Embedded Intelligence in the Internet-of-Things

Robert P. Dick
University of Michigan, Ann Arbor

Marilyn Wolf
University of Nebraska–Lincoln

Li Shang
University of Colorado Boulder

Shao-Wen Yang
Amazon.com, Inc.

Editor's note:
With widespread use of IoT devices in our daily life, it is imperative to incorporate analysis and decision-making capabilities in these devices. This article describes the application and technology trends leading to embedded intelligence in the IoT.
—Partha Pratim Pande, Washington State University

The Internet-of-Things (IoT) is a distributed system of embedded computers that sense, analyze, and possibly actuate in the physical world. IoT systems frequently use wireless communication. Users face constraints on price and energy consumption that favor efficient designs. Embedded intelligence is the capacity for in-system analysis and decision-making. It stands in contrast to transferring raw data for analysis in the cloud. It can benefit a wide range of IoT applications, including computer vision, transportation, industrial automation, medical devices, wearables, and agriculture.

This article describes the applications and technology trends leading to embedded intelligence in the IoT, indicates challenges faced by designers and maintainers of IoT systems, explains research concepts and technologies being used to make these systems practical, and points out the implications for several application domains.

There are three main factors leading to, or necessitating, embedded intelligence in the IoT, foremost among which is the frequent requirement for wireless communication in these broadly distributed systems. Wireless communication typically imposes orders of magnitude higher energy and time costs per bit than computation; advances in the design of efficient machine-learning algorithms and hardware have enabled embedded intelligence; and automated methods for feature definition, selection, and training have removed many constraints on using it, but the difficulty in accessing training data in the fragmented IoT market slows progress.

Figure 1 illustrates an example IoT application system architecture. Several physical plants are sensed and acted upon by edge devices, which communicate with each other and/or upstream networking nodes, and ultimately to hubs that are often connected to the Internet. Thereafter, data may flow to private or cloud servers, which then transmit decisions back to edge devices. Without very sophisticated algorithm, software, and hardware designs to minimize the amount of data flowing through the network (particularly upstream), energy consumption and throughput constraints will undermine many applications. For an efficient architecture, embedded intelligence in the edge devices and local networks is expected to solve many data-intensive inference problems, with the cloud being used for planning based on higher-order analysis products transmitted at relatively low data rates. It is sometimes possible to reduce energy comsumption, e.g., by increasing interference latency. However, high latency is unacceptable for many IoT applications.

Digital Object Identifier 10.1109/MDAT.2019.2957352
Date of publication: 3 December 2019; date of current version: 6 February 2020.

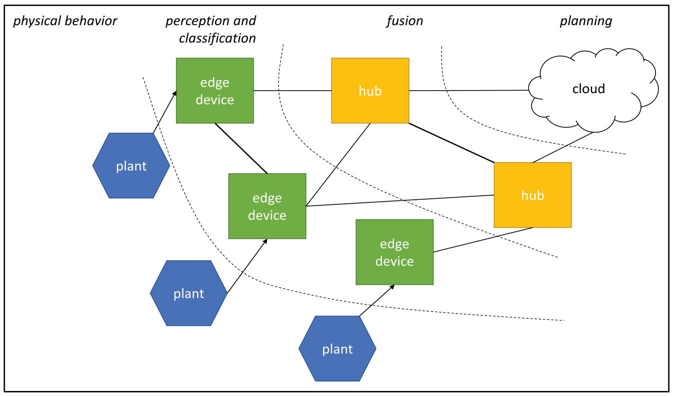

Figure 1. Overview of IoT system architecture.

Wireless communication trends and IoT implications

Were it not for the high energy consumption and latency of wireless communication, it is likely that IoT systems would be intelligent, but it is not clear that the intelligence would be embedded instead of hosted in remote servers in the cloud. IoT system architectures are very heavily influenced by the properties of available wireless communication technologies. Table 1 indicates typical ranges, data rates, and power consumption of several communication technologies. Until recently, few wireless communication technologies were appropriate for large-scale IoT deployments, especially those with battery-powered edge devices. As a result, most IoT system designers were forced to select technologies similar to the first four mentioned in Table 1. Each either requires high power consumption that complicates use in battery-powered applications or has an inadequate range for large-scale IoT applications.

Recently, a new class of wireless communication technology, called low-power wide-area networks (LPWANs) has experienced rapid development and increasing use. The market size was $1.5 billion in 2018 and is expected to grow by 60% per year until 2025 [1]. Several companies have developed LPWAN communication technologies primarily for use in IoT applications [2], with the following goals:

- low enough average power consumption to support ten years of battery lifespan without having batteries dominating the size;
- long range, typically supporting at least a 10 km direct transmission range; and
- using protocols that simple enough to support commodity embedded microcontrollers.

To achieve these goals, compromises were necessary, the most common of which was tightly constraining data transfer rates. Other compromises include restriction to very simple network structures such as star topology or star-of-stars topology, each of which requires gateways to be placed within a single hop of all edge devices.

The recent explosion in IoT communication technologies (e.g., LoRaWAN [3]–[5], Weightless, narrow band (NB)-IoT [6]–[8], and SigFox [9], [10]) is enabling long-range deployment of battery-powered IoT systems. However, each faces tight constraints on data transfer rates and, in many cases, maximum data transferred per day. As a result, despite very low power consumption during transfer (e.g., 10 mW for LoRaWAN), the energy cost per bit is high. For example, the energy cost for LoRaWAN is 500 nJ/b, which compares unfavorably with WiFi's 125 nJ/b.

Taking representative examples from short-range and long-range non-LPWAN and LPWAN communication technologies, the energy cost per bit of data transfer exceeds the energy cost per bit of computation by orders of magnitude. To be specific, when we define energy cost per bit computed as the typical cost per arithmetic instruction on a processor representative of a near-future IoT microcontroller (e.g., an ARM Cortex A57), divided by the word size of the processor, we find that the non-LPWAN communication technologies allow communication to computation energy ratios ranging from 670× to 33,000× and the LPWAN ratio ranges from 27,000× to 33,000×. In short, for a fixed amount of energy, it is possible to carry out at least two orders of magnitude more bit-computations than bit-communications, with four orders of magnitude common for long-range LPWAN communication technologies. This isolating influence of energy-intensive, or slow,

Table 1. Comparison of wireless communication technologies.

Technology	LPWAN	Power (mW)	Range (m)	Typical rate (kb/s)
4G	No	1,000	70,000	10,000
5G	No	1,000	40,000	100,000
WiFi / 802.11x	No	250	140	20,000
Zigbee / 802.15.4	No	1–100	10–1,500	20–200
LoRaWAN	Yes	10	15,000	20
NB-IoT	Yes	100	15,000	250

communication drives the requirement for embedded intelligence in the IoT including edge-node signal processing, feature extraction, compression, and machine learning. It is a root cause of the need for embedded intelligence.

Remote servers and the cloud will play roles in many IoT systems. In some cases, where the number of computations required for analysis of a single upstream bit number in the thousands to tens of thousands, it may be more energy efficient to transmit raw data to the cloud than to carry out inference locally. However, in these cases it will be more efficient, still, to do some degree of data analysis and reduction on the edge device, and in the network. For most IoT applications, naïve solutions in which edge nodes wirelessly transmit all raw data for analysis in the cloud will be impractical for all but small-scale and low data rate applications. IoT system designers will be forced to use sophisticated compression, feature extraction, and local inference techniques to minimize communication volume and dependence on wireless communication. Communication with the cloud will play an important role in aggregating information from diverse environments and helping to globally coordinate IoT systems, but it is an expensive resource to be economized or rationed.

To illustrate the impossibility of avoiding embedding intelligence, consider the amount of data generated by a single autonomous automobile in one day: 4 TB. Making just 1000 automobiles intelligent (0.001% of the 1.3 billion in the world) requires data to be processed at the same rate as Facebook [11]. Without embedded intelligence, IoT systems cannot scale.

Embedded intelligence has advantages beyond reduction in communication. Wireless communication is not generally reliable. Data can be lost, and retransmission is not a robust solution: changes in wireless communication environments caused by rain, moving doors, and swaying trees isolate edge devices for long periods of time [12], [13]. Even when error detection and retransmission are feasible, they introduce latency that can produce deadline violations. Embedded intelligence enables local decision-making, thereby removing error-prone and unbounded-latency wireless communication from the sense–analyze–decide–actuate control loop. Even systems that rely on the cloud for most decisions will frequently support local decision-making capability to enable continued operation during network outages.

Embedded intelligence can also have privacy benefits relative to the alternative of transmitting more raw data to the cloud. Layers of analysis typically have the effect of eliminating information superfluous to the application goal. For example, a security system focused on determining whether a stranger is in a house can eliminate information about the detailed actions of residents. In general, the more analysis done on edge devices, the less superfluous and potentially privacy-adverse information needs to be shared with the cloud [14].

Machine learning in the IoT

Although many existing IoT systems rely on embedded intelligence, technical barriers have slowed its adoption and remain in many applications. To understand the cause, consider the history of a highly successful branch of machine learning: deep learning. Many of the fundamental concepts enabling deep learning were developed and well understood in the 1990s. For example, multilayer perceptrons were first described in 1969 [15], and convolutional neural networks (CNNs) were invented in 1980 [16]. However, successful application to practical problems of widespread importance remained rare. Almost 20 years later, LeCun et al. [17] described a moderately deep CNN capable of classifying handwritten digits. In the next decade, two necessary conditions were satisfied: 1) access to data sets large and diverse enough to enable effective training [18] and 2) computers fast enough to handle problems of substantial size and complexity. It was not until the 2010s that CNNs were often able to match human accuracy [19]–[21]. Improving efficiency, especially energy efficiency, took a back seat to improving accuracy; although machine-learning systems had finally achieved useful accuracy, many of the most accurate machine-learning techniques (e.g., deep-learning techniques) were too computationally demanding to use on performance- and energy-constrained systems. Only recently did that begin to change.

The 2010s have seen a flurry of work reimagining the design of machine-learning algorithms and hardware, and this activity continues. From algorithmic techniques that eliminate wasted computation and state, to hardware architectures tuned to the specific computational needs of deep-learning systems, a revolution is underway in efficient automated analysis and decision-making.

