# Smart City System Design: A Comprehensive Study of the Application and Data Planes

HADI HABIBZADEH, University at Albany, SUNY, USA
CEM KAPTAN, University of Ottawa, Canada
TOLGA SOYATA, University at Albany, SUNY, USA
BURAK KANTARCI and AZZEDINE BOUKERCHE, University of Ottawa, Canada



Recent global smart city efforts resemble the establishment of electricity networks when electricity was first invented, which meant the start of a new era to sell electricity as a utility. A century later, in the smart era, the network to deliver services goes far beyond a single entity like electricity. Supplemented by a well-established Internet infrastructure that can run an endless number of applications, abundant processing and storage capabilities of clouds, resilient edge computing, and sophisticated data analysis like machine learning and deep learning, an already-booming Internet of Things movement makes this new era far more exciting.

In this article, we present a multi-faceted survey of machine intelligence in modern implementations. We partition smart city infrastructure into application, sensing, communication, security, and data planes and put an emphasis on the data plane as the mainstay of computing and data storage. We investigate (i) a centralized and distributed implementation of data plane's physical infrastructure and (ii) a complementary application of data analytics, machine learning, deep learning, and data visualization to implement robust machine intelligence in a smart city software core. We finalize our article with pointers to open issues and challenges.

CCS Concepts: • **Networks** → **Cyber-physical networks**; **Layering**; **Network protocols**; **Wireless access networks**; • **Computer systems organization** → **Sensors and actuators**; **Cloud computing**; • **Computing methodologies** → **Machine learning**; • **Security and privacy**; • **Information systems** → *Information storage systems*; *Data management systems*; • **Human-centered computing** → *Visualization*;

Additional Key Words and Phrases: Crowdsensing, edge-computing, diagnostic, predictive, and prescriptive analytics, supervised learning, unsupervised learning, smart sustainable cities, deep learning, big data, cybersecurity, mobile computing, data science

**ACM Reference format:**
Hadi Habibzadeh, Cem Kaptan, Tolga Soyata, Burak Kantarci, and Azzedine Boukerche. 2019. Smart City System Design: A Comprehensive Study of the Application and Data Planes. *ACM Comput. Surv.* 52, 2, Article 41 (May 2019), 38 pages.
https://doi.org/10.1145/3309545

This work was supported in part by NSERC-CREATE-TRANSIT, Canada Research Chairs Program, NSERC-Strategic Project OpContent, the U.S. National Science Foundation grant CNS-1239423, and the Natural Sciences and Engineering Research Council of Canada (NSERC) DISCOVERY Program.

Authors' addresses: H. Habibzadeh, University at Albany, SUNY, Albany, NY, 12222, USA; email: hhabibzadeh@albany.edu; C. Kaptan, University of Ottawa, Ottawa, ON, K1N 6N5, Canada; email: ckapt046@uottawa.ca; T. Soyata, University at Albany, SUNY, Albany, NY, 12222, USA; email: tsoyata@albany.edu; B. Kantarci, University of Ottawa, Ottawa, ON, K1N 6N5, Canada; email: burak.kantarci@uottawa.ca; A. Boukerche, University of Ottawa, Ottawa, ON, K1N 6N5, Canada; email: aboukerc@uottawa.ca.


0360-0300/2019/05-ART41 $15.00
https://doi.org/10.1145/3309545

## 1 INTRODUCTION

While the original ideas behind smart cities go back to the late 1990s [41, 60, 92], the emergence of actual smart city application concepts and deployments—such as smart transportation [18], smart health [70], and smart lighting [123]—are fairly recent. This is primarily because smart city applications would be limited in scope without the technological advances that support the infrastructure that they need [42], such as the emergence of the 5G cellular technology [2, 104] and Internet-of-Things (IoT) sensors [5], both through dedicated sensors deployments [52] and the emerging mobile crowd-sensing paradigm [48].

Existing surveys in the literature study smart cities from specific angles. Guo et al. [45] analyze the smart city through the lens of mobile crowd-sensing and computing (MCSC); they emphasize a wider aspect of crowd-sensing (including participatory, opportunistic, and hybrid sensing, as well as mobile social networking) and focus on the integration of machine and human intelligence in this platform as a nascent yet promising solution. This approach creates a synergistic operation between the machines and humans; while machines can process the bulk of raw data and improve the decision-making process, humans *supervise* the sensing and computing operations of the machines. By thoroughly investigating the literature, Pejovic et al. [105] provide recommendations regarding the implementation of smartphone-oriented anticipatory platforms, which collect data from a variety of sources (e.g., sensors and the Internet) to perceive the *context* and apply machine intelligence to predict the future outcome. Based on this model, the device can make recommendations to the user depending on the applications (e.g., recommending movies and multimedia context, recreational activity, etc.). Siow et al. [127] study the recent development in data analytics in the context of IoT and Big Data. They investigate the domain, objectives, required resources, and available frameworks (cloud-based and centralized) for implementation of data-analytic techniques. A recent survey on edge computing is provided by Li et al. [79], which approaches the edge computing from an architectural perspective, where the authors evaluate the main components and address resource and task management considerations.

The survey conducted in this article does not intend to replace or update the preceding works. Instead, it aims to provide an examination of the most recent developments in the software core (*data plane*) of smart cities, seen from a data science–oriented perspective. From this standpoint, this article is more in line with the work of Siow et al. [127]. Nonetheless, as opposed to underlining the application of data analytics to Big Data, this article studies the applicability of specific existing data processing solutions to given applications and services; this approach is aimed to help researchers and developers select the most compliant alternatives for their intended application. To this end, this manuscript views smart city applications from multiple angles.

Smart city applications occupy the *application plane* and each application requires data input that is obtained from the *sensing plane*. The sensed data are transmitted to its destination (where it is processed) through the *communication plane*. The *data plane* is where sophisticated processing takes place through data analytic and machine intelligence. An overarching *security plane* is responsible for the privacy and security of the data that exist in all of the previous planes. We term each one of these viewpoints a *plane*, as shown in Figure 1, and provide an extensive study of the application and data planes.

The remainder of this article is organized as follows. In Section 2, we introduce the overarching architecture of smart city applications, consisting of five planes. In Section 3, we provide a survey of well-established smart city applications, which form the application plane. In Sections 4 and 5, we study the data plane. We provide a set of open issues related to the data-plane aspects in Section 6 and conclude our article in Section 7.

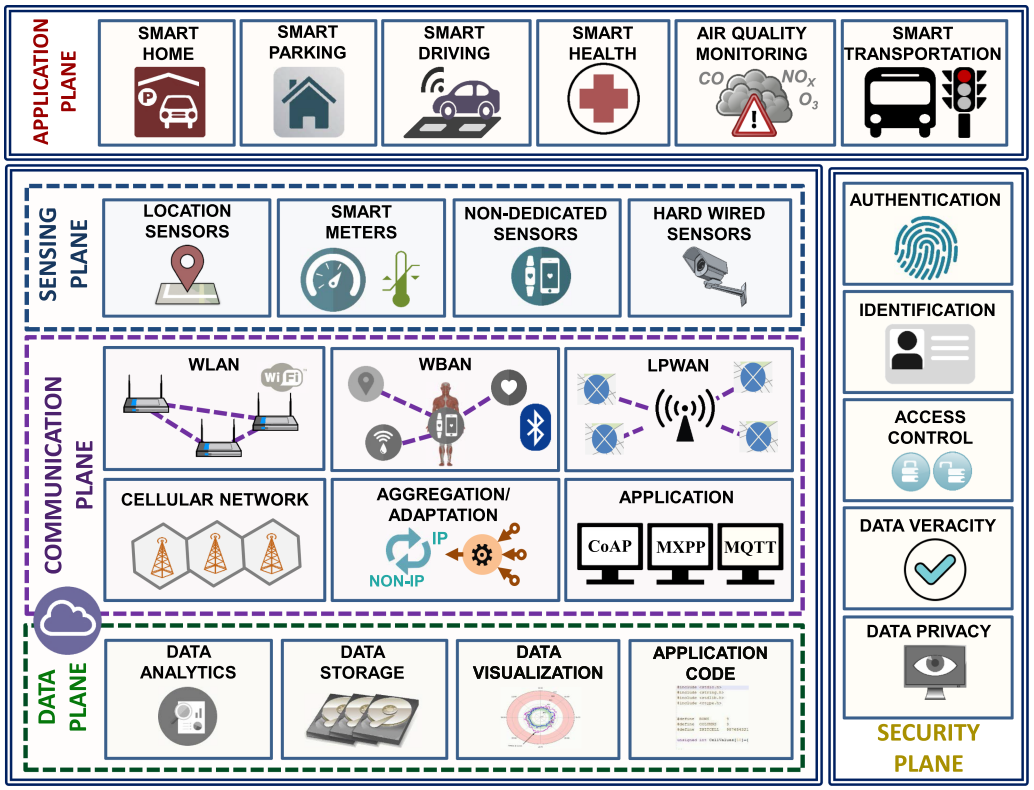


Fig. 1. An abstraction of smart city architecture encompassing five planes: (i) application plane, (ii) sensing plane, (iii) communication plane, (iv) data plane, and (v) security plane (Hadi Habibzadeh).

