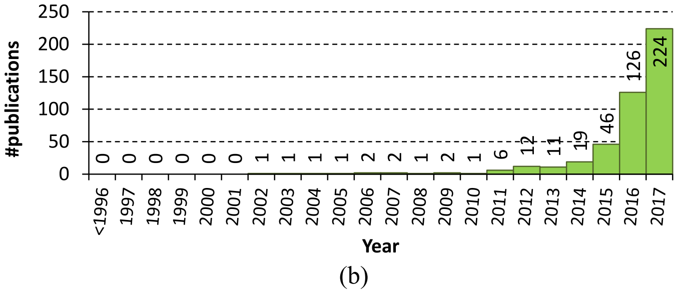



Fig. 8. Number of publications over past years. (a) Number of publications on ML/IoT. (b) Number of publications on ML and IoT.

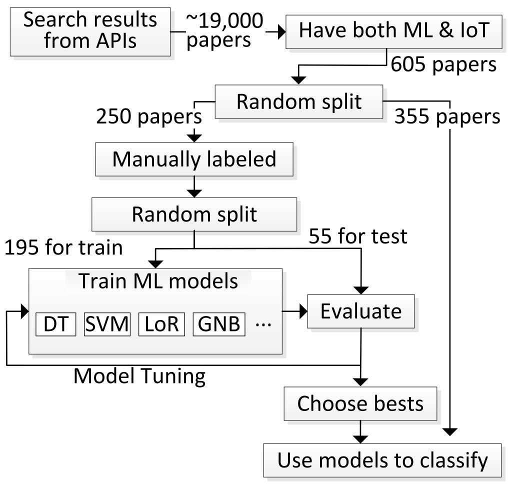


Fig. 9. Using an ML approach to classify the published papers about ML in IoT.

label include but not limited to "security," "malware," "side channel," "attack," "encryption," etc.

B. Feature Selection and Classification

We extract the following features for our classification model.

1) The number of occurrence of each keyword in the title and sum of them.
2) The number of occurrence of each keyword in the abstract and sum of them.
3) Whether the author-provided terms are available.
4) The number of occurrence of each keyword among author-provided terms and sum of them (if they are provided by API).

Fig. 9 shows the flow of our systematic approach to analyze the published papers about ML in IoT. For each label, we train binary classifiers using different models, including LoR, DT, KNN, LDA, Gaussian NB, and SVM. Fig. 10 shows the accuracy of these classifiers for each label. DT outperforms the other classifiers with an average accuracy of 96.7%.

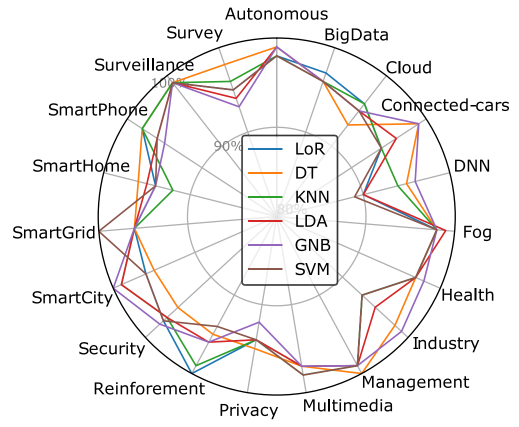


Fig. 10. Accuracy of different classifiers for each label.

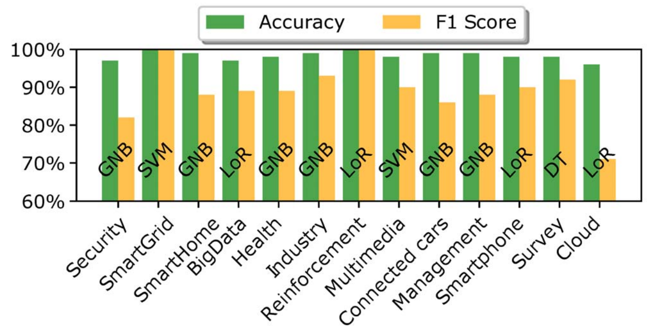


Fig. 11. Classification model which performs best for each topic in terms of accuracy and F1 score.