Although these are changes in magnitude, they are not incremental improvements; they allow analysis on edge devices that a few years earlier would have required a datacenter, and a few years before that would have been impossible for automated systems. The impact of improving efficiency is most strongly felt in IoT applications where it is now becoming practical to use learned analysis and decision-making techniques instead of relying on systems in which parameters and rules are manually selected and designed by application experts.

Feature design and selection

In the general field of machine learning, the process of feature design and selection has been evolving rapidly toward increased automation. However, there are special considerations in IoT applications that may work against this general trend.

Until recently, training and inference on large quantities of raw data have been too computationally intensive for practical use. As a consequence, designers developed machine-learning algorithms to run on "features," e.g., reduced, highly relevant vectors produced through computation on sparser raw data, generally using signal-processing algorithms designed by domain experts, e.g., Kanade–Lucas–Tomasi (KLT) features [22], [23] and the scale-invariant feature transform (SIFT) [24] for computer vision. Feature vectors are generally of lower dimensionality than raw data and are encoded with fewer total bits, thus reducing the computational burden on later analysis stages, which generally have computational complexities scaling at least linearly in the input data size. This approach also has the advantage of using the knowledge already encoded in the memories of experts who have been exposed to large amounts of data and have internally formed models for the processes generating the data. Although it may be inferior when good, large data sets are available, it may be superior when data are limited.

CNNs [20], [25]–[28] provide an example of the trend toward integrating feature extraction into the automated learning process. In this case, convolutional layers learn filter matrices that are applied to raw data to produce convolved feature maps, which are used in latter layers to accomplish a specialized task, e.g., classification or detection. Instead of defining feature extraction as something distinct from training and inference, it became merely the first few layers of the network, for which training and inference are consistent with other portions of the network. Although it increases the computational cost of training, this process enables higher accuracy and better training results if adequate data are available. Its use is now common and increasing. However, there are several reasons why the shift may not be as rapid in IoT applications as in more general-purpose applications.

Whether a better result is possible using conventional feature design and selection or integrating these functions into machine-learning algorithms that take raw data as input depends on whether there is more relevant information available in training data sets or in heuristics arrived at by experts. For problem domains with broad applications, such as general image classification, large amounts of high-quality data are available [29], [30], working in favor of automatic feature design. However, IoT applications [31] are fragmented into many problem domains. It is also common for only small amounts of data to be available to several small organizations in competition with each other. These conditions complicate data sharing and undermine the development of large, high-quality training data sets. As a consequence, we expect many IoT applications to rely on manual feature design to a greater degree than general-purpose applications, and for this to remain true for each problem domain until large, high-quality data sets are available.

Dependence on labeled data for automated feature design will also encourage use of transfer learning [32], [33] in the IoT and other embedded applications, where large labeled training data sets are available for related applications, but not for the specific application under consideration. The key idea is to permit knowledge of a particular task and/or application domain to be used to improve accuracy on others. One- or few-shot learning [34], [35], which attempts to generalize given very few training samples, will also be important in applications suffering from the "small data" problem, as may metalearning [36], [37].

State of the art and promising directions

This section describes the state of practice and research in several domains relevant to embedded intelligence in the IoT. We take a top-down approach, starting with distributed system architectural trends and ending with application-specific hardware for IoT systems. Finally, we discuss IoT safety and security, which spans many levels of the IoT hierarchy.

Distributed system architectural trends

Operational technology encompasses hardware–software systems for monitoring physical processes. The evolving architecture for information technology systems and the demands on operational technology meet in the IoT. From the information technology perspective, architectures have shifted from single-user mainframe, to time-shared mainframe, to personal computer, and then to networked computing. Today's IoT systems are generally distributed systems, in which subsystems for data management, business logic, and user experience are physically separated. The challenge from the operational technology perspective is to enable these distributed systems to operate in unison during learning, inference, decision making, and actuation. Specifically, the move to client–server computing has been a key technology trend in the 1990s. It enabled thin, and therefore, inexpensive clients. The client–server paradigm allows minimal client hardware with on-demand, often over-the-air (OTA) software updates and information exchange. A thin client may sense and interact with users and the physical world but does little analysis. It primarily communicates with a server.

Cloud-centric design [38]–[40] leveraging client–server computing systems often suffers from poor responsiveness. Some applications cannot tolerate the high or unpredictable latencies of long-range wired, or worse, wireless networks. Time-critical applications require predictability, ruling out architectures relying on unpredictable network states.

A decentralized system [41] is a distributed system that minimizes the dependency on servers, thus enabling responsiveness for a wide range of server and network conditions. This system architecture is better suited to many IoT applications in which edge devices have intermittent connectivity or limited bandwidth. The need for decentralized operation is pushing IoT system architectures toward increased autonomy, i.e., toward embedded intelligence.

Efficient machine learning algorithms for IoT devices

There have been several recent developments that improve the efficiency of machine-learning algorithms enough for use on edge devices. Many are relevant to deep learning and are most naturally described in the context of its multilayer weighted networks. The trends described here are mirrored in other machine-learning application domains.

State-of-the-art neural networks are composed of at least thousands of interconnected neurons. For example, an early network used by LeCun et al. [42] for handwritten digit recognition includes 4635 units, 98,442 connections, and 2578 independent parameters with modern vision CNNs such as ResNet-152 containing 60-million parameters [21]. Parameters must be stored in state variables; techniques that reduce the number of parameters or their sizes improve learning and computation times, memory requirements, and energy consumptions. Some tasks, such as classification, can achieve useful accuracy using low-resolution, highly decimated images, e.g., 224×224 pixels. However, others, such as denoising and super-resolution, require much larger images, e.g., 1920×1080 pixels. Siu et al. [43] demonstrated that storing the activation values for deep CNNs running such applications would require over 500 MB of memory, and if this memory is not on the same die as the processor(s), the performance penalties can be substantial.

CNN optimization methods have been developed for both filter pruning and reduced-precision arithmetic; most methods try to reduce CNN size while maintaining accuracy. Guo et al. [44] combined pruning and splicing to avoid local minima during network pruning. Zhuang et al. [45] reduced the precision of values and arithmetic operations by first quantizing weights, then activations. They used progressive training to incrementally refine bit widths and used a full-precision model to guide the optimization of the reduced-precision network. Albericio et al. [46] designed a deep neural network architecture that avoids superfluous calculation of zero terms during multiplication. Coussy et al. [47] developed a fully binary Gripon–Berrou neural network, thereby transforming integer arithmetic to logical operations. Courbariaux et al. [48] developed a method of training deep neural networks using binary weights during forward and backward propagation that makes intentional use of quantization noise. Yu et al. [49] implemented a fully binary neural network for handwritten digit recognition in hardware. Howard et al. [50] describe a method for improving CNN efficiency by using an unconventional network structure in which parallel input channels are handled by separate, isolated convolutional filters, the outputs of which are subsequently combined using another layer.

In this special issue, Imani et al. [51] describe a hierarchical hyperdimensional computing system that uses an unconventional quantizer to dramatically improve efficiency on resource-constrained embedded systems. Also in this issue, He et al. [52] present a framework to optimize pruning and precision reduction to improve energy efficiency in deep neural networks.

Online learning, potentially by deployed systems, enables machine-learning systems to adapt to unique characteristics of their local environments and use cases, thereby improving accuracy and robustness to deception. However, it faces three challenges. 1) It is difficult to validate information learned in-network because access to appropriate labels and testing data sets is rare: if the appropriate testing data were known, then the network could have been pretrained. 2) Many machine learning techniques are susceptible to catastrophic forgetting, the loss of state learned from earlier training data during online training. 3) The computational cost of training is high for many machine learning systems, e.g., deep neural networks, preventing implementation in low-cost, energy-constrained edge devices. The validation problem is fundamental but can be addressed to some degree by periodically testing accuracy on static labeled data sets.

Researchers have studied the catastrophic forgetting problem. Toneva et al. [53] studied the characteristics of training instances that are likely to be forgotten, and those that are not. They found that instances may be forgotten even when there is only a single classification task and that difficult-to-forget instances may be removed from the training data without reducing accuracy. Work like this is building a foundation that may lead to changes in training strategies to reduce catastrophic forgetting. Loshchilov and Hutter [54] showed that ignoring samples with little influence on loss accelerated training on the MNIST data set by 5×. Developing efficient methods of estimating sample importance is an active area [55]–[57], and information theory-based approaches appear promising.

He et al. [21] address the training challenge for deep networks, which had historically required very large training times. They describe a change in network structure and weight updates that enables efficient (and accurate) training. By adding a shortcut term to the post-activation function output of neural network layers, they propagate the inputs of prior layers into layer outputs. They demonstrated that these residual networks can be efficiently trained with depths an order of magnitude greater than state-of-the-art deep neural networks, achieving improved accuracy. Although the authors demonstrated that the approach worked, the mathematical intuition for why it worked was limited. However, there has been recent progress in this area. He et al. had observed that shortcut lengths of two layers resulted in reduced training error, which translates to reduced training overhead given a fixed error, but that one-layer and three-layer shortcut lengths did not. Li et al. [58] show that networks with depth-two shortcuts have depth-invariant loss function Hessians at the zero initial points. This implies that training difficulty is depth-invariant for such networks. These recent findings demonstrate that the theoretical foundations for efficient learning are still under development.

A comprehensive tutorial and survey by Sze et al. [59] provides additional information on the topic of deep neural network efficiency.

Compressed sensing and inference

Researchers have developed techniques that reduce sensing and computational cost by carefully selecting samples necessary for signal reconstruction (compressed sensing) or decision making. This makes sampling more efficient. It can also be useful in the signal processing and analysis pipeline to eliminate superfluous data, thereby decreasing computational costs that often (superlinearly) depend on data volume.

When using compressed sensing [60], designers exploit knowledge of (potentially stochastic) restrictions on data to enable reconstruction using only sparse, but carefully selected, samples. There are tests for signal compressibility and sparsity. These approaches rely on the fact that many parameters describing real-world signals in commonly used (but naïvely chosen) domains are relatively unimportant in signal reconstruction. Eliminating all but the essential samples reduces data capture, communication, and processing rates. This approach has enabled dramatic reductions in data volume in real-world applications. For example, the amount of data necessary to enable acceptably accurate reconstruction of electrocardiogram signals was found to be only 20% of that implied by the Nyquist frequency [61].