## 2 SMART CITY SYSTEM ARCHITECTURE

Smart city concept embodies a wide range of services, including smart transportation, smart healthcare, smart clothing, smart grid, and smart utilities, to name a few. Although these applications differ significantly in terms of their requirements, almost all of them can be studied under a standard operational architecture, consisting of five distinct planes: (i) application, (ii) sensing, (iii) communication, (iv) data, and (v) security planes [47]. As depicted in Figure 1, the application plane and security plane are connected to the other three planes, because not only every application needs the other four planes, but also the responsibility of the security plane is to protect the other four planes from outside attacks. The rest of this section discusses individual planes of smart city architecture. In Section 3, we investigate a selected set of smart city applications from the standpoint of sensing, communication, and applications. However, the main focus of this article remains limited to the data plane, which is studied in depth in Section 4 and Section 5.

### 2.1 Application Plane

The application plane is the interface between a smart city and its users (mostly its residents). Applications aim to reduce the city's expenses by controlling resource misuse, promoting task automation, and improving the safety and security of the city through ubiquitous and continuous monitoring. These objectives are fulfilled by establishing a link between the users and the data

plane either *directly* (e.g., an online portal or a smartphone application that visualizes information) or *indirectly* (e.g., through actuators that control the residents' living environment).

Interactions between the users and the data plane bring about multiple technical challenges. Interoperability is one such challenge, particularly considering that traditional smart city applications such smart transportation, smart home, and smart healthcare have started to blend into homogeneous and integrative services. For example, the prevalence of Electric Vehicles (EVs) has entangled smart grid and smart transportation applications [118, 147]; furthermore, interactions between smart healthcare and smart transportation (for example, to decrease response time of emergency units [129]) commingle the latter two with smart grid implementations, creating a *uniform IoT ecosystem*. Managing the synergy among different applications, which are designed for different objectives, is proven orders of magnitude more difficult than developing each application as a stand-alone service. In addition to technical challenges, social aspects of the application plane are also an open research question, because, in contrast to other planes of the smart city framework, applications are in direct contact with users. Examples of such social concerns include social acceptance and compliance with existing security and safety regulations imposed by city authorities.

### 2.2 Sensing Plane

A typical sensing plane incorporates a wide variety of sensing devices and actuators to measure physical signals (e.g., environmental irradiation) and interact with *things* (e.g., city lights). Implementation of the sensing plane in smart cities resembles traditional Wireless Sensor Networks (WSNs) and is subject to similar limitations. Most noticeably, meager power availability remains a major impeding challenge. However, unlike WSNs, this limitation is not due to the scarcity of power distribution infrastructure; instead, it is an implication of the plane's large scale. Self-power-harvesting (such as solar and wind energy harvesting [49, 50]) is typically the most viable solution to circumvent excessive installation and maintenance costs (*recurring* and *non-recurring* expenses [52]).

Many smart city applications involve monitoring a wide variety of physical parameters with different accuracy, precision, and sensitivity requirements. This *heterogeneity* is considered to be a key differentiating feature between smart city sensing and WSNs, which necessitates more sophisticated data acquisition, aggregation, and processing algorithms to be executed using resource-constrained devices that are already burdened by a *tsunami* of information [138]. In-field implementation of the sensing plane, its constant interactions with its users and the environment, and the mobility of its many constituents also imply that the plane must be flexible and expandable. At the device level, this translates to additional restrictions that ensure non-invasiveness, environmental friendliness, and safety concerns are robustly addressed. In many cases, flexibility and expandability features, such as over-the-air firmware updates, can introduce additional privacy and security vulnerabilities [10].

Intuitively, a universal solution that can meet all these requirements is nonexistent; tradeoffs must be made according to priorities of the target applications. Before delving into the details, designers must choose between *dedicated* and *non-dedicated* sensing architectures [48, 52]. Traditional *dedicated* sensing approach—where city administrators have direct control over the operation of a set of sensors with pre-defined and fixed sensing objectives—arms the platform's supervisors with a greater flexibility to re-architect the sensing plane based on their assessment of constantly changing requirements. On the downside, dedicated sensing leaves the city administration as the sole entity liable for all *recurring* and *non-recurring* expenses, thereby practically limiting the scale and flexibility of the system. In *non-dedicated* sensing, general-purpose sensors built into portable devices (e.g. smartwatches or smartphones) are used to reduce the overall expenses of the system (especially recurring). The efficacy of the system, however, hinges on the

cooperation of its volunteers, who must be incentivized to share their resources [154]. In the literature, non-dedicated sensing is often referred to as *crowd sensing* [38].

### 2.3 Communication Plane

The communication plane *pre-processes* and *aggregates* the data acquired from the sensing plane and delivers them to the other layers, effectively connecting the field devices to the cloud. Ideally, this link is expected to support low-latency, high-throughput, flexible, and secure communication. However, a multitude of challenges impedes the feasibility of such a conceptual communication service. This compels developers to make tradeoffs according to the requirements of their applications.

Inherited from WSNs (the ancestor of IoT), communication capabilities are mainly hampered by scant power availability. Effects of this scarcity are more pronounced in the communication plane (as opposed to the data plane), because communication circuitry consumes orders of magnitude more energy than sensing. Although ambient energy harvesting solutions can alleviate this negative effect in certain cases, the most practical solution is to make tradeoffs among the data rate, latency, and transmission range to prolong the battery life of sensing devices (which host the front-end circuitry for communication).

The harsh environment of cities—where urban canyons and interference with existing communication infrastructure are inevitable—further complicates the implementation of this plane. Communication protocols utilize techniques such as frequency hopping to curtail these adverse effects to some extent. Nonetheless, interference remains an insurmountable problem, especially in dense networks [77]. Because no universal implementation of the communication plane exists for IoT applications, existing networks typically include an amalgamation of various standards. Clearly, guaranteeing interoperability in such a *heterogeneous* network is not a trivial task.

*Data transmission* in the communication plane is typically initiated by the request of on-node communication modules to send the data in the aggregated format to the *cloud*. In some implementations, download capability is also provisioned to support over-the-air command and control. Although wired connectivity can provide high-speed and reliable communication, an overwhelming majority of sensing devices use wireless modules to support scalability and mobility. Due to the aforementioned limitations in the communication plane, however, communication modules cannot support a direct and reliable connection to the cloud. Instead, this requirement is satisfied through a hierarchical implementation; on-node communication modules transmit the data to a local gateway or an Access Point (AP) through either a single-hop or a multi-hop architecture. Gateways are moderately computationally capable devices, which can support cloud connectivity by providing TCP/IP adaptation services. In dedicated sensing, gateways are typically implemented by application specific routers, whereas in non-dedicated sensing, users' smartphones can serve as gateways. Single-hop and multi-hop architectures differ in terms of power consumption, resistance against interference, latency, congestion, and so on.

*Data aggregation* is a complementary service in the communication plane, which aims to improve communication quality and increase the longevity of battery-powered devices by reducing communication overhead. Substituting communication with computation is the fundamental driving force behind data aggregation. Since the former dissipates more power the latter, this practice can result in a drastic increase in battery life. In this approach, selected devices (typically APs and cloudlets) perform rudimentary data processing to eliminate erroneous, duplicate, and redundant data before forwarding the information to the cloud. Data *pre-processing* algorithms can also be applied for feature extraction [101] and noise suppression to complement data aggregation and further eliminate redundancy in the early stages. A complete and recent survey on communication plane of the smart city is provided in Reference [47].

### 2.4 Data Plane

A data plane is the convergence point of the acquired data, where the bulk of seemingly incoherent data are converted into meaningful information. These processing techniques often involve advanced algorithms that require highly computationally capable hosts. The vast extent of the IoT along with limited delay tolerance of many smart city applications (e.g. smart grid, transportation, and health services) further emphasizes the significance of computational power. No single node, in the data and communication planes, is capable of satisfying this requirement. However, it is possible to combine their resources as a single abstract computation infrastructure (e.g., a cluster), in which every node executes a portion of an algorithm. This *distributed* (or *decentralized*) implementation can significantly reduce recurring and non-recurring expenses by recycling existing available resources. Clearly, the efficacy of the solution rests on the algorithm and its potential for distributed execution. As an alternative approach, a data plane can be *centralized* in physically separated, cloud-based servers, which provide ample processing and communication capability. Although more expensive, cloud-based servers are typically more resilient and are capable of executing conventional off-the-shelf data processing algorithms.

Regardless of its implementation, the data plane must provide two fundamental services to the other planes: (i) data processing and (ii) data storage. *Data processing* is the software core that involves utilization of processing hardware (e.g., CPUs and GPUs) to run a wide array of algorithms, while *data storage* functionality centers around the collection, storage, and creation of databases for both raw data and processed information. Data storage is critical to the accuracy of various smart city applications, as past trends can provide a basis for veracity evaluation of newly acquired data. Because of its large scale, long-term data storage is typically associated with big data science and advanced database management tools.