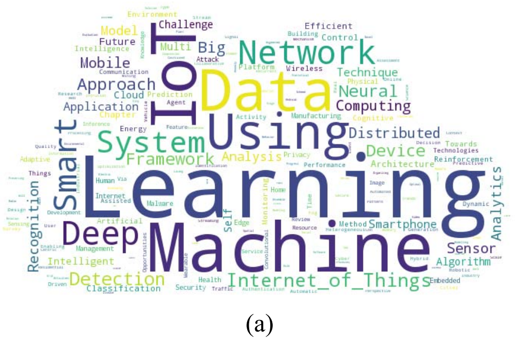
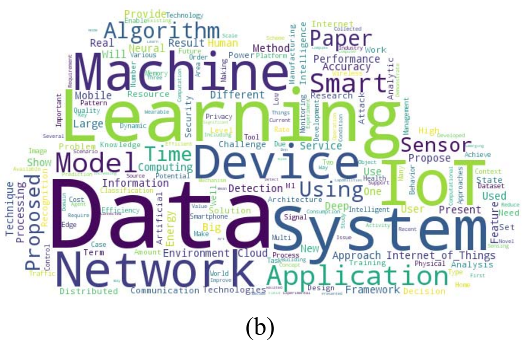



Fig. 12. Frequent keywords in the (a) title and (b) abstract of published papers on ML and IoT.

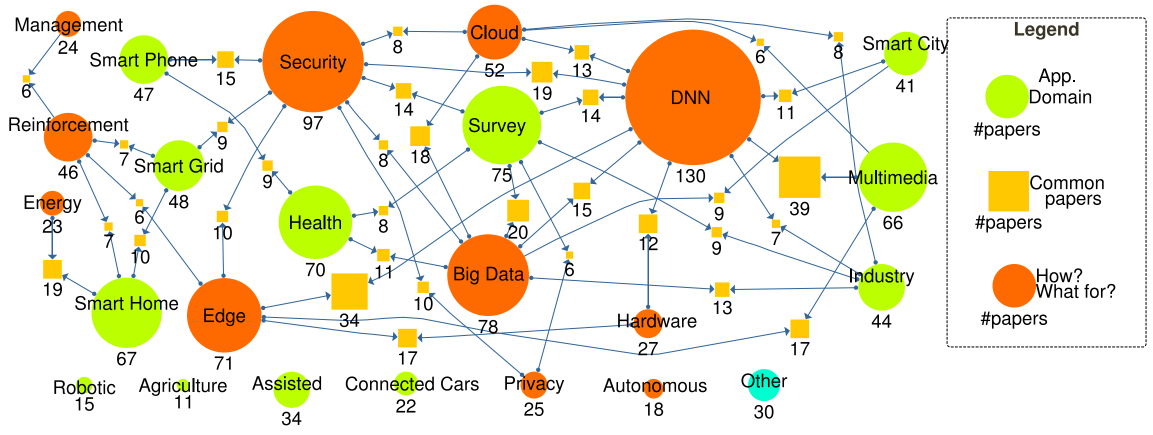


Fig. 13. Accuracy of different classifiers for each label.

One of the widely used performance metrics for the classifiers is F1 score which combines the precision and recall in one metric. Fig. 11 reports the classification model which performs the best for each class in terms of F1 score and accuracy. For the classes like *smart grid* and *reinforcement*, the F1 score and accuracy are 100%. This is due to distinctive and specific terms and keywords in these two topics. On the other hand, the *cloud* class has the lowest F1 score (=71%) which is because most of the publications use the related keywords to "cloud computing" in their abstract. The average accuracy and F1 score for all classes is 98% and 89%, respectively.