Compressed sensing focuses on selecting a minimal set of samples capable of adequately reconstructing the original signal. Researchers have applied concepts from compressed sensing to the problem of inference, in which the relevant error metric is inference accuracy, not reconstruction accuracy. As is the case in compressed sensing, these approaches exploit sparsity in data. However, they also exploit variation in the relevance of data to the inference problem of interest. Put another way, they exploit the sparsity of input data from the perspective of inference accuracy.

A common idiom in the design of energy-efficient inference algorithms is to use a staged, hierarchical approach. For example, consider a system for detecting vehicles that must, on average, have an accuracy of 97%. One might use a general-purpose vision system composed of a single CNN. This approach may meet the accuracy requirement but at a high energy and computation time cost. Alternatively, one might use a subsystem with lower computational cost to filter out all images that are very unlikely to contain vehicles before running the high-cost CNN on an order of magnitude less data. This subsystem may safely have much lower accuracy than the CNN, provided that few of the errors are false-negatives. This staged analysis concept has been widely used in energy-efficient inference systems and it is related to, but distinct from, compressive inference.

In its purest form, compressive inference eliminates superfluous data from all stages of the inference pipeline. Instead of capturing every bit in every pixel in every timestep, it captures only those bits necessary for accurate inference. This eliminates the energy and time costs of superfluous data processing throughout the inference pipeline. Static approaches to determine which data are important are challenging. However, even static approaches can improve full-system energy efficiency by 5× with little reduction in accuracy for useful applications [57].

Dynamic approaches are also possible and can be observed in energy-efficient biological vision systems such as the human vision system. This system uses a highly spatially heterogeneous, multiround approach to data capture and analysis. Only 1% of the human retina (the area of a thumbnail with an outstretched arm) is capable of spatially dense data capture, but 50% of later stages of the vision pipeline are dedicated to processing the data it captures. A static, context-independent method of determining where to direct this sensor array would be highly inaccurate. The human vision system instead enables dynamic, context-dependent capture, and processing by using a multiround approach to vision in which the sensor array is directed to the areas of the scene most likely to be important to the inference task at hand, based on feedback from later stages of the inference process. This concept, applied to computer vision, improves energy efficiency by 5× with less than 1% loss in accuracy [62].

Online learning

Gradual changes in the operating environments and objectives of IoT applications may require learned models to be updated over time. Conventional batch learning methods are often not well suited to efficient relearning in such applications [63]–[65]. Incremental, online learning based on captured data has the potential to enable efficient adaptation of IoT systems. However, online learning can be energy-intensive and many IoT systems are broadly distributed, making physical maintenance such as battery replacement expensive or intractable [66], [67].

Several IoT communication technologies have been designed with these concerns in mind. For example, NB-IoT, the cellular licensed LPWAN networking technology, enables device battery lifespans of up to 14 years from a single charge. This poses a major challenge for machine-learning algorithms because uploading raw data to the cloud is energy inefficient. Embedded inference has the potential to solve the problem, but inference alone is insufficient. Many-year lifespans imply that operating environments will change substantially during deployment, meaning that pretrained machine learning systems will be too inflexible for many applications. For example, in factory automation, subjects and manufacturing processes change over time in ways that may be difficult to predict before system deployment. In retail, merchandise changes over time. In surveillance, appearances change over time. Online learning can enable machine learning systems to adapt to these changes.

Distributed system architectures and virtualized IoT infrastructure

A critical decision in the architectural design of IoT systems is assigning specific machine learning tasks to particular components in the network architecture. In the case of classification, the location of the machine learning nodes may depend both on the sensors required and on the computational demands of inference algorithms. Training typically requires more computation than classification, which may lead to different functional partitions for training and classification.

Some IoT system use cases are well suited to multitenancy architectures in which several applications share hardware and network connections. Those applications may be administered by different entities. Cherrier et al. [68] proposed a generic object architecture for multitenant IoT systems. They identify several issues: managing multiple control flows that desire to manipulate objects, controlling access rights, and managing conflicting orders issued by different agents.

IoT-oriented machine learning systems can use distributed algorithms to control latency and break dependencies on long-range communication. For example, Kim and Wolf [69] developed a distributed consensus algorithm for multitarget tracking in camera networks. They showed that small neighborhoods for consensus can enable high accuracy. The requirements of multitenancy—performance, power, and privacy—may influence the design of IoT nodes with machine learning capabilities.

Edge-to-cloud workload distribution

Edge-to-cloud architectures allow flexible workload offloading [70]. They contain a wide range of heterogeneous resource types, including diverse sensing, computational, and communication elements, and have varying design considerations. As a result, there is no standard platform, interface, or application programming interface (API) that can be used across heterogeneous devices in the edge-to-cloud paradigm, which may include powerful servers in data centers, gateways in retail stores or households, smartphones, and cameras.

High workload complexity, use case diversity, and hardware heterogeneity make manual workload placement unrealistic, especially in large-scale systems that may contain tens of thousands of edge devices. Edge-to-cloud workload distribution requires precision benchmarking of workloads over a diverse set of hardware and edge-to-cloud task, and communication assignments and schedules of interdependent tasks to interconnected devices [71], [72]. Workload benchmarking is a means to predict resource utilizations and execution times of individual tasks of a job. A service level agreement (SLA) requires precision workload benchmarking, especially over heterogeneous hardware resources, to meet performance requirements.

Existing multiprocessor schedulers do not explicitly take into account communication constraints or scale well [71], [73], [74]. In addition, workload benchmarking [74] is a daunting task for the rather broad hardware choices and their diverse characteristics. In the context of the IoT, both white box and black box benchmarking are required in a multitenant environment. Blackbox benchmarking is particularly challenging because the complexity of a workload is unknown; benchmarking over a comprehensive set of hardware types is required.

Given workload benchmarking data, a scheduler determines workload placement over devices from edge to cloud. Edge-to-cloud scheduling is an optimization problem, taking into account not only computational constraints but also networking constraints. It has the goal of finding an optimal solution for a predetermined objective, e.g., minimal network utilization [71], [75]. The scale of the problem increases significantly when moving from hundreds of servers in a datacenter to tens of thousands of devices over a multitenant infrastructure. Automation is needed to meet the scale of IoT applications—a new area requiring major research investments.

Node architectures

Ultra-low power consumption enables IoT devices to operate in a broad range of applications. Installation cost is an important part of total-cost-of-ownership for many IoT devices; wiring costs can far exceed the cost of the device itself [76]. Small batteries such as coin cells are sufficient to allow many such devices to operate for months or years but replacing these batteries is often expensive.

Energy scavenging can be used to recover energy from the environment [77], thereby extending unattended lifespan. A variety of energy sources can be used, e.g., radio, thermoelectric, vibration, movement, and light [78]. These sources vary in the amount of power they can harvest as well as the physical characteristics of the scavenging unit.

To achieve power consumption rates consistent with small batteries or energy scavenging, an IoT device may be designed to operate at low duty cycle [79]: the device turns on for a brief interval, performs its sensing/computing/communication tasks, and then goes back to sleep. Deep sleep modes minimize both static and dynamic power, thus conserving the device's available energy. A low duty cycle is consistent with an event-driven architecture in which devices communicate significant or unpredictable changes to signals rather than the entirety of signals. Given that wireless communication requires orders of magnitude more energy per bit than computation, an event-driven architecture that uses local computation to identify events enables low-power operation.

Heterogeneous architectures have been widely used in high-performance embedded systems to reduce power consumption while meeting real-time performance requirements [80]. Given the high computational requirements of machine learning, heterogeneous architectures may provide improved performance/power characteristics. For example, Shen and Srivastava [81] used heterogeneous hardware architectures to improve the energy efficiency of wearable devices that perform inference based on context.

Researchers have modeled hardware implementation platforms for machine learning systems with the goal of cooptimizing algorithms (e.g., deep neural network structures) and hardware. Dai et al. [82] developed a platform-aware neural network architecture search methodology. Cai et al. [83] developed an energy estimation methodology appropriate for guiding neural network architecture search.

In this special issue, Xiao and Liang [84] describe a framework to identify promising locations in deep neural network execution at which to reconfigure dynamically reconfigurable field-programmable gate arrays (FPGAs), and to optimize the specific configurations used. Also in this issue, Zhu et al. [85] present a neural network accelerator appropriate for both convolutional and fully-connected layers. Their architecture is designed to eliminate redundant data transfers and superfluous computations and to restructure the implementation to trade-off precision and parallelism in an application-dependent way.

Microarchitectures, circuits, and devices for IoT intelligence

Research on hardware for efficient machine learning can be divided into three main categories: microarchitectures, circuits, and devices. This section summarizes the current state of the art in each of these research areas.

Many practically important machine learning algorithms require large amounts of memory and computation. Providing adequate memory implies long worst-case access times. Achieving adequate computation speed requires parallelization over many processing elements. For many real-world deep learning networks, there exists no possible assignment of network nodes to (distributed) processing elements that avoids global communication or worst-case memory access penalties because many network nodes must frequently share data with many others. This makes such networks ill-suited for implementation on conventional von Neumann architectures: serial access of weight values through a globally shared memory interface undermines performance and energy efficiency. Thus, conventional central processing units and multiprocessor-based architectures are ill-suited to many machine learning tasks.

Graphics processing units (GPUs) avoid some of the problems of conventional instruction processor microarchitectures. They support highly parallel single-instruction multiple datapath (SIMD) operation, which is well suited to parallelizing propagation of activation function calculation in deep networks, i.e., arithmetic operations on sums of products. Moreover, GPUs often support limited-width arithmetic, making them appropriate for evaluating the (very common) networks in which high-precision weights and activations are unnecessary. GPUs provide some advantages over CPUs in systems requiring high memory bandwidths and high degrees of parallelism. For example, the memory bus speeds of GPUs are often several times higher than those of similarly priced CPUs. For example, the Intel Xeon E7-2830 has a peak memory bus speed of 2.25 GB/s while the Nvidia Quadro Pro P6000 has a memory bus speed of 423 GB/s. This allows higher transfer rates before the bottleneck is reached. However, the memory architecture still requires transfers on a shared bus. GPU-based systems are also often more difficult to program than general-purpose processors, although there has been recent progress on simplifying the process of implementing some classes of machine learning algorithms on GPU-based systems [86]–[91]. Notably, some of these focus on resource-constrained embedded platforms [92].