To provide these two fundamental services, the data plane incorporates three algorithmic modules: (i) *Data-analytics* techniques extract useful information from a large volume of raw data samples. Cluster analysis, correlation, and regression are prominent examples of such methods. (ii) *Machine-learning* (*ML*) algorithms have been recently used as a viable alternative to predict future events based on past data. ML solutions are gaining popularity in smart city applications due to their remarkable accuracy and superior scalability. (iii) *Data visualization* techniques are required to effectively present the results of data analytics and machine learning to users. Visualization, however, requires careful implementation; for example, cramming excessive information into a graph can cause a distraction, while oversimplifying data might lead to the loss of critical details.

### 2.5 Security Plane

Regardless of many breakthroughs in IoT, the field is still considered to be in its infancy; a multitude of immature services are introduced merely to experiment with the feasibility of various ideas and their acceptance by the users. In such a climate, security and privacy considerations are often neglected. Recently, however, an increasing number of sophisticated cyberattacks in the IoT arena [17, 84] have raised consciousness about the importance of security and privacy in the IoT sphere. While most of these attacks are aimed to extort money at a smaller scale, there are incessant concerns about the possibility of large-scale attacks targeting critical infrastructures, which can cripple the economy and endanger the lives of many smart city residents.

The difficulty of ensuring security and privacy varies among the planes of the smart city architecture. For example, off-the-shelf encryption can secure communication plane even in the presence of meager power availability limitations. On the contrary, providing device-level security against hardware and software attacks requires more customized solutions. The expediency of security solutions in communication technology oftentimes results in a disparity in security of

networks vs. devices and data (while the latter two are typically overlooked). However, as shown in Figure 2, an effective security plane must uniformly protect all components of the system; otherwise, vulnerabilities in a weak link can endanger the entire system.

Eavesdropping, man-in-the-middle (or equivalently manipulation), spoofing, DDoS attacks, and data leakage are widespread concerns for a smart city. Conventional cryptographic solutions such as Advanced Encryption Standard (AES), Elliptic Curve Cryptography (ECC), and RSA can provide immunity against many of these attacks. Less conventional solutions such as Montgomery's multiplication [72] are needed to provide protection against variants of side-channel attacks [74]. Aside from these ordinary threats, some vulnerabilities and challenges arise from the exclusive characteristics of IoT. The large scale, mobility, and dynamic nature of smart cities make it more complicated to identify and authenticate participants, without violating their anonymity. Particularly, access control has proven to be challenging, as the access privileges of participants change dynamically, depending on the context.

## 3 APPLICATION PLANE FROM PERSPECTIVE OF SENSING AND COMMUNICATION

As depicted in Figure 1, applications operate atop a three-component framework. Raw data captured by smart city sensors travels through the network to reach the data plane, where processing and storage take place. The output of the latter is used within the interpretation of an application to ensure safety and security of the city and its citizens, as well as improve life quality of residents. Aside from data processing, which we discuss in Section 4 and Section 5, sensing and communication are the most integral constituents of any smart city application. In this section, we investigate several smart city applications and discuss their utilization of available sensing and communicating technologies.

### 3.1 Smart Environments

A typical smart environment application embodies a broad physical setting that is equipped with an overwhelming number of dedicated and non-dedicated sensors, actuators, displays, and components with processing power. These components are seamlessly integrated with everyday objects and inter-connected through a ubiquitous network platform. Autonomy, adaptability, and effective user interaction are the key requirements to achieve smart environments [44].

Additionally, ensuring interoperability among main enablers of smart environment communication (WSNs, RFID tags, and mobile networks) appears as a major challenge for many applications. The authors of Reference [21] propose an interoperability framework that utilizes the Constrained Application Protocol (CoAP) over an infrastructure of IPv6 low-power wireless personal area network (6LoWPAN) and representational state transfer (REST) to facilitate communication among these complementary technologies. An interesting application in Reference [71] formulates a classroom (termed *Smart Classroom*) in which a set of microphones and cameras acquire a presenter's (either a student or a teacher) audio and video. These data are pre-processed and input into emotion recognition algorithms.

### 3.2 Smart Homes and Buildings

Smart homes and buildings are two important constituents of smart cities. Smart homes provide control capabilities to their users to create a comfortable living environment at home. Smart buildings can be conceptualized at two different levels: (i) a *physical level*, which includes a community of smart buildings that have wired and wireless networking capabilities and are integrated with the power grid and the transportation system, and (ii) a *virtual level*, which refers to information sharing, collaboration, and inter-operation among people and utilities in a community. For example, the smart building developed in Referencee [78] aims to monitor concrete corrosion

using RFID tags and other sensors. Data correctness is evaluated by RFID CRC check and a collision detection mechanism.

Smart home energy management is another area under smart homes. In Reference [29], authors introduce a Bluetooth Low Energy–(BLE) based system for smart home energy management, which aims to reduce the electricity bill by turning off certain devices during peak usage hours. The system includes several sensors, such as temperature, humidity, light, and presence, to control the HVAC operation. Furthermore, smoke detectors activate alarms in the case of hazardous gas detection in the air (e.g., carbon monoxide). For the wireless access technology, the authors choose BLE over 802.15.4 due to its compatibility with many laptops and smartphones.

### 3.3 Smart Surveillance

Gradual—yet constant—reduction in sensor energy consumption, which is fueled by the evolution of sensing technologies, has paved the way for the implementation of a diverse set of surveillance services.

In Reference [11], the authors introduce an application that tracks a crowd of users in a high-density location using inexpensive coin battery-powered BLE-based tags, which operate up to a year without requiring a battery replacement. The system requires each user to wear a tag. A subset of the users can interrogate these tags using their smartphones and transmit the data into the cloud for processing (via either cellular networks or WiFi). Another example of a smart surveillance system is presented in Reference [139], which also employs BLE technology. This study aims to detect clusters of people based on their proximity, facilitating the identification of individuals who spend time together. WiFi and BLE signals are used as an indicator of distance between individuals.

### 3.4 Smart Transportation and Driving

Smart transportation and driving involves equipping vehicles with sensing, communication, computing, and processing capabilities as a means to improve safety, efficiency, and service quality for both drivers and passengers. An intra-vehicular wireless sensor network (IVWSN) can reduce vehicle weights by avoiding excessive wiring [115]. An IVSN can operate with BLE or ZigBee and is required to meet the following objectives: acceptable transmission rate, low delay, stationary sensors, and robustness to obstacles in the deployed environment. Through a feasibility study on possible wireless access technologies for an IVWSN, BLE has been shown to be cost effective. It also has a higher data rate than other technologies, boasts low delay, and is compatible with portable devices.

### 3.5 Smart Health

Widespread use of sensors and actuators, as well as the coupling of sensory data with analytics, leads to a smarter way of offering healthcare services to improve quality of life and ensure healthier communities for sustainable cities [58]. Existing smart health applications span various medical, social, and behavioral fields. A typical smart health application area is human gait activity recognition, which is crucial for monitoring of the orthopedic health of individuals, especially the elderly. The authors of Reference [85] propose a system to recognize human gait activity by incorporating 48 embedded pressure sensors, accelerometers, a gyroscope, and a magnetometer in an insolelike mechanism. Sensors help with the measurement of step count, center of pressure, swing time, and shifting velocity. Captured data are transferred via BLE to a smartphone for processing and visualization.

Privacy and security of the medical information are major concerns for Smart Health. Studies in References [8, 73] aim at formulating a medical cloud computing environment for cardiac health

monitoring, in which the medical data are transmitted from the sensing plane to the data plane in a privacy-preserving fashion.

### 3.6 Smart Lighting

Inherent characteristics of LED-based light resources allow manipulation of spectral power and spatial distribution, color temperature, temporal modulation, and polarization, which lend themselves well to many smart city applications. An example of the state of the art in smart lighting is the use of smart road signs that flash to raise alerts for the drivers about potential dangers on their way. Adaptively controlling the intensity of light based on occupancy is also a popular application area of smart lighting. The authors of Reference [140] propose an evolutionary algorithm to optimize the intensity of lights as human occupancy changes in a museum.

Another enabling technology for smart lighting is Visible Light Communication (VLC). While various multiple access solutions can be considered/utilized or tailored for VLC networks that are equipped with inherently directional sources, the IEEE 802.15.7 standard defines protocols for the physical and medium access layers for VLC networks [34]. The authors of Reference [123] employ a smart lighting technology that utilizes LEDs to attain high-performance and low-cost wireless communication. They suggest the integration of free-space-optical (FSO) communication via smart lighting techniques.

### 3.7 Smart Parking

In urban settings, smart parking systems have emerged to reduce negative environmental and financial impacts of parking [75]. To discover available spots in a parking space, smart parking systems commonly make use of either *on-road sensors* (e.g., Magnetometers and RFID tags) or light sensors and cameras that are known as off-road sensors. Generally, a cloud-based reservation framework is a part of the whole system to assign parking spaces to drivers. A comprehensive survey of smart parking solutions between the years 2000 and 2017 is presented in Reference [86] along with enabling technologies, communication and computing challenges, and proposed solutions. The study in Reference [75] presents smart parking solutions from the standpoint of monitoring techniques for surveillance and data acquisition and algorithmic solutions for parking reservation.