C. Analysis

We extract the following classes and labels for the queries: Security, Privacy, BigData, Cloud, Smart-Building, Smart-Home, Assisted Living (also include gesture recognition, brain–machine-interface, and activity recognition), Smart-Grid, Smart-Industry, Agriculture, Health and Wellbeing, Smart City (traffic, pollution, and navigation), Smart-Phone, Smart Cars (also includes driver monitoring), surveillance, multimedia (image, video, and speech), Reinforcement, DNN, hardware (proposed hardware architectures and designs), energy (for buildings and homes), power (for IoT devices and cloud servers), Edge, Survey (overview, survey and perspective papers, tutorials, and keynotes), robotic (drones, assistant robots, and unmanned vehicles), Management (scheduling, resource allocation, and operation mode selection), Autonomous (self-learning, self-organizing, and self-adapting systems), and "other" (that did not fall into any other category).

Fig. 12 shows the word cloud of the keywords in the title and abstract of all the 605 retrieved papers. This figure shows which keywords have been used more frequent in the titles [Fig. 12(a)] and abstract [Fig. 12(b)] of these papers.

Fig. 13 shows the number of papers in each class. To show the overlapping classes, a shared square node is used which is connected to both classes.

Fig. 14 shows the trend of publication number in ML usage for some selected IoT topics. We choose the topics whose trend present interesting or unexpected information. The highest boost in the number of publications in recent years belongs to the usage of DNNs in IoT. Moreover, the security, healthcare,

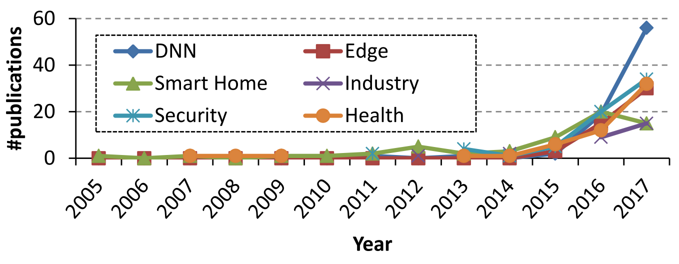


Fig. 14. Publication trend of ML usage in different IoT applications domains.

and edge computing has also received increasing attention in past few years. On the contrary, the number of publications on the usage of ML in smart home has decreased after moderate growth. Note that the data is for the papers published till 2018. However, the presented details in this section allow other researchers to reproduce the analysis with updated publications at a later time.

VI. Conclusion

With the massive increase in the collected data from IoT devices, the traditional cloud computing paradigm is no longer fulfilling the diverse requirements of IoT applications and data inference. ML is a key tool for data inference and decision making in IoT which can be integrated at different processing layers.

This paper reviews the usage of ML in different IoT application domains both for data processing and management tasks. Moreover, we systematically investigate all the publications on the use of ML in IoT. We used classification techniques to determine the topics of each publication based on its title, abstract, etc. The findings show that the number of publications on "DNN," security, "edge computing," and "healthcare" has boosted in recent years, while "smart home" shows a slight decrease. In addition, we highlight the main research directions toward efficient ML at the edge of IoT.

References

[1] F. Samie, L. Bauer, and J. Henkel, "IoT technologies for embedded computing: A survey," in *Proc. CODES+ISSS*, 2016, pp. 1–10.
[2] W. Yu *et al.*, "A survey on the edge computing for the Internet of Things," *IEEE Access*, vol. 6, pp. 6900–6919, 2018.