Many researchers have reported GPU-based system speedups of 10×–1000× relative to CPU-based systems for applications requiring high degrees of parallelism [93]–[96]. However, others argue that most of the reported speedups are the result of asymmetric development and optimization processes followed when making these comparisons [97]. In short, they hypothesize that the lower-level approach to programming and memory management required by GPUs forces developers to think deeply about memory management and expose new opportunities for algorithmic optimization that are not generally back-ported to CPU implementations. In their own evaluation of GPU speed-ups, they found that the much lower speedup of 2.5× is typical when comparing the similarly priced Nvidia GTX280 GPU and Intel Core i7 960 CPU.

FPGAs have been used to accelerate machine learning applications, including deep learning [98], [99]. They enable fine-grained parallelism and therefore have similar advantages to GPUs. Their greater flexibility sometimes facilitates improved performance and energy efficiency but also generally increases implementation cost relative to GPU-based systems, although there has been progress on libraries and frameworks to ease this process [100], [101].

Although GPUs and FPGAs can be used to implement machine learning systems, they were not designed specifically for it. Researchers have also designed architectures specifically for machine learning. Some of their efforts focused on optimizing the mathematical operations common in machine learning systems, e.g., the vector–matrix multiplication operations frequently encountered in deep learning. Several commercially available accelerators designed for neural network applications are already available, e.g., the Intel Movidius NCS2 and Google Coral Dev Board (EdgeTPU) peripheral accelerators, the NVIDIA Tegra TX2 stand-alone accelerator, and the Google Pixel 3, a smartphone containing accelerators.

Bioinspired systems have been designed with several goals in mind, from biological science to solving pressing engineering problems. Some systems were designed to better understand the operation of biological learning systems [102], [103]. These systems generally mimic their biological inspirations as closely as implementation technology permits. Others used only those features of biological systems that improve performance for a particular application of interest. One example includes systems that use analog voltages to represent activation function values within individual network nodes but reverts to digital signals for long-range transmission [104], [105]. TrueNorth is a notable brain-inspired programmable processor using a million digital neurons and 256-million synapses [106]. Biological learning systems also use a combination of analog communication for short-range connections and digital (spiking) communication for long-range connections. Another frequent theme in bioinspired systems is distributing computation capability throughout memory to ameliorate memory access and many-to-many communication bottlenecks [107]. These processor-in-memory systems face several challenges that can be met through novel circuits and devices. For example, researchers have considered 3-D integration of memory and logic to support this class of architecture.

A primary challenge for embedded deep learning systems is the difficulty of very efficiently multiplying inputs with vectors of weight values and accumulating the (generally nonbinary) results. Unconventional architectures, alone, are insufficient to optimally solve this problem; researchers are developing novel devices and circuits capable of performing efficient vector–matrix multiply–accumulate operations with locally stored scalars that can be updated through learning processes.

Energy efficiency, low cost, and high performance are all important enablers of embedded intelligence. Optimizing them requires innovation in each level of the system design process, including the design of devices and their support circuits. Support for efficient vector–matrix multiplication accounts for much of the existing work on using novel devices for machine learning. Many classes of devices have been evaluated for this role. The key requirement is that they can form circuits capable of performing rapid and efficient vector–matrix multiply–accumulate operations. The most common approach is to represent activation function outputs using voltages that are multiplied by weights and summed by arrays of variable-resistance devices connected in parallel. The weight values associated with the devices are programmable. In this approach, matrices are implemented using physical arrays with the same number of rows and columns as the matrix. Devices sit at the row–column intersections and their analog properties are used to multiply input activation values with locally stored weights; typically these weights are resistance states of variable-resistance devices.

The pre-weighted values, represented as currents, are then aggregated, thus completing the multiply–accumulate operation.

The devices needed to implement such efficient vector–matrix multiply–accumulate arrays would ideally have the following properties:

- more compact than digital designs capable of performing the same operations;
- have non-volatile, multibit state for storing weight variables within devices;
- support adequate output variation range to operate in the presence of noise;
- programmability, in which state change magnitude is ideally independent of the current state;
- energy efficiency; and
- reliability, including both controllable process-dependent variation in device parameters and the ability to retain state for long periods of time and survive many reprogramming cycles.

Several contenders for this role exist, including memristors and floating gate transistors as well as electrochemical, capacitance-based, ferroelectric, and optical devices [108]. However, none yet meets all the requirements to implement trainable, highly efficient, fast vector multiply–accumulate operations. A key problem not yet well-addressed by any existing memristive devices is that of training latency, with the best existing devices requiring a day for the relatively simple task of training a two-layer network using 1 million images from the MNIST data set [17]. For additional information, see Yu's survey article [108].

IoT safety and security

IoT and cyber-physical systems are designed to interact with the physical world. Physical safety and information security of IoT devices cannot be treated as separate requirements nor can they be pursued as separate design tasks [109]. Systems can be attacked via networks, physical components such as sensors, and by presenting deceptive inputs to their machine learning algorithms. Researchers have studied attacks that use the special properties of IoT systems. There are several properties that are unfavorable to security, and at least one that favors it.

IoT systems typically have large attack surfaces. They generally contain numerous sensors that are subject to a variety of attacks that can inject false measurements into the system [110], [111]. They also frequently have multiple wireless and/or wired network interfaces that are vulnerable to network attacks. This includes interfaces used for communication with other edge devices, and for communication with the Internet. Researchers found that it was common for IoT systems to have default configurations that are highly vulnerable to infection by viruses [112].

IoT sensors enlarge the attack surface. One example sensor-based attack subjects Micro Electro Mechanical Systems (MEMS) accelerometers to sound at their resonance frequencies, thus causing their signal processing pipelines to record false signals at other frequencies due to aliasing [110]. The attackers were able to produce deceptive variation and offsets in the signals captured by accelerometers and described both hardware and software modifications to guard against the attack.

The difficulty of physically accessing edge devices encourages support for in-network firmware updates. Unfortunately, this opens powerful attack vectors that can enable attackers to gain complete control over entire networks. Millions of IoT devices are publicly visible on the Internet, and thus exposed to network attacks [113]. Ronen et al. [114] demonstrated an attack on Philips Hue IoT-enabled lightbulbs that can gain complete control over the network and spread autonomously via direct communication among edge devices. They exploited weaknesses in secret key protection and a flaw in the implementation of a protocol designed to ensure that physically distant devices be unable to control association with particular networks.

The Mirai botnet was one of the most consequential IoT attacks thus far. It grew to 65,000 infections during its first 20 hours, peaked at 600,000 infections, and maintained a steady-state of over 200,000 [112]. The infected IoT devices were used for denial-of-service attacks on several machines, including those owned by Krebs on Security, the OVH hosting provider and Dyn, with a peak rate of 600 Gb/s. Mirai primarily relied on attacks using a 62-entry dictionary of common user-password combinations, thus exploiting insecure, generally default, configurations.

Researchers have developed several honeypots to attract and observe attacks on IoT systems. These include IoTCandy-Jar [115], which uses machine learning to learn enough about IoT devices to build an interactive honeypot emulating them, and ThingPot [116], which mimics Philips Hue smart lighting systems.

Evaluating IoT system safety and security using techniques that rely on formal system models is difficult. IoTs are extremely large-scale systems composed of heterogeneous components. The components are designed by many companies at different times, many of the designers have little exposure to formal security analysis methods, and different components are owned and governed by different organizations and people. This resulting lack of complete formal specifications by any design team makes it all but impossible to analyze an entire large-scale IoT system: no single person knows the appropriate model. Even if a formal model were known, it would generally contain too many irreducible components for model checking to be feasible. In summary, formal methods such as model checking are generally infeasible for large-scale IoT systems.

IoT analysis challenges are compounded by the fact that the directed partially random fault processes contributing to IoT failure are interdependent: stochastic independence cannot be assumed. Determining the parameters of the random processes modeling faults in IoT components is difficult; determining the correlations among these parameters is harder still. Components are coupled via often implicit or unknown environmental parameters including social/political events (e.g., in targeted attacks), and cyclical and acyclic natural phenomena such as weather patterns. Identifying catastrophic IoT system failure modes will be more akin to predicting rare, catastrophic events in financial systems than analyzing more isolated embedded systems. Manually characterizing indirect relationships among IoT component fault processes is impractical; the number of intercomponent correlations and their governing parameters is beyond the capacity of any single designer to manually determine and understand.

A recent report produced as a result of a workshop on embedded system security provides additional details on security challenges in the IoT [111].

Application domains

We now describe several IoT application domains for which embedded intelligence is particularly relevant and point out prior work and application-relevant technologies supporting embedded intelligence.

Computer vision and surveillance

Computer vision has transformed numerous domains including security, healthcare, banking, and transportation. Its applications are expected to have a market value of $15 billion by 2022 [117]. Although there are many opportunities for computer vision in IoT applications including transportation, security, agriculture, and other domains, the compute power and energy constraints of IoT systems sometimes prevent its use.

To illustrate the computational cost of a typical computer vision application, we will consider a highly optimized version of the inception neural network architecture [118]. Although it reduces the number of parameters by 2.5× relative to another state-of-the-art architecture (VG-GNet [119]), it still requires 60-billion multiply–accumulate operations per inference, i.e., 0.1 frames/s using a 20-core Xeon E5-2680 system, putting the frame rate for processors typically found in commodity mass-market embedded vision systems such as the Raspberry Pi 3 at a frame every 2 minutes for a four-core Arm Cortex-A53 running at 1.2 GHz. Simpler networks and faster vision algorithms exist, especially for narrowly defined applications. However, speed is often bought with a reduction in accuracy. In short, efficient embedded computer vision is possible, but requires careful hardware and algorithm designs that account for optimization opportunities unique to the application of interest; the approaches used for highly accurate general-purpose computer vision (even those that have been highly optimized) are too energy-intensive and inefficient for use on edge nodes in the IoT. We describe technologies supporting efficient vision in the "State of the art and promising directions" section.

Video data will account for 82% of all IP traffic, with a volume of 396 EB per month, by 2022 [120]. Video analytics is becoming an increasingly important source for deriving business insights into various use cases like digital surveillance, retail analytics, and smart cities. Globally installed video surveillance cameras were estimated to generate more than 859 PB/day in 2017 [121]. Despite its abundance and ubiquity, visual data are still significantly underutilized due to the computation requirements and data volume. There have been advances in converting video data into actionable information at scale, but they have often limited to the use of cloud infrastructure and/or focused on narrow applications.