### 3.8 Smart Grid

A smart grid denotes a power grid infrastructure that utilizes wired/wireless networks and computing/storage resources to collect and process data about the grid. The gathered data can present produced and consumed energy, operational characteristics of distribution lines, or the availability of various electricity resources. The data are used to generate real-time models, which help sustain an optimal state in the grid [44].

In Reference [50], a smart city overlay network is investigated that consists of self-energy-harvesting *smart boxes*. These boxes incorporate limited communication capability in the case of a catastrophic event such as a natural disaster. Due to their reliance on solar and wind energy harvesting [49, 51] and long-lifetime supercapacitor-based energy buffering [35, 96], the proposed smart boxes can function entirely independent of the grid. The utilization of very low maintenance supercapacitor-based energy buffering [59] allows these boxes to form a citywide emergency network with virtually zero recurring maintenance expenses.

## 4 DATA PLANE

The *data plane* (i) employs its computational resources to apply sophisticated data processing algorithms on the acquired data, (ii) utilizes advanced *visualization* techniques to condense and

summarize the outcomes of these computations, and (iii) provides *data storage* for both raw and processed information. This section details the requirements of the data plane for smart city applications (Section 4.1), elaborates on the data-plane architectures (Section 4.2), and provides a review of the storage architectures (Section 4.3).

### 4.1 Requirements

The data plane takes the raw data as the input and extracts *useful* information from it. It presents the extracted information to the end users in a summarized visualization framework. In some applications, a set of actuators are used to utilize the data (e.g., dimming the street lights) rather than visualization. Regardless of the implementation or target application, smart city data plane must be able to overcome the challenges concerning the five Vs of the big data concept [48]: (i) Volume, (ii) Velocity, (iii) Variety, (iv) Veracity, and (v) Value.

*Volume* is a direct implication of the vast amount of data collected in smart city applications, which could render traditional databases ineffective [66]. Furthermore, applying sophisticated processing algorithms, visualization techniques, and storage services to such a high volume of data may not be feasible using conventional servers.

*Velocity* is associated with the throughput of the incoming data; the streaming of the data can either be real time or non-real-time. Therefore, data velocity is closely related to data volume. Indeed, the communication plane introduces uncertainties to the time domain, which further complicate the problem.

*Variety* indicates data-level heterogeneity. This heterogeneity not only concerns the architecture and format of the data (e.g., images, videos, audio, text, etc.) but also evinces itself in volume and velocity, because each data type non-uniformly contributes to the size and timing requirements of the data plane. For instance, the contribution of the video-based sensors to data *volume* is not comparable to that of text-based traffic.

*Veracity* quantifies the trustworthiness of data [112] or alternatively the uncertainty in the data. Quantifying the uncertainty in the incoming data is one of the important tasks of the data plane. In non-dedicated sensing scenarios, the cause of the uncertainty can be related to intentional manipulation by users [111] or the malfunctioning sensors [112]. Although veracity evaluation at a small scale is straightforward, ad hoc participation, ubiquity, and rapid connectivity in sensing and communication planes substantially complicate veracity evaluation. The unreliable communication links of wireless mediums further increase the uncertainty.

*Value* is used to quantify the importance and usefulness of raw data prior to their fusion. The higher the former four Vs get, the more challenging it becomes to assess data value.

### 4.2 Architectures

The IoT-Smart City ecosystems are composed of a plethora of sensors that generate a massive amount of heterogeneous and mostly unstructured data. To handle the sensed data, reliable and scalable storage and processing systems are necessary. Considering that traditional centralized and large-scale data storage infrastructures suffer from poor scalability and high operational costs, the following two architectures emerge as alternative solutions: (i) Cloud-based architecture and (ii) edge-based architecture.

*4.2.1 Cloud-based Architecture.* Cloud-based architectures utilize cloud computing to enable ubiquitous, on-demand, and broadband access to a shared pool of resources that are re-configurable. Resource pool in a cloud-based architecture may include network equipment, computer servers, disks, and storage equipment and applications. Cloud-centric applications take advantage of auto-scaling and metered service features of cloud computing. As auto-scaling enables

adjustment of the type and amount of allocated resources based on the load, it helps improve the sustainability of smart city applications by introducing adaptive sleep or switch-off mechanisms for servers. By combining auto-scaling feature with metered services, cloud-centric solutions further reduce the operational expenses, as a cloud provider charges for applications, platforms, or hardware on a pay-per-use basis instead of maximum capacity. Advanced virtualization and multi-tenancy features of a cloud-based architecture offer a pool of virtual resources that can be rapidly assigned and released based on the needs of a smart city application. As we have discussed so far, service-oriented and service-managed nature of the cloud improves the

- *visibility*, which helps Information Technology (IT) officers of the smart city application respond faster with better decisions;
- *control*, which helps with managing risks, monitoring the usage of shared resources, and reducing costs; and
- *automation*, which helps reduce the cost and build agility into the operations of smart city applications. Leveraging automation also addresses the operational complexity of managing virtual resources.

*4.2.2 Edge-based Architecture.* Considering the rapid growth in the data volume, effective cloud infrastructures are expected to self-accommodate and react to the rising demand without any intervention from users. Moreover, cloud solutions are preferably expected to be self-correcting to minimize the failure probability. Furthermore, the latency between sensing devices and the cloud poses an additional challenge. The transmission latency is a function of the physical and virtual (hop count) distance between the user and cloud. It increases the response time and degrades the quality of service perceived by the end users.

Edge-based architectures perform *edge computing* by outsoursing their workload to computational devices (or servers) that are relatively closer to the *field*. This reduces the load on the servers and eventually makes the applications that require low-latency response feasible [124]. As an example of edge computing, Dell EMC Micro Modular Data Center (Micro MDC) devices are equipped with storage, computing, networking, and power and cooling resources tailored to the needs of smart city applications. Micro MDCs can be stationed indoors, outdoors, at the base of a local cell tower, and virtually anywhere in the world. They can be administered remotely from anywhere, offering ultimate flexibility to the manager of smart city applications while being managed as a unified software-defined environment.

### 4.3 Storage and Processing

Utilizing conventional data centers for smart city applications creates scalability issues when there is a need for additional storage; furthermore, the location of the data center creates latency concerns for delay-sensitive applications, which can be alleviated partially by edge processing. This section investigates storage and processing solutions available in cloud and edge-based architectures.

*4.3.1 Cloud Storage and Processing.* In Reference [125], the authors address large-scale storage and decision-making (processing) challenges through a distributed architecture. Their solution offers a cloud data storage and management platform, which is based on a wireless mesh network that consists of a two-tier network and supports hierarchical scheduling and multi-level decision-making. In the first tier of the system, real-time data acquired from smart city sensors is stored in wireless mesh access points (APs) in a distributed manner; in the second-tier, long-term and historical information of the system is stored. Each mesh AP broadcasts data for the network and has an internal storage capacity to construct an efficient private data storage network. While the

APs are responsible for data delivery, gateways on the edge are responsible for collecting, storing, and pre-processing the data. Collected data are categorized into the following types:

- *Type-1* data are stored in the central data storage system, which serves as a global and historical warehouse.
- *Type-2* data are real time, recurrent, and time-critical; they are stored in mesh APs.

Since processing data and making decisions by accessing the central decision server is time-consuming, in the proposed solution, these operations are performed by accessing regional mesh APs. The benefits of this mesh cloud data storage and management platform can be listed as dynamic cloud topology, improved throughput by making the data available in APs (rather than at data storage center only), data storage transparency, analyzing and exchanging data between different locations, reduced cost of data communications thanks to the distributed mesh APs in each geographic location, *on-the-fly scaling*, and easy-access-to-information in a massive system.

A majority of the sensed data volume is from multimedia streams generated by the video cameras around the city. The authors of Reference [32] propose an open source Sensor Observation Services (SOS) framework to handle such multimedia data using a distributed cloud storage system. Integrating cloud-based storage into this system brings benefits of reliability, scalability, and cost-effectiveness. Elastic scaling can be ensured in the case of unpredicted events, such as natural disasters, which could generate large amounts of data. To determine which cloud storage system is suitable for a given application, the following metrics can be used: Data storage technique (e.g., row-column, key-value, documents), distribution of data throughout the network (e.g., load distribution), fault tolerance strategy (e.g., replication, redundancy), and I/O performance of the data storage system. In the same study, a hybrid cloud storage system is deployed on Amazon EC2 with video feeds using the YCSB tool [30] that are collected from a smart city surveillance system. The proposed cloud storage system uses OpenStack Swift Object [37] and MongoDB [64] for media files. A feasibility study concludes the following: MongoDB is suitable for heavy applications like video surveillance, whereas OpenStack Swift Object scales better while performing slightly poorer than MongoDB.

When monitoring critical infrastructures such as the power grid, a data-plane architecture tailored for large-scale and real-time data analysis is of paramount importance [152]. Since cloud storage of a power grid system collects data from millions of sensors, the architecture is expected to be capable of running large-scale data analysis in real time, as well as on historical data. Furthermore, maintaining continuous operation is essential to meet reliability requirements. By employing replication techniques, the system can handle tolerable failures without causing any service interruption. Each replica maintains the consistency of the power grid data. For performance improvement, data reduction is adopted by removing the redundant data from the cloud. However, data reduction introduces a tradeoff between accuracy and performance. Therefore, the use of data reduction algorithms can only be considered for power utility applications that are tolerant of some accuracy loss [152].