[3] F. Samie *et al.*, "Computation offloading and resource allocation for low-power IoT edge devices," in *Proc. IEEE WF-IoT*, 2016, pp. 7–12.
[4] C. Raphael. (2016). *Why Edge Computing Is Crucial for the IoT*. Accessed: Jul. 6, 2016. [Online]. Available: http://www.rtinsights.com/why-edge-computing-and-analytics-is-crucial-for-the-iot/
[5] F. Samie, "Resource management for edge computing in Internet of Things (IoT)," Ph.D. dissertation, Dept. Comput. Sci., Karlsruhe Inst. Technol., Karlsruhe, Germany, 2018. [Online]. Available: https://publikationen.bibliothek.kit.edu/1000081031/7142406
[6] F. Samie *et al.*, "Oops: Optimizing operation-mode selection for IoT edge devices," *ACM Trans. Internet Technol.*, to be published.
[7] O. B. Sezer, E. Dogdu, and A. M. Ozbayoglu, "Context-aware computing, learning, and big data in Internet of Things: A survey," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 1–27, Feb. 2018.
[8] U. S. Shanthamallu, A. Spanias, C. Tepedelenlioglu, and M. Stanley, "A brief survey of machine learning methods and their sensor and IoT applications," in *Proc. Int. Conf. Inf. Intell. Syst. Appl. (IISA)*, 2017, pp. 1–8.
[9] M. Shafique *et al.*, "An overview of next-generation architectures for machine learning: Roadmap, opportunities and challenges in the IoT era," in *Proc. DATE*, 2018, pp. 827–832.
[10] M. Shafique *et al.*, "Adaptive and energy-efficient architectures for machine learning: Challenges, opportunities, and research roadmap," in *Proc. Annu. Symp. VLSI (ISVLSI)*, 2017, pp. 627–632.
[11] M. Mohammadi, A. Al-Fuqaha, S. Sorour, and M. Guizani, "Deep learning for IoT big data and streaming analytics: A survey," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 2923–2960, 4th Quart., 2018.
[12] M. Verhelst and B. Moons, "Embedded deep neural network processing: Algorithmic and processor techniques bring deep learning to IoT and edge devices," *IEEE Solid-State Circuits Mag.*, vol. 9, no. 4, pp. 55–65, Nov. 2017.
[13] V. Sze, Y.-H. Chen, J. Einer, A. Suleiman, and Z. Zhang, "Hardware for machine learning: Challenges and opportunities," in *Proc. Custom Integr. Circuits Conf. (CICC)*, 2017, pp. 1–8.
[14] V. Maini and S. Sabri. (2017). *Machine Learning for Humans*. Accessed: Sep. 6, 2018. [Online]. Available: https://medium.com/machine-learning-for-humans
[15] R. S. Sutton *et al.*, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 1998.
[16] J. Henkel, S. Pagani, H. Amrouch, L. Bauer, and F. Samie, "Ultra-low power and dependability for IoT devices (invited paper for IoT technologies)," in *Proc. DATE*, 2017, pp. 954–959.
[17] N. D. Lane, S. Bhattacharya, P. Georgiev, C. Forlivesi, and F. Kawsar, "An early resource characterization of deep learning on wearables, smartphones and Internet-of-Things devices," in *Proc. Int. Workshop Internet Things Toward Appl.*, 2015, pp. 7–12.
[18] D. Azariadi, V. Tsoutsouras, S. Xydis, and D. Soudris, "ECG signal analysis and arrhythmia detection on IoT wearable medical devices," in *Proc. Int. Conf. Mod. Circuits Syst. Technol.*, 2016, pp. 1–4.
[19] M. Hadjem, O. Salem, and F. Naït-Abdesselam, "An ECG monitoring system for prediction of cardiac anomalies using WBAN," in *Proc. Int. Conf. e-Health Netw. Appl. Services*, 2014, pp. 441–446.
[20] N. Keshan, P. V. Parimi, and I. Bichindaritz, "Machine learning for stress detection from ECG signals in automobile drivers," in *Proc. Int. Conf. Big Data (Big Data)*, 2015.
[21] P. S. Pandey, "Machine learning and IoT for prediction and detection of stress," in *Proc. Int. Conf. Comput. Sci. Appl. (ICCSA)*, 2017, pp. 1–5.
[22] F. Samie, S. Paul, L. Bauer, and J. Henkel, "Highly efficient and accurate seizure prediction on constrained IoT systems," in *Proc. DATE*, 2018, pp. 955–960.
[23] G. Li, B.-L. Lee, and W.-Y. Chung, "Smartwatch-based wearable EEG system for driver drowsiness detection," *IEEE Sensors J.*, vol. 5, no. 12, pp. 7169–7180, Dec. 2015.
[24] D. Borthakur, H. Dubey, N. Constant, L. Mahler, and K. Mankodiya, "Smart fog: Fog computing framework for unsupervised clustering analytics in wearable Internet of Things," in *Proc. IEEE Glob. Conf. Signal Inf. Process. (GlobalSIP)*, 2017, pp. 472–476.
[25] W. Li, T. Logenthiran, V.-T. Phan, and W. L. Woo, "Implemented IoT-based self-learning home management system (SHMS) for Singapore," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 2212–2219, Jun. 2018.
[26] U. A. B. U. A. Bakar, H. Ghayvat, S. F. Hasanm, and S. C. Mukhopadhyay, "Activity and anomaly detection in smart home: A survey," in *Next Generation Sensors and Systems*. Cham, Switzerland: Springer, 2016, pp. 191–220.
[27] X. Fafoutis *et al.*, "Extending the battery lifetime of wearable sensors with embedded machine learning," in *Proc. IEEE 4th World Forum Internet Things (WF-IoT)*, 2018, pp. 269–274.