Embedded intelligence with ubiquitous connectivity makes multimodal inference possible. This is particularly useful for inventory tracking and warehouse management in logistics applications. In these applications, connected cameras and ubiquitously deployed sensor tags, e.g., Bluetooth beacons and radio-frequency identification (RFID) tags, are used to track personnel and inventory and identify their interactions.

As a result of the data rates common for vision applications, multiple approaches are being used to optimize IoT vision systems, including special-purpose hardware and architectures to accelerate deep CNNs [122], spatial and temporal decimation, and using a multiphase process in which efficient, low-accuracy (but low false-negative rate) analysis algorithms are used to filter out most images, leaving only a few for analysis by more computationally intensive but accurate algorithms.

Transportation

The automotive market is valued at over $1.5 trillion [123] with the smart transportation market somewhere around $100 million. It is now being disrupted by four main trends: electrification, connectivity, autonomous driving, and mobility affordability. Electrification and connectivity make data sharing easier; electrification digitizes data in the first place whereas connectivity allows data sharing. Both are prerequisites of autonomous driving. Mobility affordability is a direct implication of a sharing economy. On-demand mobility will further decouple data ownership from vehicle ownership.

Autonomous driving requires an abrupt transformation to dealing with 4.5 TB of data per day [124]. Connectivity is a key prerequisite. It is essentially a multitier IoT network. In multivehicle networks, each car shares information with others and acts upon the collective information. Within each vehicle, there are other networks connecting hundreds of electronic control units (ECUs) in a highly distributed system. In addition to the challenges of processing video at high frame rates, autonomous driving poses a major challenge for real-time operations. This requires ECUs within a vehicle and across vehicles to be able to sense, communicate, and act in real-time in a decentralized manner, driving embedded in-network analysis and decision making.

Industrial automation

Networked sensors and actuators have a long history of use in industrial automation applications, and wireless networks are seeing increasing use in the IoT. Depending on legal requirements, the cost of installing cables for an industrial automation system ranges from hundreds to thousands of U.S. dollars per foot [125]. Recent advances in wireless network reliability (>99.999%) and battery lifespan (up to 10 years) in some industrial automation applications allows them to be substituted for wired networks [126]. IEEE 802.15.4 is commonly used for communication in these circumstances. Wired and wireless, the IoT is being used to enable distributed plant control in industrial automation applications.

IoT security is a pressing problem for industrial automation and transportation systems [127]. For example, in 2003 the Slammer worm infected a nuclear power plant [128] and a computer virus halted passenger and freight trains. In addition, several attacks on general-purpose computers have damaged industrial automation systems. Stuxnet provides an example of a virus targeting a particular industrial material purification process [129].

Fragmentation is another substantial challenge for the IoT in industrial automation applications. There are many different communication standards for the heterogeneous sensors and actuators in this application. The need for interdomain interfaces results in complex network architectures.

Wearables and medical devices

Wearables are miniature embedded devices worn by people. They perform day-to-day data acquisition and analysis. Thanks to the ongoing miniaturization of electronics, there has been an increase in wearables in health, sports, surveillance, and social applications.

Energy consumption has been a primary constraint for wearable sensing and pattern recognition. Their miniature form factor constrains battery capacity, i.e., the total stored energy. Data sensing and pattern recognition are the primary functions of wearables. Progress on low-power sensors, e.g., MEMS-based inertial measurement units (IMUs) and low-power CMOS camera sensors, has made energy-efficient wearable sensing increasingly practical.

MEMS-based IMUs, e.g., accelerometers and gyroscopes, have been widely used in wearables for motion tracking and activity recognition for over a decade. The active power consumption of MEMS-based accelerometers can now be kept under 10 pW for low-frequency sampling, e.g., the ST Microelectronics LIS3DH. Activity-aware adaptive sensing techniques can further reduce the power consumption of motion sensing by another order of magnitude. Progress has also been made on machine learning-based human activity pattern recognition [130]–[133]. A wide range of machine learning algorithms, such as random forest and Bayesian networks, have been widely adopted in motion-based wearable sensing systems. Existing commercial wearable products are capable of real-time recognition of a wide range of human physical activities, such as running, cycling, swimming, and sleeping. High-precision quantification of human motion, e.g., 3-D human motion tracking, has also been achieved, empowering a range of sports and medical applications.

Compared to IMUs, wearable cameras are more challenging, in terms of performance requirements and power consumption. For example, running face detection on Google Glass depletes its battery in 38 minutes [134]. Conventional machine learning algorithms have made tremendous progress on human activity recognition, and recent progress on deep learning techniques has delivered unparalleled accuracy. Deep-learning-based methods can automatically derive high-level features from raw data, hence effectively reducing the need for domain-specific, often handcrafted, feature design.

The medical industry has slowly started entering the connected world of IoT. Electronic medical records are being implemented at scale. Connected devices have arisen so that doctors can digitally prescribe medication, and patients can easily access their records over the Internet. The IoT provides medical experts with ubiquitous, continuous, and remote sensing and actuation technologies. This enables quick turnaround of information sharing and processing, and may reduce clinical error rates. It also enables predictive medicine.

The IoT has the potential to enable continuous personalized automated monitoring of medical conditions. Currently, most evaluation and feedback occur at rare appointments with nurse practitioners and doctors, and these events are likely to remain rare due to time constraints on medical practitioners and patients. The IoT can make this continuous, enabling improved diagnoses and treatment of conditions with intermittent or variable symptoms. Recent progress on ensembles of machine learning systems has enabled energy-efficient pattern recognition in wearable medical devices [135].

Agriculture

Wireless distributed sensing has a long history of use in agricultural applications, and we expect its use to expand. Data on the total market size are in disagreement, but a multibillion US dollar estimate for the agricultural IoT market is safe. As indicated in a recent survey by Ojha et al. [136], the main applications within agriculture are monitoring of automated farming systems (including irrigation management [137], [138]) [139]–[142], pest and disease monitoring [143], [144], controlled application of soil amendments [145], livestock surveillance [146], [147], water quality monitoring [148], [149], greenhouse gas monitoring [150], and asset tracking [151].

There is a wide range of agriculture-relevant sensing modes available, including moisture, temperature, conductivity, salinity, pressure, wind, rainfall, and solar radiation. The states of plants can also be monitored, including leaf hydration and photosynthesis rates. Livestock [152] and pest [153]–[156] activities and locations can be detected using several sensing modes, including global positioning system (GPS) tracking, computer vision, and audio classification.

The agricultural sensing problem domain has several special characteristics. Network spans are often quite large, with sensors within a single network spread over kilometers or more. This has traditionally been managed using cellular communication, or multihop networks and moderate-range communication technologies such as IEEE 802.15.4. Fortunately, many agricultural applications require low data rates, with each sensor providing samples every hour or day, not every millisecond. As a consequence, LPWAN communication technologies (see the "Wireless communication trends and IoT implications" section) are particularly well suited to many agricultural sensing applications; we can expect their rapid expansion in this market during the next 5 years. Pest detection applications [154]–[158] provide a contrasting example, with many requiring high data rates and complex analysis, e.g., video or laser-based insect detection.

Some agricultural applications benefit from subterranean sensing, requiring unconventional approaches to wireless communication, the simplest of which is designing sensor nodes that separate antennas from sensors by a meter or more of wires. Researchers have also considered using unusual communication channels such as ultralow frequency radio for short-range underground communication [159].

Embedded intelligence is valuable in this application because it breaks the high-latency dependence on potentially multihop, long-range communication.

A thorough treatment of agricultural sensing research requires a dedicated survey, and there are several available to the interested reader [136], [160].

Challenges

This section describes several challenges facing embedded intelligence in the IoT.

Small data

Data scarcity is the foremost roadblock to enabling edge intelligence for many IoT domains. A fundamental challenge is that accurate modeling of a complex system builds upon high-dimensional feature representations. A large amount of high-quality data are needed to develop such models. Let us use predictive maintenance to illustrate the challenge.

Predictive maintenance is a widely used embedded intelligence service targeting a broad range of industrial domains. It uses data-driven models to detect progressive system defects and schedule maintenance in time to prevent system failures. This is challenging because system faults are uncommon and difficult to capture, so representative data are scarce and poorly labeled. As a result, data-driven system modeling methods are unable to accurately describe critical yet rare system states. Therefore, fault diagnosis and classification methods are often developed by experts using domain-specific knowledge. This process is ad hoc and error-prone. Yield models are domain-specific and of limited use to other applications.

Recent research efforts have focused on the small data challenge. Learning methods, such as transfer learning and one-shot learning, hold the potential to reduce the amount of data required for model development. However, data scarcity will remain a primary challenge to embedded intelligence in the foreseeable future.

Cost of ownership

The total cost of ownership is an important consideration for both cloud systems and edge devices. The components of cost of ownership in these two use cases are, however, very different. If wiring is required to supply either power or network connectivity, the cost of the wire drop may exceed the cost of the IoT device [79]. Physical visits to the device, whether for maintenance or replacement, are also expensive. IoT devices should be designed for relatively long lifetimes. One important factor is thermal behavior given the strong dependence of degradation rate on temperature and thermal cycling.

Power and energy consumption

Power and energy are first-order considerations in any battery-powered or energy-scavenging IoT system. Given the high computational requirements of machine learning systems, power and energy considerations have broad impacts on system design. For example, they may dictate sampling rates for inference.

Wireless communication power consumption, reliability, and latency implications

Some use cases may allow wired network connections. However, many IoT applications will require wireless operation. Wireless links generally provide lower bandwidth than wired links. Furthermore, the energy required to run the radio may limit its duty cycle, further restricting throughput. Training updates for machine learning systems may require large data transfers.

Multitenancy and heterogeneity

With the emerging trend toward distributed and decentralized architectures, heterogeneity is becoming the norm. Specialized hardware is developed for resource-constrained devices at the edge while general-purpose processors and GPUs still remain dominant in the cloud. The ultimate goal is a minimized total cost of ownership for an end-to-end system. Application developers intend to save cost with specialized hardware at the edge and cloud service providers intend to provide general-purpose platforms for their diverse customer bases. This implies a multitenant environment where multiple stakeholders share a common infrastructure; this makes resource optimization even more challenging due to privacy and security issues. A virtualized infrastructure is required for multitenancy. On the other hand, scheduling of multitenant workloads over devices from edge to cloud is still an open problem.