Because cloud storage is prone to failures, thorough performance analysis is required prior to the cloud deployment for storage services [53]. In the presence of an incident causing a server failure, a backup storage system must become immediately available either locally or remotely to undertake the workload of the failed server. To ensure maximum availability, data are replicated into multiple servers. Conventionally, data have three replications that are stored as complete files in the servers of a data center [128]. To solve the problems that arise from substantial storage requirements, *erasure codes* provide a viable solution by reducing the storage cost while at the same time maintaining redundant data for robustness [109]. Operational characteristics of erasure codes are as follows: (i) mathematically encode the data into blocks, (ii) store these blocks at different

locations within the data center, and (iii) decode blocks when necessary. A data block produced by an encoder function is composed of original data bits and a parity block that contains parity bits. A complete set of data bits and parity blocks that is shaped using a single erasure coding technique makes a stripe, the size of which varies between 32KB and 1024K. Blocks are stored at different servers within the data center but in some cases, multiple blocks could be consolidated into one server. During the reconstruction process of the data, the blocks scattered to be stored in various locations across the data center are gathered by the decoder function. After collecting all parts that are required to reconstruct the data, the decoder starts constructing the source file.

*4.3.2 Storage and Processing at the Mobile Edge.* Edge devices can be used to implement distributed storage with the principles of Peer-to-Peer (P2P) computing by using smartphones, personal computers, set-top boxes, modems, and storage area networks. The motivation behind edge storage is to keep the data relatively close to the user, increase its availability, and reduce access latency. In Reference [97], an optimization framework is presented with a joint objective function to minimize system level energy consumption and maximize user satisfaction. To this end, real-time Quality of Service (QoS)-aware scheduling is an open problem that needs to be addressed. Store Cloud Using Edge Devices (STACEE) is a P2P distributed platform that is realized in a participatory manner. Participants partially share their processing power, disk storage, and network bandwidth without the need for any intermediaries. This allows the cloud network to expand dynamically by ad hoc addition of edge devices. Despite the benefits that make edge-based P2P cloud storage attractive, the following limitations remain: (i) system-level instability due to high churn rate of peers, (ii) edge device-level stability, and (iii) non-uniform device properties, such as energy consumption, the storage capacity of devices, and so on.

Offering effective incentives is essential to improve user participation in the resource sharing for a P2P mobile cloud. Moreover, compliance is another open issue to be addressed prior to resource allocation, since there are some restrictions including heterogeneous resources, delay/time sensitive patterns of supply and demand, as well as strategic actions by peers [16, 54, 122, 134]. The resources from the users in a P2P cloud act as public property. However, there are problems with hidden action, hidden intention, and free rider issues. In the case of hidden action, storage requester (user) is not in charge of controlling the actions of the resource provider and has no information about their intentions. This issue appears to be rather crucial when integrating security and encryption related solutions into the STACEE system. The free rider problem denotes the use of resources with making any payment. P2P networks handle this problem to some extent but abusive use can result in communication failures for reliable resources. Reputation-based service provisioning may alleviate this problem. For an edge storage system to be viable, the following must be ensured for its participating edge devices: (i) contribution, (ii) long-term predictable availability, and (iii) reputation and contractual agreement-based reliability.

Fog computing inherits heavily from cloud-based systems. A fog server is a generic entity that is virtualized and equipped with computing, communication, and storage functionality [87]. fog servers integrated into a specific service area predict the demands of mobile users, which can be either fetched from the cloud or uploaded by their owner and pre-cache the most required contents based on the information gained during prediction. As an example use case, fog servers in an airport can pre-cache information regarding flights and ground transportation based on the demand of mobile users. Therefore, the key issues in fog computing are forecasting user demand and determining the contents in a proactive manner.

The Content Delivery Network (CDN) [106] works based on fog computing concepts. It deploys cache servers at the edge of the backbone network to reduce the download latency. On one hand, CDN primarily targets Internet users with broad (and hence hard-to-predict) interests and service demands. On the other hand, a fog server is placed at a specific region to target users that share

specific service demands. Another service that is similar to a fog server is the Information Centric Network (ICN) [3]. This wireless cache infrastructure utilizes distributed cache services to provide content distribution services to users. The difference between the cache servers in ICN and fog servers is that the latter is equipped with intelligent computing functionality, whereas the former is solely employed for caching purposes. Thus, fog nodes can handle real-time data produced by mobile users and devices and can be a part of the cloud to empower scalable computing for rich applications such as vehicular communications and smart grid applications [13].

The authors in Reference [133] expand the application scope of fog computing that is introduced by Cisco. They include decentralized smart building control along with cloudlets under a software defined network (SDN) [80] scenario to support applications such as smart grid, connected vehicles, and wireless sensor and actuator networks. Considering the storage, computation, and communication resources of fog nodes, all of these applications are compatible with temporary and semi-permanent storage at the lowest and highest tiers, respectively.

Many smart city application require in-edge and real-time data analytics as a service. This translates to a requirement for computational power and an ability to store real-time sensor recordings. In Reference [88], the authors propose a methodology to store the data at the edge efficiently, based on the observation that most of the existing data produced by IoT end devices flow from an end device to a data center for storage and analysis. The latency between the data center and the end device needs to be minimized. Presently, this dataflow introduces high latency, since data centers are not close to the field. To process real-time data to provide near real-time accurate decisions at the *edge*, edge devices must locally store some historical data. However, edge devices are not equipped with a storage capacity to handle a workload that is usually managed by a data center. Hence, there exists a tradeoff between the amount of the data stored and the accuracy of near real-time decisions.

In Reference [88], the authors present a three-layer architecture to manage data storage in edge devices and introduce an adaptive algorithm to evaluate the aforementioned tradeoff. By locating the data-analytics engine near stored data, the amount of data that traverses the network can be minimized, which will eventually reduce the latency and overall cost [151]. However, storing data collected by devices in a smart city at the edge introduces some problems [135]. For accurate data analytics, a significant amount of historical data is required, which introduces a demand for high storage capacity at the edge [1]. The proposed three-layer architecture provides flow, analysis, and storage of data by taking into account the limited storage capacity at the edge.

The study conducted in Reference [149] introduces Algorithm of Collaborative Mobile Edge Storage (ACMES). A typical edge device cannot handle video or audio streams effectively, as they require a large storage space. With this motivation, ACMES lays the foundation of a framework to integrate distributed resources of edge devices, such as portable sensors and vehicle equipment. Aside from avoiding the backbone network latency, this approach ultimately reduces the workload at data centers. Indeed, in the implementation of such an effective storage solution, certain limitations of the edge devices need to be taken into account. These include battery limitation, limited storage capacity, withdrawal risk of a node, and system reliability. Parallel and distributed operation of the algorithm reduces the response time while lowering the probability of service outages due to a single-point failure. To support distributed storage scheduling and to handle the volatility and slow convergence issues, the theory of Alternating Direction Method of Multipliers (ADMM) [14] can be employed. Adopting the theory of ADMM can help the system effectively solve a large-scale distributed optimization problem.

There are three existing strategies that are intuitive in nature. These include the random distribution method (RDM), average distribution method (ADM), and the ERASURE method. *RDM* adopts random distribution, whereas *ADM* evenly distributes the tasks to all nodes at the mobile

Table 1. Overview of Storage and Processing Methods That Are Used in Smart City Applications

| Method | Architecture | Advantages and Disadvantages | Suggested Applications |
|---|---|---|---|
| Cloud Storage and Processing | Wireless Mesh-Based [125]<br>SOS Framework [32]<br>Data Replication and Reduction-Based [152]<br>Data Replication and Erasure cCode-Based [53] | ⇑ Low Cost, Ubiquity, and Scalability<br>⇑ Data Centralization, Backup, and Recovery<br>⇑ Easy Deployment<br>⇓ Performance Dependence on Bandwidth<br>⇓ Security Vulnerabilities<br>⇓ Creating A Single Failure Point<br>⇓ Dependence on Service Providers<br>⇓ Extended Delays Caused By Distance | Resource-Intensive Application with Emphasis on Ubiquity and Reliability |
| Edge Storage and Processing | STACEE [97]<br>Fog Server-Based [87]<br>Cloudlet-Based [133]<br>Three-layer [88]<br>ACMES [149] | ⇑ Low Cost and High Scalability<br>⇑ Real-Time/Near Real-Time Suitability<br>⇑ Avoiding Server Traffic Congestion<br>⇑ Low Overall Latency<br>⇓ Lack of Robust Security Methods<br>⇓ Resource Constraints of Edge Devices<br>⇓ Data Localization at Edge | Real-Time Applications With Emphasis on Performance and Low Bandwidth Utilization |

edge cloud. *ERASURE* method calls a centralized function to evenly distribute the tasks to a subset of mobile edge cloud nodes. The subset of nodes is an output of a prediction algorithm for the upcoming failure rates of the nodes in the mobile edge cloud [63]. The authors of Reference [149] show that ACMES significantly improves these three existing algorithms in terms of total cost, reliability, power usage, and withdrawal risks. Table 1 provides a summarized comparison between cloud-based and edge-based implementations. It lists some of the domain-specific architectures associated with each implementation and considering the advantages and disadvantages of each method, it explores some of the suggested applications.