[28] S. Idowu, S. Saguna, C. Åhlund, and O. Schelén, "Forecasting heat load for smart district heating systems: A machine learning approach," in *Proc. Int. Conf. Smart Grid Commun. (SmartGridComm)*, 2014.
[29] Y. Yue *et al.*, "Deep recursive super resolution network with Laplacian pyramid for better agricultural pest surveillance and detection," *Comput. Electron. Agricult.*, vol. 150, pp. 26–32, Jul. 2018.
[30] E. Mwebaze and G. Owomugisha, "Machine learning for plant disease incidence and severity measurements from leaf images," in *Proc. Int. Conf. Mach. Learn. Appl. (ICMLA)*, 2016, pp. 158–163.
[31] S. S. Patil and S. A. Thorat, "Early detection of grapes diseases using machine learning and IoT," in *Proc. Int. Conf. Cogn. Comput. Inf. Process. (CCIP)*, 2016, pp. 1–5.
[32] S. Yahata *et al.*, "A hybrid machine learning approach to automatic plant phenotyping for smart agriculture," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, 2017, pp. 1787–1793.
[33] A. Mangal and N. Kumar, "Using big data to enhance the Bosch production line performance: A Kaggle challenge," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, 2016, pp. 2029–2035.
[34] R. Carbonneau, K. Laframboise, and R. Vahidov, "Application of machine learning techniques for supply chain demand forecasting," *Eur. J. Oper. Res.*, vol. 184, no. 3, pp. 1140–1154, 2008.
[35] G. A. Susto, A. Schirru, S. Pampuri, S. McLoone, and A. Beghi, "Machine learning for predictive maintenance: A multiple classifier approach," *IEEE Trans. Ind. Informat.*, vol. 11, no. 3, pp. 812–820, Jun. 2015.
[36] Z. Wu, H. Luo, Y. Yang, X. Zhu, and X. Qiu, "An unsupervised degradation estimation framework for diagnostics and prognostics in cyber-physical system," in *Proc. IEEE 4th World Forum Internet Things (WF-IoT)*, 2018, pp. 784–789.
[37] Q. Wei, D. Liu, and G. Shi, "A novel dual iterative *Q*-learning method for optimal battery management in smart residential environments," *IEEE Trans. Ind. Electron.*, vol. 62, no. 4, pp. 2509–2518, Apr. 2015.
[38] D. O'Neill, M. Levorato, A. Goldsmith, and U. Mitra, "Residential demand response using reinforcement learning," in *Proc. Int. Conf. Smart Grid Commun. (SmartGridComm)*, 2010, pp. 409–414.
[39] R. Thapa, L. Jiao, B. J. Oommen, and A. Yazidi, "A learning automaton-based scheme for scheduling domestic shiftable loads in smart grids," *IEEE Access*, vol. 6, pp. 5348–5361, 2018.
[40] N. Sharma, P. Sharma, D. Irwin, and P. Shenoy, "Predicting solar generation from weather forecasts using machine learning," in *Proc. Int. Conf. Smart Grid Commun. (SmartGridComm)*, 2011, pp. 528–533.
[41] K. Chen, J. Jiang, F. Zheng, and K. Chen, "A novel data-driven approach for residential electricity consumption prediction based on ensemble learning," *Energy*, vol. 150, pp. 49–60, May 2018.
[42] T. Vantuch *et al.*, "Machine learning based electric load forecasting for short and long-term period," in *Proc. IEEE 4th World Forum Internet Things (WF-IoT)*, 2018, pp. 511–516.
[43] V. Ford, A. Siraj, and W. Eberle, "Smart grid energy fraud detection using artificial neural networks," in *Proc. Comput. Intell. Appl. Smart Grid (CIASG)*, 2014, pp. 1–6.
[44] S.-Y. Lee, S.-R. Wi, E. Seo, J.-K. Jung, and T.-M. Chung, "ProFiOt: Abnormal behavior profiling (ABP) of IoT devices based on a machine learning approach," in *Proc. ITNAC*, 2017, pp. 1–6.
[45] S. Zhao, W. Li, T. Zia, and A. Y. Zomaya, "A dimension reduction model and classifier for anomaly-based intrusion detection in Internet of Things," in *Proc. DASC/PiCom/DataCom/CyberSciTech*, 2017, pp. 836–843.
[46] J. Wei and G. J. Mendis, "A deep learning-based cyber-physical strategy to mitigate false data injection attack in smart grids," in *Proc. Cyber Phys. Security Resilience Smart Grids (CPSR-SG)*, 2016, pp. 1–6.
[47] S. Sharmeen, S. Huda, J. H. Abawajy, W. N. Ismail, and M. M. Hassan, "Malware threats and detection for industrial mobile-IoT networks," *IEEE Access*, vol. 6, pp. 15941–15957, 2018.
[48] P. Sun, J. Li, M. Z. A. Bhuiyan, L. Wang, and B. Li, "Modeling and clustering attacker activities in IoT through machine learning techniques," *Inf. Sci.*, vol. 479, pp. 456–471, Apr. 2019.
[49] H.-J. Jeong, H.-J. Lee, and S.-M. Moon, "Work-in-progress: Cloud-based machine learning for IoT devices with better privacy," in *Proc. EMSOFT*, 2017, pp. 1–2.
[50] J. Chauhan *et al.*, "Breathing-based authentication on resource-constrained IoT devices using recurrent neural networks," *Computer*, vol. 51, no. 5, pp. 60–67, 2018.
[51] G. M. Dias, M. Nurchis, and B. Bellalta, "Adapting sampling interval of sensor networks using on-line reinforcement learning," in *Proc. WF-IoT*, 2016, pp. 460–465.