The growth of embedded intelligence in the IoT is inevitable. This trend is motivated by high wireless communication costs, requiring analysis and potentially decision making on edge devices or within the network instead of relying on remote servers or other cloud infrastructure. It is enabled by ongoing advances in algorithms, architectures, circuits, and devices supporting efficient machine learning. The trend will have major impacts on security and privacy, some positive and some negative. This article has summarized the causes and implications of this trend in the context of several economically important application domains. ■

Acknowledgments

We would like to thank E. S. Lubana for his numerous helpful comments on a draft of this article.

References

[1] A. Bhutani and P. Wadhwani, *Low Power Wide Area Network (LPWAN) Market Size by Component*, Global Market Insights, Tech. Rep., Mar. 2019.

[2] U. Raza, P. Kulkarni, and M. Sooriyabandara, "Low power wide area networks: An overview," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 2, pp. 855–873. May 2017.

[3] L. Casals et al., "Modeling the energy performance of LoRaWAN," *Sensors*, vol. 17, no. 10, 2017, pp. 2364–2393.

[4] F. Adelantado et al., "Understanding the limits of LoRaWAN," *IEEE Commun. Mag.*, vol. 55, no. 9, pp. 34–40, 2016.

[5] LoRa Alliance Technical Committee, "LoRaWAN 1.1 Specification," Technical Specification (TS), version 1.1, Oct. 2017. [Online]. Available: https://lora-alliance.org/sites/default/files/2018-04/lorawantm_specification_-v1.1.pdf

[6] A. D. Zayas and P. Merino, "The 3GPP NB-IoT system architecture for the Internet of Things," in *Proc. Int. Conf. Commun. Workshop*, May 2017, pp. 277–282.

[7] A. Adhikary, X. Lin, and Y. P. E. Wang, "Performance evaluation of NB-IoT coverage," in *Proc. Veh. Technol. Conf.*, Sep. 2016, pp. 1–5.

[8] J. Schlienz and D. Raddino, "Narrowband Internet of Things whitepaper," *Microw. Mag.*, vol. 8, no. 1, pp. 76–82, 2016.

[9] J. C. Zuniga and B. Ponsard, *SigFox System Description*, SigFox, Tech. Rep., Jun. 2017. [Online]. Available: https://tools.ietf.org/html/draft-zuniga-lpwan-sigfox-system-description-03

[10] SigFox, *SigFox Technical Overview*, SigFox, Tech. Rep., Jul. 2017. [Online]. Available: https://www.element14.com/community/docs/DOC-87914/l/sigfox-technical-overview

[11] M. Wolf, *International Roadmap for Devices and Systems, 2017 Edition: System and Architecture*, IEEE, Tech. Rep., 2017. [Online]. Available: https://irds.ieee.org/images/files/pdf/2017/2017IRDS_SA.pdf

[12] J. Polastre et al., "Analysis of wireless sensor networks for habitat monitoring," in *Wireless Sensor Networks*, C. S. Raghavendra, K. M. Sivalingam, and T. Znati, Eds. Boston, MA, USA: Springer, 2004, pp. 399–423.

[13] A. Haeberlen et al., "Practical robust localization over large-scale 802.11 wireless networks," in *Proc. Int. Conf. Mobile Comput. Netw.*, Sep. 2004, pp. 70–84.

[14] J. Zhou et al., "Security and privacy for cloud-based IoT: Challenges," *IEEE Commun. Mag.*, vol. 65, no. 1, pp. 26–33, Jan. 2017.

[15] M. L. Minsky and S. A. Papert, *Perceptrons: An Introduction to Computational Geometry.* The MIT Press, 1969.

[16] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biol. Cybern.*, vol. 36, no. 4, pp. 193–202, Apr. 1980.

[17] Y. LeCun et al., "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[18] A. Halevy, P. Norvig, and F. Pereira, "The unreasonable effectiveness of data," *IEEE Intell. Syst.*, vol. 24, no. 2, pp. 8–12, Mar. 2009.

[19] D. Ciresan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 3642–3649.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[21] K. He et al., "Deep residual learning for image recognition," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[22] B. D. Lucas et al., "An iterative image registration technique with an application to stereo vision," in *Proc. Image Understand. Workshop*, Vancouver, BC, 1981, pp. 121–130.

[23] C. Tomasi and T. K. Detection, *Tracking of Point Features*, Carnegie Mellon University, Tech. Rep. CMU-CS-91-132, 1991.

[24] D. G. Lowe et al., “Object recognition from local scale-invariant features,” in *Proc. Int. Conf. Comput. Vis.*, vol. 99, no. 2, 1999, pp. 1150–1157.

[25] Y. LeCun et al., “Gradient-based learning applied to document recognition,” *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[26] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint*, arXiv:1409.1556, 2014.

[27] C. Szegedy et al., “Going deeper with convolutions,” in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 1–9.

[28] K. He et al., “Deep residual learning for image recognition,” in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[29] Stanford Vision Lab, Stanford University, and Princeton University, “Imagenet website.” [Online]. Available: http://www.image-net.org/

[30] D. Pisinger, “Deep learning website.” [Online]. Available: http: //deeplearning.net/datasets/

[31] P. P. Ray, “A survey of IoT cloud platforms,” *Future Comput. Inf. J.*, vol. 1, nos. 1–2, pp. 35–46, 2016.

[32] S. J. Pan and Q. Yang, “A survey on transfer learning,” *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2009.

[33] R. Raina et al., “Self-taught learning: Transfer learning from unlabeled data,” in *Proc. Int. Conf. Mach. Learn.*, Jun. 2007, pp. 759–766.

[34] Y. Wang and Q. Yao, “Few-shot learning: A survey,” *arXiv*, May 2019. [Online]. Available: http://arxiv.org/abs/1904.05046

[35] J. Snell, K. Swersky, and R. Zemel, “Prototypical networks for few-shot learning,” in *Proc. Adv. Neural Inf. Process. Syst.*, Dec. 2017, pp. 4077–4087.

[36] R. Vilalta and Y. Drissi, “A perspective view and survey of meta-learning,” *Artif. Intell. Rev.*, vol. 18, no. 2, pp. 77–95, Jun. 2002.

[37] C. Finn et al., “One-shot visual imitation learning via meta-learning,” *arXiv*, Sep. 2017. [Online]. Available: https://arxiv.org/abs/1709.04905

[38] R. Buyya et al., “Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility,” *Future Gen. Comput. Syst.*, vol. 25, no. 6, pp. 599–616, 2009.

[39] R. Caceres and A. Friday, “Ubicomp systems at 20: Progress, opportunities, and challenges,” *IEEE Perv. Comput.*, vol. 11, no. 1, pp. 14–21, 2011.

[40] J. Gubbi et al., “Internet of Things (IoT): A vision, architectural elements, and future directions,” *Future Gen. Comput. Syst.*, vol. 29, no. 7, pp. 1645–1660, 2013.

[41] C. Brewster et al., “IoT in agriculture: Designing a Europe-wide large-scale pilot,” *IEEE Commun. Mag.*, vol. 55, no. 9, pp. 26–33, 2017.

[42] Y. Lecun et al., “Handwritten digit recognition with a back-propagation network,” *Neural Inf. Process. Syst.*, vol. 2, pp. 396–404, Nov. 1997.

[43] K. Siu et al., “Memory requirements for convolutional neural network hardware accelerators,” in *Proc. Int. Symp. Workload Characterization*, Sep. 2018, pp. 111–121.

[44] Y. Guo, A. Yao, and Y. Chen, “Dynamic network surgery for efficient DNNs,” in *Proc. Conf. Neural Inf. Process. Syst.*, 2016, pp. 1379–1387.

[45] B. Zhuang et al., “Towards effective low-bitwidth convolutional neural networks,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7920–7928.

[46] J. Albericio et al., “Bit-pragmatic deep neural network computing,” in *Proc. Int. Symp. Microarchit.*, Oct. 2017, pp. 382–394.

[47] P. Coussy et al., “Fully binary neural network model and optimized hardware architectures for associative memories,” *ACM J. Emerg. Technol. Comput. Syst.*, vol. 11, no. 4, Apr. 2015, Art. no. 35.

[48] M. Courbariaux, Y. Bengio, and J.-P. David, “BinaryConnect: Training deep neural networks with binary weights during propagations,” in *Proc. Adv. Neural Inf. Process. Syst.*, Dec. 2015, pp. 3123–3131.

[49] S. Yu et al, “Binary neural network with 16 Mb RRAM macro chip for classification and online training,” in *Proc. Int. Electron Devices Meeting*, Sep. 2016, pp. 16.2.1–16.2.4.

[50] A. G. Howard et al., “MobileNets: Efficient convolutional neural networks for mobile vision applications,” *arXiv*, Apr. 2017.

[51] M. Imani et al., “Efficient associative search in brain-inspired hyperdimensional computing,” *IEEE Design Test*, vol. 40, no. 1, Jan./Feb. 2020, doi: 10.1109/MDAT.2019.2919954.

[52] X. He et al., “A quantitative exploration of collaborative pruning and approximation towards energy-efficient deep neural networks,” *IEEE Design Test*, vol. 40, no. 1, Jan./Feb. 2020, doi: 10.1109/MDAT.2019.2943575.

[53] M. Toneva et al., “An empirical study of example forgetting during deep neural network learning,” in *Proc. Int. Conf. Learn. Represent.*, May 2019.

[54] I. Loshchilov and F. Hutter, “Online batch selection for faster training of neural networks,” in *Proc. Int. Conf. Learn. Represent.*, May 2015.

[55] P. Molchanov et al., “Importance estimation for neural network pruning,” in *Proc. Conf. Comput. Vis. Pattern Recogn.*, Jun. 2019, pp. 11264–11272.

[56] R. R. Selvaraju et al., “Grad-CAM: Visual explanations from deep networks via gradient-based localization,” in *Proc. Int. Conf. Comput. Vis.*, Oct. 2017, pp. 618–626.

[57] E. S. Lubana, V. Aggarwal, and R. P. Dick, “Machine Foveation: An application-aware compressive sensing framework,” in *Proc. Data Compress. Conf.*, Mar. 2019.

[58] S. Li et al., “Demystifying ResNet,” *arXiv*, arXiv:1611.01186, Nov. 2016.

[59] V. Sze et al., “Efficient processing of deep neural networks: A tutorial and survey,” *Proc. IEEE*, vol. 105, no. 12, pp. 3123–3131, Dec. 2017.