## 5 ALGORITHMIC BUILDING BLOCKS FOR DATA PLANE SERVICES

The data plane incorporates analysis, interpretation, visualization, and decision-making tools that utilize the captured and stored data. The true power of the big data cannot be harnessed without the use of the algorithms that constitute backbone of the data plane. In this section, these algorithms and tools are studied in different categories: data analytics, machine learning, and deep learning. Visualization tools for the captured data are elaborated on in Section 5.4, which provide an intuitive way for users to extract summarized information.

### 5.1 Data Analytics

Data analytics is a science of examining both quantitative and qualitative information to provide a solid framework toward assisting the citizens with decision-making processes. Although advanced data storage techniques allow storage and retrieval of massive datasets (whether selectively or not), extracting emergent characteristics from a vast collection of data can only be achieved through the application of sophisticated—and computationally intensive—analytics. Aside from big data and data-analytics tools, which are required for the aggregation and control of large volumes of constantly changing data, various tools and solutions are needed to effectively extract comprehensible conclusions and provide insight for city decision makers [108]. However, fulfilling these objectives within the boundaries of data plane's requirements (discussed in Section 4.1) has proven challenging [57]. This section studies various data-analytics algorithms that are proposed in the literature toward fulfilling these tasks. Depending on the insights they provide, data-analytics algorithms can be categorized into four groups: *Descriptive*, *Diagnostic*, *Predictive*, and *Prescriptive* analytics.

**Descriptive Analytics:** Being more general than others, descriptive analytics portrays the demographics of the observed data by providing metrics and measures about it. For example, in a traffic controlling application, we can run descriptive analytics to reveal information regarding the time at which traffic congestion reaches its maximum [33]. The final stage of descriptive analytics techniques involves result visualization and representation.

**Diagnostic Analytics:** After detecting issues, diagnostic analytics can be applied to thoroughly investigate the root causes of events. For example, after inferring that the maximum traffic congestion occurs daily from 17:00 to 18:30, diagnostic analytics can be employed to reveal that, e.g., 80% of the congestion can be attributed to after-work hours and 20% can be associated with construction in specific sites.

**Predictive Analytics:** Based on evaluation results, predictive analytics forecasts the likelihood of an event to transpire in the future. Statistical analysis of past data can yield a multi-parameter function, which can be used to reveal information regarding the nature or timing of future events. For example, using predictive data-analytics techniques, it is possible to use past and current traffic information to forecast future traffic trends [43].

**Prescriptive Analytics:** By detecting the correlation between current data and history of past events, prescriptive analytics develops several scenarios based on the logic of "what might happen if that happens." This analysis of alternative possible chain reactions can help users decide the best course of action. Output of the prescriptive analytics is generally several actions that lead to the best resolution for a given problem. For example, a smart home solution that tracks energy consumption profile of residents can provide them with recommendations to reduce their energy consumption [36], thereby allowing residents to adjust their lifestyle accordingly.

These categories of data analytics can be applied to any smart city application; however, their implementation details remain dependent of the data-plane architecture. Particularly, considering a data plane's requirements and characteristics (see Section 4.2), conventional architectures are often ineffective for the emerging generation of applications. Novel structures are therefore proposed in the literature. For example, *City Data and Analytics Platform* (*CiDAP*) [121] is a Big Data-based framework that particularly targets IoT communication and data collection. Capable of processing large datasets collected by IoT middleware, CiDAP imparts context awareness and intelligence in a wide range of smart city applications. Collected data are stored as JSON documents and processed in the *Big Data Repository* component using a *CouchDB NoSQL* database. Comprehensive processing of the stored data in the *Big Data Repository* is relegated to the *Big Data Processing* component, which is powered by Apache Spark [132]—an open source Distributed Stream Processing Framework (DSPF). Storing and processing large amounts of data, having real-time and batch processing modules, and being tested in the SmartSantander testbed are key advantages of the CiDAP platform [121].

### 5.2 Machine Learning

By detecting patterns and trends in past data, ML enables a system to automatically *learn* (without specifically being programmed) and *improve*. Overall, ML algorithms can be categorized as (i) supervised, (ii) unsupervised, and (iii) reinforcement learning. In supervised learning, a *label* is associated with each input value, whereas in unsupervised learning, input values remain unlabeled. Reinforcement learning utilizes a reward-based mechanism, where the goal is to choose the sets of actions in the environment that maximize the accumulated reward. Within the smart city and IoT context, supervised and unsupervised learning techniques are more commonplace than reinforcement solutions. In this section, we analyze the applicability of each category to smart city applications and investigate the suitability of ML algorithms for various types of sensors.

*5.2.1 Supervised Learning.* Supervised learning techniques center around predicting the best fitting output values for given inputs. Supervised learning can be applied to both *classification* and *regression* problems. A problem falls under classification if the solution can be associated with only a finite number of discrete categories such as "apple" or "orange." Applications of ML are numerous. Internet Service Providers (ISPs) can use it to *classify* network traffic flow based on its source and destination application (e.g., HTTP, DNS, Online Game, etc.) [98], which enables them to improve QoS and ensure Lawful Interception (LI). Classifying sound patterns in a city into a limited number of categories such as a siren, gunshot, car horn, and street music is also an example of classification [119]. In Reference [69], an application is studied to determine the route of buses by using accelerometers and microphones, without using GPS. In this classification application, a bus is determined to be in one of the multiple pre-determined locations. In contrast to classification, regression is used for problems where the output can assume an infinite set of outcomes. For example, forecasting utility demand is a regression problem, because its output is a continuous set of possibilities [15, 143]. Among the algorithms studied in this section, support vector machine, $k$-nearest neighbor, Random Forest, and adaptive boosting are typically used for classification purposes, while linear regression and decision tree are primarily used for regression problems.

*Support Vector Machine (SVM).* SVMs are applicable to both classification and regression problems, although they are generally used for the former. A binary SVM performs binary classification, where a hyperplane is created to divide input values into two classes. This hyperplane is re-adjusted with abrupt changes in input data (such as removal/addition of input values) to minimize its margin. *Hyperplane margin* denotes the distance from the hyperplane to the nearest input values, and it quantifies the accuracy of the determined hyperplane. SVM is recommended for smaller and less noisy datasets, since training time can become prohibitively long as the dataset size increases. Furthermore, the accuracy of SVM is known to be sensitive to noise.

SVMs are particularly suitable for applications with a limited number of constituents. The work proposed in Reference [7] introduces an SVM-based Parkinson Diseases (PD) detection framework, which facilitates the detection of the illness in its early stages. The framework relies on various voice recording sensors (e.g., smartphones, portable computers, and voice recorders) to capture patients' voices. SVM is then applied to the data (on a cloud-based server) to separate the features of healthy and unhealthy users with an accuracy of 97.2%.

Taking advantage of the computational efficiency of SVMs in comparison to stochastic models such as the Markov model, the authors of Reference [46] propose an SVM-based blackout prediction system for smart grid applications. By inputting past data and the real-time status of the transmissions lines (e.g., load distribution, mean, variance, and cumulative distribution function of power), SVM algorithm classifies grid's status into either *normal* or *blackout*. The latter indicates the onset of a cascade failure, where a domino-like effect of failures can lead to a large blackout. Experimental analysis shows that blackout prediction could reach an impressive 100% accuracy.

The Winner-Takes-All (WTA) technique can be used to create multi-class SVMs. In this approach, a dedicated binary SVM is created for every category, which determines whether a data sample belongs to a given class. The SVM with the highest confidence determines the overall output. Multi-class SVM is widely used in smart cities. The study conducted in Reference [40] proposes a multi-class SVM-based system that can classify human motion into multiple categories such as walking, standing, lying, and running. To minimize the invasiveness of the proposed system, SVM features are extracted from RF signals, which include various parameters such as variance, skewness, expectation of Received Signal Strength (RSS), Doppler power spectrum, and Root-Mean-Square (RMS). Using a Gaussian kernel, the authors report an overall (all categories combined) accuracy of 88.6%.

Another multi-class SVM is investigated in Reference [155], where the authors propose a fuzzy SVM for Facial Emotion Recognition (FER). The algorithm can detect seven types of emotions (happy, sad, surprised, angry, disgusted, frightened, and neutral). To overcome major challenges of FER such as large feature set size and excessive computational resource requirements, this work proposes a novel Biorthogonal Wavelet Entropy, where Shannon entropy is executed on every coefficient (gray-level image) to extract features. An accuracy of 96% is reported.

*K-nearest Neighbor (KNN).* KNN can be used in both classification and regression problems. However, similar to SVM, classification is typically considered the primary target application. KNN leverages a simple yet efficient algorithm to the extent where it is typically used as a benchmark for comparing other algorithms. KNN is a non-parametric, lazy algorithm. The former implies that specifications of given input values are not important. This is one of the major strengths of KNN, which circumvents challenges posed by the discrepancies between actual data in real-world implementations and data formulated in theoretical studies. Additionally, because the training data are not used for generalization—meaning that there may be no training phase or if there is, it remains minimal—KNN is assumed a lazy algorithm, which substantially accelerates the training process. However, this limits the algorithm's scalability, because increasing population size can increase execution time and memory usage [130]. In contrast to SVM, KNN remains efficient when data dimensionality increases [95].