[52] M. S. Golanbari, S. Kiamehr, F. Oboril, A. Gebregiorgis, and M. B. Tahoori, "Post-fabrication calibration of near-threshold circuits for energy efficiency," in *Proc. ISQED*, 2017, pp. 385–390.

[53] M. Chafii, F. Bader, and J. Palicot, "Enhancing coverage in narrow band-IoT using machine learning," in *Proc. IEEE (WCNC)*, 2018, pp. 1–6.

[54] F. A. Kraemer, D. Ammar, A. E. Braten, N. Tamkittikhun, and D. Palma, "Solar energy prediction for constrained IoT nodes based on public weather forecasts," in *Proc. Int. Conf. Internet Things*, 2017, p. 2.

[55] W. Cui, Y. Kim, and T. S. Rosing, "Cross-platform machine learning characterization for task allocation in IoT ecosystems," in *Proc. Comput. Commun. Workshop Conf. (CCWC)*, 2017, pp. 1–7.

[56] X. He *et al.*, "Green resource allocation based on deep reinforcement learning in content-centric IoT," *IEEE Trans. Emerg. Topics Comput.*, to be published.

[57] A. Iranfar, W. A. Simon, M. Zapater, and D. Atienza, "A machine learning-based strategy for efficient resource management of video encoding on heterogeneous MPSoCs," in *Proc. ISCAS*, 2018, pp. 1–5.