[60] D. L. Donoho, “For most large underdetermined systems of linear equations the minimal 1-norm solution is also the sparsest solution,” *Commun. Pure Appl. Math.*, vol. 59, no. 6, pp. 797–829, 2006.

[61] S. Li, L. D. Xu, and X. Wang, “Compressed sensing signal and data acquisition in wireless sensor networks and Internet of Things,” *IEEE Trans. Ind. Inform.*, vol. 9, no. 4, pp. 2177–2186, Nov. 2013.

[62] E. S. Lubana and R. P. Dick, “Digital Foveation: An energy-aware machine vision framework,” *IEEE Trans. Comput.-Aided Design Integrat. Circuits Syst.*, vol. 37, pp. 2371–2380, Nov. 2018.

[63] K. Yasumoto, H. Yamaguchi, and H. Shigeno, “Survey of real-time processing technologies of IoT data streams,” *J. Inf. Process.*, vol. 24, no. 2, pp. 195–202, 2016.

[64] S. Ahmad et al., “Unsupervised real-time anomaly detection for streaming data,” *Neurocomputing*, vol. 262, pp. 134–147, 2017.

[65] M. Mohammadi et al., “Deep learning for IoT big data and streaming analytics: A survey,” *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 2923–2960, 2018.

[66] S. Chen et al., “A vision of IoT: Applications, challenges, and opportunities with china perspective,” *IEEE Internet Things J.*, vol. 1, no. 4, pp. 349–359, 2014.

[67] A. Al-Fuqaha et al., “Internet of Things: A survey on enabling technologies, protocols, and applications,” *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2347–2376, 2015.

[68] S. Cherrier, Z. Movahedi, and Y. M. Ghamri-Doudane, “Multi-tenancy in decentralised IoT,” in *Proc. World Forum Internet Things (WF-IoT)*, Dec. 2015, pp. 256–261.

[69] H. Kim and M. Wolf, “Distributed tracking in a large-scale network of smart cameras,” in *Proc. Int. Conf. Distrib. Smart Cameras*, New York, NY, USA, 2010, pp. 8–16. [Online]. Available: http://doi.acm.org/10.1145/1865987.1865990

[70] M. Satyanarayanan, “The emergence of edge computing,” *Computer*, vol. 50, no. 1, pp. 30–39, 2017.

[71] H.-M. Chu et al., “Scheduling in visual fog computing: NP-completeness and practical efficient solutions,” in *Proc. AAAI*, 2018, pp 6127–6135.

[72] M. Olguín et al., “Scaling on the edge-a benchmarking suite for human-in-the-loop applications,” in *Proc. Symp. Edge Comput.*, Oct. 2018, pp. 323–325.

[73] S. Sriram and S. S. Bhattacharyya, *Embedded Multiprocessors: Scheduling and Synchronization.* CRC Press: Boca Raton, FL, 2018.

[74] A. Tumanov et al., “Tetrisched: Global rescheduling with adaptive plan-ahead in dynamic heterogeneous clusters,” in *Proc. Eur. Conf. Comput. Syst.*, 2016, p. 35.

[75] N. Wang and J. Wu, “Optimal cellular traffic offloading through opportunistic mobile networks by data partitioning,” in *Proc. Int. Conf. Commun. (ICC)*, 2018, pp. 1–6.

[76] M. Wolf, “The physics of event-driven IoT systems,” *IEEE Design Test*, vol. 34, no. 2, pp. 87–90, Apr. 2017.

[77] J. A. Paradiso and T. Starner, “Energy scavenging for mobile and wireless electronics,” *IEEE Perv. Comput.*, vol. 4, no. 1, pp. 18–27, Jan. 2005.

[78] S. Roundy, P. K. Wright, and J. Rabaey, “A study of low level vibrations as a power source for wireless sensor nodes,” *Comput. Commun.*, vol. 26, pp. 1131–1144, Oct. 2003.

[79] M. Wolf, “Ultralow power and the new era of Not-So-VLSI,” *IEEE Design Test*, vol. 33, no. 4, pp. 109–113, Aug. 2016.

[80] M. Wolf, *High Performance Embedded Computing: Applications in Cyber-Physical Systems and Mobile Computing*, 2nd ed. San Francisco, CA, USA: Elsevier, 2014.

[81] C. Shen and M. Srivastava, “Exploring hardware heterogeneity to improve pervasive context inferences,” *Computer*, vol. 50, no. 6, pp. 19–26, 2017.

[82] X. Dai et al., “ChamNet: Towards efficient network design through platform-aware model adaptation,” in *Proc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2019, pp. 11398–11407.

[83] E. Cai et al., “NeuralPower: Predict and deploy energy-efficient convolutional neural networks,” in *Proc. Asian Conf. Mach. Learn.*, Nov. 2017.

[84] Q. Xiao and Y. Liang, “Fune: An FPGA tuning framework for CNN acceleration,” *IEEE Design Test*, vol. 40, no. 1, Jan./Feb. 2020, doi: 10.1109/MDAT.2019.2908549.

[85] H. Zhu, Y. Wang, and C.-J. R. Shi, “Tanji: A general-purpose neural network accelerator

with unified crossbar architecture," *IEEE Design Test*, vol. 40, no. 1, Jan./Feb. 2020, doi: 10.1109/MDAT.2019.2952329.

[86] "PyTorch website." [Online]. Available: https://pytorch.org/

[87] "TensorFlow website." [Online]. Available: https://www.tensorflow.org/

[88] R. Collobert, K. Kavukcuoglu, and C. Farabet, "Torch7: A Matlab-like environment for machine learning," in *Proc. BigLearn Neural Inf. Process. Syst. Workshop*, 2011.

[89] J. Bergstra et al., "Theano: A CPU and GPU math compiler in python," in *Proc. Python Sci. Conf.*, 2010, pp. 1–48.

[90] T. Chen et al., "MXNet: A flexible and efficient machine learning library for heterogeneous distributed systems," in *Proc. Neural Inf. Process. Syst., Workshop Mach. Learn. Syst.*, 2016.

[91] "NVIDIA TensorRT website." [Online]. Available: https://developer.nvidia.com/tensorrt

[92] "TensorFlow Lite website." [Online]. Available: https://www.tensorflow.org/lite/

[93] N. Bell and M. Garland, "Implementing sparse matric-vector multiplication on throughput-oriented processors," in *Proc. Conf. Supercomput.*, 2009, pp. 18:1–18:11.

[94] L. Genovese, "Graphic processing units: A possible answer to HPC," in *Proc. ABINIT Develop. Workshop*, 2009.

[95] C. Kim et al., "FAST: Fast architecture sensitive tree search on modern CPUs and GPUs," in *Proc. Int. Conf. Manage. Data*, 2010, pp. 339–350.

[96] J. Tölke and M. Krafczyk, "TeraFLOP computing on a desktop PC with GPUs for 3D CFD," *Int. J. Comput. Fluid Dynam.*, vol. 7, pp. 443–456, 2008.

[97] V. W. Lee et al., "Debunking the 100X GPU vs. CPU myth: An evaluation of throughput computing on CPU and GPU," in *Proc. Int. Symp. Comput. Archit.*, Jun. 2010, pp. 451–460.

[98] G. Lacey, G. W. Taylor, and S. Areibi, "Deep learning on FPGAs: Past, present, and future," *arXiv*, arXiv:1602.04283, Feb. 2016.

[99] C. Zhang et al., "Optimizing FPGA-based accelerator design for deep convolutional neural networks," in *Proc. Int. Symp. Field-Programmable Gate Arrays*, Feb. 2015, pp. 161–170.

[100] G. Hegde et al., "CaffePresso: An optimized library for deep learning on embedded accelerator-based platforms," in *Proc. Int. Conf. Compliers, Archit. Synthesis Embedded Syst.*, Oct. 2016, pp. 1–10.

[101] Q. Yu et al., "A deep learning prediction process accelerator based FPGA," in *Proc. Int. Symp. Cluster, Cloud Grid Comput.*, May 2015, pp. 1159–1162.

[102] M. Djurfeldt et al., "Brain-scale simulation of the neocortex on the IBM Blue Gene/L supercomputer," *IBM J. Res. Develop.*, vol. 52, pp. 31–41, Jan. 2008.

[103] R. Cattell and A. Parker, "Challenges for brain emulation: Why is building a brain so difficult?" *Natural Intell.*, vol.1, no. 3, Feb. 2012.

[104] S. B. Laughlin, "Energy as a constraint on the coding and processing of sensory information," *Curr. Opin. Neurobiol.*, vol. 11, pp. 475–480, 2001.

[105] J. E. Niven and S. B. Laughlin, "Energy limitation as a selective pressure on the evolution of sensory systems," *J. Exp. Biol.*, vol. 211, pp. 1792–1804, Apr. 2008.

[106] F. Akopyan et al., "TrueNorth: Design and tool flow of a 65 mW 1 million neuron programmable neurosynaptic chip," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 34, no. 10, pp. 1537–1557, Aug. 2015.

[107] J. Ahn et al., "A scalable processing-in-memory accelerator for parallel graph processing," in *Proc. Int. Symp. Comput. Archit.*, vol. 43, no. 3, Jun. 2015, pp. 105–117.

[108] S. Yu, "Neuro-inspired computing with emerging nonvolatile memory," *Proc. IEEE*, vol. 106, no. 2, pp. 108–122, Jan. 2018.

[109] M. Wolf and D. Serpanos, "Safety and security in cyber-physical systems and Internet-of-Things systems," *Proc. IEEE*, vol. 106, no. 1, pp. 9–20, Jan. 2018.

[110] T. Trippel et al., "WALNUT: Waging doubt on the integrity of MEMS accelerometers with acoustic injection attacks," in *Proc. Eur. Symp. Security Privacy*, Apr. 2017, pp. 3–18.

[111] D. Anthony et al., *Embedded Security Challenges in a Connected World*, Computing Community Consortium, Tech. Rep. 1, 2019.

[112] M. Antonakakis et al., "Understanding the Mirai botnet," in *Proc. USENIX Security Symp.*, Aug. 2017, pp. 1093–1110.

[113] "Shodan website." [Online]. Available: https://www.shodan.io/

[114] E. Ronen et al., "IoT goes nuclear: Creating a ZigBee chain reaction," in *Proc. Symp. Security Privacy*, May 2017, pp. 195–212.

[115] T. Luo et al., "IoTCandyJar: Towards an intelligent-interaction honeypot for IoT devices," in *Proc. Black Hat*, Aug. 2017.