Utilizing the simplicity and versatility of KNN, the authors of Reference [142] propose a KNN-based Structural Health Monitoring System (SHMS) that can detect damages in a structure. They use piezoelectric (PZT) sensors to collect data on a structure's status. The feature set is extracted from raw data by applying Principle Component Analysis (PCA), which results in a substantial reduction in input data size. The authors use both *fine* (which uses one neighbor) and *coarse* (which uses 100 neighbors) data to classify a structure's status into one of the four different types of damage and a *non-damage* or *healthy* state. The proposed system achieves an 85–93% accuracy.

The study conducted in Reference [91] proposes a KNN-based Power Management System (PMS) to optimize energy usage of electric appliances. The proposed architecture consists of three components. (i) A data acquisition (DAQ) gathers data from an appliance's power outlet. (ii) A recognition algorithm processes DAQ readings to identify the connected appliance. (iii) The output of the appliance recognition algorithm is fed to a PMS to provide energy-saving recommendations. Each DAQ device is equipped with ACS712 current sensors to determine the energy consumption of an appliance. Appliances are classified under their specific classes by using a KNN algorithm. Recognition rates range between 84 and 94%.

*Random Forests.* Random Forest is a powerful supervised ML algorithm. While its performance is superior for classification, this algorithm can be applied to both classification and regression problems. As its name suggests, a Random Forest consists of a number of decision trees, where robustness and accuracy of the algorithm increase with the number of incorporated trees. Each tree in the forest performs a classification. For a new object to be placed in a decision tree, it has to have the most votes of all other trees in the forest. When applied to regression problems, a new object is placed in a decision tree if it has the highest average of outputs by different trees. Random Forest (RF) is particularly suitable for incomplete datasets as high accuracy can be achieved even if a large portion of data is missing. Additionally, it does not suffer from over-fitting the model when the number of trees increases. Random Forest is also effective for processing large datasets with high dimensionality [12].

An example implementation of Random Forest is discussed in Reference [148], which proposes a system for localization and identification of street light poles using a mobile cloud system. The proposed solution is called *LiDAR* and consists of vehicle-mounted RIEGL VMX-450 system

(a Mobile Laser Scanning device) [117]. LiDAR cloud combines data from three different datasets: Ring Road South (RRS), International Conference and Exhibition Center (ICEC) along with a part of RRS, and Yun Ding Tunnel (YDT). These datasets include 1,055 street light poles and 701 million points in the evaluated scenario, which necessitates a computationally powerful system to host the Random Forest algorithm. To classify street light poles, SVM and Random Forest techniques are used. While the recall, precision, and F1 scores under SVM can yield performance measures of 96.5%, 97.0%, and 96.6%, respectively, RF achieves 95.9%, 99.2%, and 97.5% in corresponding metrics, respectively.

An indoor localization system based on Random Forest is proposed in Reference [19]. Implemented for hostile environment of emergency rooms, it uses Ultra High Frequency (UHF) RFID signals to estimate the location (room) of patients. Two specific problems, however, complicate the implementation of this system. First, the limited range of RFID translates to a high ratio of missing data (because not all samples are received by all antennas). Second, transmissions generated in different rooms differ only marginally in terms of signal strength, which complicates their classification. Addressing these challenges and improving the performance and scalability of the system, the system leverages a hierarchal approach, where a *k*-means-based algorithm first clusters signals into non-overlapping *macro-regions*, each of which encompasses rooms with similar signal-strength profiles. Within each region, Random Forest is applied to find the room associated with the input signal. This hierarchal implementation reduces the number of classes, which translates to improved accuracy. The system achieves an accuracy of 98% (where either the correct room was guessed (83%) or an adjacent room was selected (15%)).

The conducted work in Reference [153] uses Random Forest classification to estimate Air Quality (AQ) in locations that lack a centralized AQ monitoring station. The proposed solution uses historical meteorology data (such as humidity, temperature, and wind speed), traffic information (such as road length and congestion), and Points of Interests (POI, such as density of schools, factories, malls, etc. in an area) to classify AQ into six categories (from excellent to severely polluted). Labeled datasets are used to grow and train 400 trees. The accuracy reaches 81.5%.

*Adaptive Boosting (AdaBoost).* Designed for classification problems, AdaBoost operates in conjunction with simple learning algorithms (termed *weak learners*) and aims to improve their performance. In comparison to sophisticated models with full feature sizes, weak learners are simpler but less accurate. Employing an ensemble of weak learners (instead of relying on a single powerful classifier) offers numerous advantages; this approach is less susceptible to processing slow-downs associated with large feature sets. This makes AdaBoost more applicable to real-time applications. For example, it is possible to use AdaBoost to accelerate object detection algorithms by a factor of ≈2.5×, making them applicable to a wide range of real-time services such as "Smart Telescope" and "Signfinder" (which facilitates the navigation of blind and visually impaired people in city streets) [25].

Reduced training time is another major advantage of AdaBoost. However, the small number of weak classifiers in rear layers can lead to overfitting problem. Cascade-AdaBoost-SVM is an algorithm for fall detection that addresses this issue [26]. The features extracted from triaxial accelerometers—amplitude of XYZ direction, signal intensity (Signal Magnitude Vector), and signal integration (Signal Magnitude Area)—are fed into a hybrid AdaBoost/SVM algorithm. The proposed system automatically switches to SVM if AdaBoost fails to meet the performance requirements with a predefined number of classifiers. The system achieves an accuracy of 98%, when the sensor is worn around the waist (0.68% better than Cascade-AdaBoost and 1.41% better than SVM).

In Reference [20], AdaBoost is used as a binary classifier to detect indoor and outdoor WiFi devices, thereby creating a localization system. The proposed algorithm involves multiple weak

classifiers that compare Received Signal Strength (RSS) between multiple Access Point (AP) pairs. If the difference exceeds a threshold, then the WiFi signal is classified as either *indoor* or *outdoor*, depending on the training set and relative positions of APs. In large buildings, the majority of APs remain out of the range of devices. Therefore, many weak classifiers involve AP pairs with $RSS = -\infty$. The proposed work addresses this issue by modifying AdaBoost classifiers and adding zero as a third possibility to their output set. This allows weak classifiers, which are incapable of contributing to the overall algorithm (those that involve two out-of-the-range APs) to output zero, which affects neither the weights nor the error of the algorithm. The algorithm only requires 540 bytes of memory for every weak classifier, making it particularity suitable for portable and resource-constrained devices.

*Linear Regression (LR).* LR is a statistical model that formulates the relation between explanatory and dependent variables through a linear equation. The performance of the algorithm is determined by two parameters. First is the *success rate* of a set of explanatory variables at predicting a dependent variable, and second involves determining the significant explanatory variables that play an important role in producing the dependent variable. These two criteria yield a relationship between explanatory and dependent variables before fitting a linear regression model. The simplest linear equation that can be defined is $Y = b \cdot X + c$, where $Y$ denotes the dependent variable, $c$ is the intercept, $X$ is the explanatory variable, and $b$ represents the slope of the line. Linear regression is not suitable for every problem type. Clearly, variables must be linearly dependent. Furthermore, the variance of the error among explanatory and dependent variables must be similar and variables must be normally distributed.

Using LR, the causes of traffic jams can be analyzed by formulating the relation between people and places they visit during a specific period of time [145]. The proposed work chooses Beijing as the testbed and determines 20 types of facilities people visit during a day (termed *POI* or *Points of Interest*), such as life services, corporations and business, government and organization, shopping services, and so on. Each POI has a name, a type, and location information (i.e., latitude and longitude). LR correlates the region function with users' mobility. By solving for the explanatory variables using a sparse representation classifier, the accuracy reaches 74%.

A Land-Use Regression (LUR) system is developed in Reference [56] to improve the spatial resolution of AQ monitoring. The proposed solution involves mobile sensors that collect AQ data, thereby trading off temporal resolution for spatial resolution. Linear Regression is then applied to formulate the correlation between samples and explanatory data (building height, terrain elevation, distance to next road, terrain slope, traffic volume, industry density, road type, distance to next large road, and terrain aspect). The resulting equation is then used for AQ estimation (or prediction using forecast information) in locations where no measurement is available.

*Decision Trees (DT).* DT is part of the supervised learning family. Therefore, similarly to other members of this category, it can be applied to both regression and classification problems. For classification, this algorithm can predict a class, whereas, for regression, training with decision rules of prior training data enables the algorithm to calculate the value of target variables. Decision trees are constructed using parameters such as *information gained* and *Gini index*. Calculating and sorting these parameters allow the algorithm to decide where to place the value in a given dataset in the decision tree. The value with a high information gain parameter is placed at the root. A Gini index is used to measure the frequency of a randomly chosen element that is identified incorrectly. Hence, a value in the dataset with a low Gini index is preferable.