[58] A. I. Orhean, F. Pop, and I. Raicu, "New scheduling approach using reinforcement learning for heterogeneous distributed systems," *J. Parallel Distrib. Comput.*, vol. 117, pp. 292–302, Jul. 2018.

[59] M. Min *et al.*, "Learning-based privacy-aware offloading for healthcare IoT with energy harvesting," *IEEE Internet Things J.*, to be published.

[60] B. Demirel, A. Ramaswamy, D. E. Quevedo, and H. Karl, "DeepCAS: A deep reinforcement learning algorithm for control-aware scheduling," *IEEE Control Syst. Lett.*, vol. 2, no. 4, pp. 737–742, Oct. 2018.

[61] E. Balevi and R. D. Gitlin, "Unsupervised machine learning in 5G networks for low latency communications," in *Proc. Int. Perform. Comput. Commun. Conf. (IPCCC)*, 2017, pp. 1–2.

[62] A. E. Eshratifar and M. Pedram, "Energy and performance efficient computation offloading for deep neural networks in a mobile cloud computing environment," in *Proc. Great Lakes Symp. VLSI*, 2018, pp. 111–116.

[63] E. A. Castillo and A. Ahmadinia, "Distributed deep convolutional neural network for smart camera image recognition," in *Proc. Int. Conf. Distrib. Smart Cameras*, 2017, pp. 169–173.

[64] H. Li, K. Ota, and M. Dong, "Learning IoT in edge: Deep learning for the Internet of Things with edge computing," *IEEE Netw.*, vol. 32, no. 1, pp. 96–101, Jan./Feb. 2018.

[65] M. I. Khan, M. M. Alam, Y. Le Moullec, and E. Yaacoub, "Cooperative reinforcement learning for adaptive power allocation in device-to-device communication," in *Proc. IEEE 4th World Forum Internet Things (WF-IoT)*, 2018, pp. 476–481.

[66] M. Mohammadi, A. Al-Fuqaha, M. Guizani, and J.-S. Oh, "Semisupervised deep reinforcement learning in support of IoT and smart city services," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 624–635, Apr. 2018.

[67] X. Zhang *et al.*, "Machine learning on FPGAs to face the IoT revolution," in *Proc. ICCAD*, 2017, pp. 819–826.

[68] A. A. Sarangdhar and V. R. Pawar, "Machine learning regression technique for cotton leaf disease detection and controlling using IoT," in *Proc. Int. Conf. Electron. Commun. Aerosp. Technol. (ICECA)*, 2017, pp. 449–454.

[69] A. Marinescu, I. Dusparic, and S. Clarke, "Prediction-based multi-agent reinforcement learning in inherently non-stationary environments," *ACM Trans. Auton. Adapt. Syst.*, vol. 12, no. 2, p. 9, 2017.

[70] M. Gao, Q. Wang, M. T. Arafin, Y. Lyu, and G. Qu, "Approximate computing for low power and security in the Internet of Things," *Computer*, vol. 50, no. 6, pp. 27–34, 2017.

[71] Q. Zhang, T. Wang, Y. Tian, F. Yuan, and Q. Xu, "ApproxANN: An approximate computing framework for artificial neural network," in *Proc. DATE*, 2015, pp. 701–706.

[72] Z. Tang *et al.*, "Energy-efficient transmission scheduling in mobile phones using machine learning and participatory sensing," *IEEE Trans. Veh. Technol.*, vol. 64, no. 7, pp. 3167–3176, Jul. 2015.

[73] Y. Liu, L. Liu, and W.-P. Chen, "Intelligent traffic light control using distributed multi-agent Q learning," in *Proc. Int. Conf. Intell. Transp. Syst. (ITSC)*, 2017, pp. 1–8.

[74] J. Teittinen *et al.*, "A 5.3 pJ/op approximate TTA VLIW tailored for machine learning," *Microelectron. J.*, vol. 61, pp. 106–113, Mar. 2017.

[75] S. Venkataramani, K. Roy, and A. Raghunathan, "Efficient embedded learning for IoT devices," in *Proc. Asia South Pac. Design Autom. Conf. (ASP-DAC)*, 2016, pp. 308–311.