[116] M. Wang, J. Santillan, and F. Kuipers, “ThingPot: An interactive Internet-of-Things honeypot,” *arXiv*, Aug. 2018.

[117] Market Research Future, “Machine vision market research report-global forecast 2022,” May 2017. [Online]. Available: https://www.marketresearchfuture.com/reports/machine-vision-market-1510

[118] C. Szegedy et al., “Rethinking the inception architecture for computer vision,” in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2016.

[119] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in *Proc. Int. Conf. Learn. Represent.*, 2015.

[120] “Cisco visual networking index: Forecast and trends, 2017–2022 white paper.” [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html

[121] J. Kropley, “Top video surveillance trends for 2015,” IHS Technology, Tech. Rep., 2015. [Online]. Available: https://technology.ihs.com/api/binary/520143

[122] S. Mukhopodhyay et al., “The camel approach to stacked sensor smart cameras,” in *Proc. Design Autom. Test Eur. Conf. Exhibit.*, Mar. 2018, pp. 1299–1303.

[123] “Data is the new oil in the future of automated driving.” [Online]. Available: https://www.mckinsey.com/~/media/mckinsey/industries/hightech/ourinsights/disruptivetrendsthatwilltransformtheautoindustry/auto2030reportjan2016.ashx

[124] “Data is the new oil in the future of automated driving.” [Online]. Available: https://newsroom.intel.com/editorials/krzanich-the-future-of-automated-driving/#gs.bhu7hv

[125] D. Dujovne et al., “6TiSCH: Deterministic IP-enabled industrial Internet (of things),” *IEEE Commun. Mag.*, vol. 52, pp. 36–41, Dec. 2014.

[126] T. Watteyne et al., “Technical overview of SmartMesh IP,” in *Proc. Int. Conf. Innovative Mobile Internet Services Ubiquitous Comput.*, Jul. 2013.

[127] A.-R. Sadeghi, C. Wachsmann, and M. Waidner, “Security and privacy challenges in industrial internet of things,” in *Proc. Design Autom. Conf.*, Jun. 2015, pp. 1–6.

[128] K. Poulsen, “Slammer worm crashed Ohio nuke plant network,” *The Register*, Aug. 2003.

[129] B. Miller and D. Rowe, “A survey of SCADA and critical infrastructure incidents,” in *Proc. Res. Inf. Technol.*, 2012, pp. 51–56.

[130] A. Mosenia et al., “Wearable medical sensor-based system design: A survey,” *Trans. Multi-Scale Comput. Syst.*, vol. 3, no. 2, pp. 124–138, Mar. 2017.

[131] S. C. Mukhopadhyay, “Wearable sensors for human activity monitoring: A review,” *Sensors J.*, vol. 15, no. 3, pp. 1321–1330, Dec. 2015.

[132] O. D. Lara and M. A. Labrador, “A survey on human activity recognition using wearable sensors,” *IEEE Commun. Surveys Tuts.*, vol. 15, no. 3, pp. 1192–1209, Nov. 2012.

[133] Q. Liu et al., “Gazelle: Energy-efficient wearable analysis for running,” *IEEE Trans. Mobile Comput.*, vol. 16, no. 9, pp. 2531–2544, Sep. 2017.

[134] R. LiKamWa et al., “Draining our glass: An energy and heat characterization of Google Glass,” in *Proc. Asia-Pacific Workshop Syst.*, Jun. 2014, Art. no. 10.

[135] H. Yin and N. K. Jha, “A health decision support system for disease diagnosis based on wearable medical sensors and machine learning ensembles,” *IEEE Trans. Multi-Scale Comput. Syst.*, vol. 3, no. 4, pp. 228–241, Dec. 2017.

[136] T. Ojha, S. Misra, and N. S. Raghuwanshi, “Wireless sensor networks for agriculture: The state-of-the-art in practice and future challenges,” *Comput. Electron. Agriculture*, vol. 118, pp. 66–84, Sep. 2015.

[137] S. Adamala, N. S. Raghuwanshi, and A. Mishra, “Development of surface irrigation systems design and evaluation software,” *Comput. Electron. Agriculture*, vol. 100, pp. 100–109, 2014.

[138] D. J. Greenwood et al., “Opportunities for improving irrigation efficiency with quantitative models, soul water sensors and wireless technologies,” *J. Agricultural Sci.*, vol. 148, pp. 1–16, 2010.

[139] M. Srbinovska et al., “Environmental parameters monitoring in prevision agriculture using wireless sensor networks,” *J. Cleaner Production*, vol. 88, pp. 297–307, 2015.

[140] G.-H. Kim, S. Trimi, and J.-H. Chung, “Big-data applications in the government sector,” *Commun. ACM*, vol. 57, no. 3, pp. 78–85, 2014.

[141] T. Fukatsu, T. Kiura, and M. Hirafuji, “A web-based sensor network system with distributed data processing approach via web application,” *Comput. Stand. Interfaces*, vol. 33, no. 6, pp. 565–573, 2011.

[142] R. W. Coates et al., “Wireless sensor network with irrigation valve control,” *Comput. Electron. Agriculture*, vol. 96, pp. 657–662, 2013.

[143] A. Matese et al., “A wireless sensor network for precision viticulture: The NAV system,” *Comput. Electron. Agriculture*, vol. 69, no. 1, pp. 51–58, 2009.

[144] A. G. Bhave, A. Mishra, and N. S. Raghuwanshi, “A combined bottom-up and top-down approach

for assessment of climate change and adaptation options," *J. Hydrology*, vol. 518, pp. 150–161, 2013.

[145] L. B. L. Goncalves et al., "Influence of mobility models in precision spray aided by wireless sensor networks," *J. Phys.*, vol. 574, no. 1, 2015.

[146] A. S. Vouldimos et al., "A complete farm management system based on animal identification using RFID technology," *Comput. Electron. Agriculture*, vol. 70, no. 2, pp. 380–388, 2010.

[147] K. H. Kwong et al., "Practical considerations for wireless sensors networks in cattle monitoring applications," *Comput. Electron. Agriculture*, vol. 81, pp. 33–44, 2012.

[148] M. Lin, Y. Wu, and I. Wassell, "Wireless sensor network: water distribution monitoring system," in *Proc. Radio Wireless Symp.*, 2008, pp. 775–778.

[149] H. Zia et al., "The impact of agricultural activities on water quality: A case for collaborative catchment-scale management using integrated wireless sensor networks," *Comput. Electron. Agriculture*, vol. 96, pp. 126–138, 2013.

[150] A. Malaver et al., "Development and integration of a solar powered unmanned aerial vehicle and a wireless sensor network to monitor greenhouse gases," *Sensors*, vol. 15, no. 2, pp. 4072–4096, 2015.

[151] S. Misra and S. Singh, "Localized policy-based target tracking using wireless sensor networks," *ACM Trans. Sensor Netw.*, vol. 8, no. 3, 2012, Art. no. 27.

[152] T. Wark et al., "Transforming agriculture through pervasive wireless sensor networks," *Pervs. Comput.*, vol. 6, no. 2, pp. 50–57, 2007.

[153] J.-A. Jiang et al., "Pest hot spot detection for the oriental fruit fly (Bactrocera dorsalis (Hendel)) via wireless sensors networks," in *Proc. Workshop Wireless Ad Hoc Sensor Netw.,* 2009.

[154] S. R. Huddar et al., "Novel algorithm for segmentation and automatic identification of pests on plants using image processing," in *Proc. Comput. Commun. Netw. Technol.*, 2012, pp. 1–5.

[155] J. Zhao, M. Liu, and M. Yao, "Study on image recognition of insect pest of sugarcane cotton aphis (SP) based on rough set and fuzzy C-means clustering," in *Proc. Symp. Intell. Inf. Technol. Appl.*, vol. 2, 2009, pp. 553–555.

[156] F. Faithpraise et al., "Automatic plant pest detection and recognition using k-means clustering algorithm and correspondence filters," *Int. J. Adv. Biotechnol. Res.*, vol. 4, no. 2, pp. 189–199, 2013.

[157] D. F. Silva et al., "Exploring low cost laser sensors to identify flying insect species," *J. Intell. Robot. Syst.*, vol. 80, no. 2015, pp. 313–330, Dec. 2015.

[158] Y. Chen et al., "Flying insect classification with inexpensive sensors," *J. Insect Behav.*, vol. 27, no. 5, pp. 657–677, Sep. 2014.

[159] M. Vuran and I. Akyildiz, "Cross-layer packet size optimization for wireless terrestrial, underwater, and underground sensor networks," in *Proc. Int. Conf. Comput. Commun.*, 2008, pp. 780–788.

[160] Aqeel-ur-Rehman et al., "A review of wireless sensors and networks' applications in agriculture," *Comput. Stand. Interfaces*, vol. 36, pp. 263–270, 2014.

Robert P. Dick is an Associate Professor of Electrical Engineering and Computer Science at the University of Michigan, Ann Arbor, MI, and co-founded the Stryd, Boulder, CO, wearable electronics company. His research focuses on embedded systems. Dick has a BS from Clarkson University, Potsdam, NY, and a PhD from Princeton University, Princeton, NJ. He is a Senior Member of the IEEE.

Li Shang is an Associate Professor at the Department of Electrical, Computer, and Energy Engineering, University of Colorado Boulder, Boulder, CO, and co-founded the Stryd, Boulder, wearable electronics company. Shang has a BE (Hons.) from Tsinghua University, Beijing, China, and a PhD from Princeton University, Princeton, NJ. He is a member of the IEEE.

Marilyn Wolf is a Professor and a Chair of the Department of Computer Science and Engineering, University of Nebraska–Lincoln, Lincoln, NE. Her research interests include cyber-physical systems, Internet-of-Things, embedded computing, embedded computer vision, and very large scale integration (VLSI) systems. Wolf has a BS, an MS, and a PhD in electrical engineering from Stanford University, Stanford, CA (1980, 1981, and 1984, respectively). She is a Fellow of the IEEE and ACM.

Shao-Wen Yang is a Senior Applied Scientist at Amazon, Bellevue, WA. Yang has a PhD in computer science, specialized in mapping and localization in robotics, from National Taiwan University, Taipei, Taiwan (2011).

■ Direct questions and comments about this article to Robert P. Dick, University of Michigan, Ann Arbor, MI, USA; dickrp@umich.edu.