[76] Y. Chen, T. Chen, Z. Xu, N. Sun, and O. Temam, "DianNao family: Energy-efficient hardware accelerators for machine learning," *Commun. ACM*, vol. 59, no. 11, pp. 105–112, 2016.

[77] V. K. Chippa *et al.*, "Scalable effort hardware design: Exploiting algorithmic resilience for energy efficiency," in *Proc. DAC*, 2010, pp. 555–560.

[78] S. Venkataramani, A. Ranjan, K. Roy, and A. Raghunathan, "AxNN: Energy-efficient neuromorphic systems using approximate computing," in *Proc. ISLPED*, 2014, pp. 27–32.

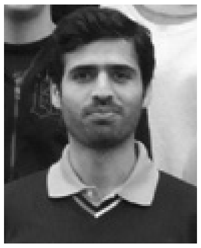
[79] Y. Huan, Y. Qin, Y. You, L. Zheng, and Z. Zou, "A multiplication reduction technique with near-zero approximation for embedded learning in IoT devices," in *Proc. Int. Syst. Chip Conf. (SOCC)*, 2016, pp. 102–107.

[80] T. Sheng *et al.*, "A quantization-friendly separable convolution for MobileNets," in *Proc. Workshop Energy Efficient Mach. Learn. Cogn. Comput. Embedded Appl. (EMC2)*, 2018, pp. 1–5.

[81] S. Venkataramani, A. Raghunathan, J. Liu, and M. Shoaib, "Scalable-effort classifiers for energy-efficient machine learning," in *Proc. Design Autom. Conf. (DAC)*, 2015, p. 67.

[82] S. Wang *et al.*, "When edge meets learning: Adaptive control for resource-constrained distributed machine learning," in *Proc. IEEE INFOCOM*, 2018, pp. 63–71.

[83] K. Portelli and C. Anagnostopoulos, "Leveraging edge computing through collaborative machine learning," in *Proc. Int. Conf. Future Internet Things Cloud Workshops (FiCloudW)*, 2017, pp. 164–169.



Farzad Samie received the B.Sc. and M.Sc. degrees in computer engineering from the Sharif University of Technology, Tehran, Iran, in 2008 and 2010, respectively, and the Ph.D. (Dr.-Ing.) degree in computer science from the Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany, in 2018.

He is currently a Post-Doctoral Researcher and a Lecturer with the Chair for Embedded Systems, KIT. His current research interests include Internet-of-Things, edge computing, and healthcare applications.

Lars Bauer received the M.Sc. and Ph.D. degrees in computer science from the University of Karlsruhe, Karlsruhe, Germany, in 2004 and 2009, respectively.

He is currently a Research Group Leader and a Lecturer with the Chair for Embedded Systems, Karlsruhe Institute of Technology, Karlsruhe. His current research interest includes architectures and management for adaptive multi-/many-core systems.

Dr. Bauer was a recipient of two Dissertation Awards (EDAA and FZI), two Best Paper Awards (AHS11 and DATE08), and several nominations.

Jörg Henkel (M'95–SM'01–F'15) received the Diploma and the Ph.D. (*summa cum laude*) degrees in electrical engineering from the Technical University of Braunschweig, Braunschweig, Germany.

He is the Chair Professor for embedded systems with the Karlsruhe Institute of Technology, Karlsruhe, Germany. His current research interest includes co-design for embedded hardware/software systems with respect to power, thermal, and reliability aspects.

Prof. Henkel served as the Editor-in-Chief for the *ACM Transactions on Embedded Computing Systems* and is currently the Editor-in-Chief of the *IEEE Design & Test Magazine*. He has led several conferences as the General Chair, including ICCAD and ESWeek, and serves as the Steering Committee Chair/member for leading conferences and journals for embedded and cyber-physical systems. He coordinates the DFG program SPP 1500 "Dependable Embedded Systems" and is a Site Coordinator of the DFG TR89 collaborative research center on "Invasive Computing."