A cloud-based smart grid application of Decision Trees is presented in Reference [126]. The initiative is supported by the U.S. Department of Energy and Los Angeles Department of Water and Power (LADWP) under the name of Los Angeles Smart Grid Project. The application

includes a web-portal and a mobile app user interface for visualization of current and past energy-consumption patterns. The objective is to detect supply/demand issues by collecting data from dedicated sensors and non-dedicated dynamic sensory sources in real time and to correct them by starting demand-side management from customers through a process termed *Dynamic Demand Response* ($D^2R$) optimization. The proposed software platform supports data and result-sharing among researchers and engineers via a secure repository; machine-learning models that are formed in a scalable manner and trained over vast historical datasets for the prediction of future demands are provided. The project uses a regression tree, which is trained using past continuous-valued target values. Datasets consist of feature vectors, each of which contains energy use in 15-minute intervals. Leaf of the regression tree is a numerical kWh value and is able to forecast energy demands corresponding to each feature vector.

*5.2.2 Applications Integrating Unsupervised Learning.* Unlike supervised learning techniques, input values in unsupervised learning are not labeled; an unsupervised learning technique constructs structures by just evaluating the relation between the given inputs. While supervised learning can be loosely defined as function approximation, unsupervised learning mostly involves determining closely related input variables. In terms of throughput (execution of more complex processing tasks), unsupervised learning outperforms supervised methods. On the downside, its results can become unpredictable on some occasions. Prominent examples of clustering techniques include $k$-means and Density-Based Spatial Clustering and Application with Noise (DBSCAN), and in reinforcement learning applications, Q-learning is typically the primary solution.

*$k$-means.* A $k$-means clustering algorithm groups given input values into $k$ distinct clusters; values in the same cluster are expected to be as similar as possible, while values in different clusters are expected to be as different as possible. Each cluster has a *centroid* (i.e., a *central value*). Initial positioning of these centroids is extremely important, as placing them in sub-optimal locations may lead to sub-optimal clustering results. Initially, the $k$ centroids are placed as far away as possible from each other. Values close to the determined centroids are assigned to their associated centroids. The similarity is calculated using a distance-based approach. Once the initial grouping is complete, a new centroid is calculated for every cluster. This process is repeated until it converges to an optimal clustering, where no more changes can be performed.

In Reference [6], an application of the $k$-means algorithm is developed to group load profiles of smart meters. Unlike the mainstream research, the proposed system focuses on both the daily and the segmented profiles (the latter are valid for less than 24 hours). A $k$-means algorithm (implemented using Python library Pycluster) is used to estimate missing information or to predict expected future load trends. It can also be incorporated in a Time of Use (ToU) tariff design. The study shows that although the error threshold increases from 1% to 10%, the clustering ratio (the ratio of the number of output clusters to the number of input profiles) decreases from 1 to 0.286.

The Wind Power Forecasting (WPF) system proposed in Reference [150] uses bagging Back Propagation Neural Networks (BPNN) to predict the expected wind power generation of wind turbines based on past data. However, BPNN alone fails to overcome the unpredictability of wind patterns. To address this problem, the proposed $k$-means algorithm first clusters days with the same wind power profile. By applying a *Relief* algorithm, the number of selected features for $k$-means is then reduced from 39 to 3—including average wind speed, blade angle, and ambient temperature. In comparison to BPNN, this hybrid model depresses the root-mean-square-error by 12.7%.

A distributed implementation of $k$-means for strongly connected Wireless Sensor Networks (WSNs) is studied in Reference [114] that provides an alternative to the centralized implementation of applications with scarce communication resources. To improve both speed and accuracy, the algorithm is based on a distributed $k$-means++, which optimally selects the initial centroids.

The maximum-consensus algorithm controls centroid selection and its propagation through the network (utilizing max-consensus implies that the upper bound of the number of nodes must be known, which can become problematic for highly dynamic WSNs). Once the initial centroids are selected and distributed, each node can locate the cluster it belongs to. The average-consensus algorithm is then used to update the centroids. Distributed $k$-means can exceed the speed and accuracy of centralized implementations.

*DBSCAN.* As implied by its name, DBSCAN is a density-based clustering algorithm. It can identify clusters, noises, and outliers in a dataset. While noise or outliers are sparsely located in the data space, clustered data are densely populated in one region. DBSCAN's objective is to determine dense regions (clusters). Each cluster is shaped by two parameters: *epsilon* ($\epsilon$) and *minimum points*. The former is the coverage area of a neighborhood around a point $x$, which is often called the $\epsilon$ neighborhood of $x$. The latter is the minimum number of neighbors within the range of $\epsilon$ radius. Any point $x$ becomes a center or core point when $x$ has a neighbor count greater than or equal to *minimum points*; alternatively, any point $x$ is regarded as a border point if its neighbor count is less than the *minimum points*. However, if it is accessible from some other cluster's centroid, then it belongs to that cluster. Any other point that does not fit into those listed categories is regarded noise or outlier. Major advantages of DBSCAN are (i) unlike $k$-means, having a preliminary fixed number of clusters is not necessary; (ii) DBSCAN detects non-linearly separable clusters and outliers; and (iii) DBSCAN only needs two parameters, i.e., $\epsilon$ distance and minimum neighbor count.

DBSCAN requires no training or prior knowledge about the application, which can be potentially used to address users' privacy concerns. This has motivated the application of DBSCAN in Reference [94], which introduces a system for analyzing households' power consumption using smart meters. In this system, DBSCAN preprocesses power traces to cluster tuples with similar power consumption patterns. Based on this information, it is possible to determine the number of occupants in the house and their activities such as sleeping and eating.

To count the number of individuals in a crowd, a feature-based detection algorithm is utilized in Reference [83] (instead of target detection-based techniques) due to its effectiveness against dense crowd populations. This approach typically involves the application of Speeded Up Robust Features (SURF) algorithm to obtain features. A clustering solution is then applied to group features pertaining to individuals' motions. Since the number of clusters is unknown (because the number of individuals is unknown), DBSCAN is employed instead of $k$-means, which requires the number of clusters as one of its inputs. However, DBSCAN's performance strongly depends on $\epsilon$. To address this drawback, an adaptive $\epsilon$ selection approach based on Minimal Spanning Trees (MSTs) is developed to compute the most optimal $\epsilon$ size.

Reinforcement Learning (RL) assumes that a system can determine the ideal behavioral pattern on its own with the objective of maximizing its performance. It is analogical to how pets get trained by their owners; they are rewarded with their favorite treat for every accomplishment, which encourages them to repeat this behavior to receive more rewards. Similarly, an intelligent agent, such as a drone trying to learn how to fly from one point to another while circumventing obstacles, can achieve this goal based on the reward feedbacks it receives from the environment. The goal is to eventually converge to the global optimum, with the maximum amount of collected rewards. There are multiple ways to implement an RL technique to solve a problem. The most popular approach, however, is to maximize the reward accumulation over a long run. Unfortunately, considering that RL algorithms store values of each state, memory storage requirements can become prohibitive when the problem complexity increases. However, memory problems can be alleviated with value approximation techniques.

Q-Learning is a *model-free* algorithm that learns a policy function based on the insight it gains from past data. In the beginning, the software agent does not have any previous knowledge of the environment it is operating in. By learning the optimal policy, the agent is able to select the best course of actions given its state and the status of the environment.

An example implementation of Q-learning in smart transportation applications is demonstrated in Reference [90]. The proposed method is called Adaptive Traffic Signal Control (ATSC) and is aimed to address the issue of urban traffic congestion caused by the increasing number of vehicles on roads. ATSC combines multiple Q-Learning algorithms with different configurations using traffic signal controls (TSC), thereby building a collection of *RL-TSC* agents. Once the RL-TSC agents are trained online and fine-tuned in real time after the deployment of the system, they can learn ATSC strategies autonomously without prior information. The conducted study evaluates three Q-Learning-based RL-TSC algorithms: RL-TSC-1, where reward functions are defined as the difference between the previous and current average of queue length at the junction; RL-TSC-2, where the reward function is defined as the difference between the sums of the current and previous waiting times; and RL-TSC-3, where the reward function is defined as the unweighted sum of the reward functions of RL-TSC-1 and RL-TSC-2. The learning rate, $\alpha = 0.008$, and the discount factor, $\gamma$, are kept the same among all three algorithms. Results of the experiments show that under unpredictable traffic conditions, RL-TC-1 surpasses the other two with the lowest queue lengths and waiting times and the highest vehicle speeds. RL-TSC-2 performs the worst, and RL-TSC-3 scores in between, as its reward function is the weighted sum of these two approaches.

In another study, Q-Learning is used as the core of an efficient solution to improve multiple access management [55]. The proposed solution minimizes network overload and paves the way for supporting massive machine type communication (MTC). Using a reinforcement learning algorithm such as Q-Learning for eNB selection enables MTC devices to connect to eNBs in a self-organizing way. Conducted experiments prove that MTC devices can choose the eNB that offers a better performance (for example, in terms of delay).

*5.2.3 Machine-learning Algorithms: Summary and Comparison.* Table 2 provides a comparison among various ML algorithms discussed in this section. Suitability of each algorithm to classification, regression, clustering, and reinforcement learning (abbreviated as Cla., Reg., Clu., and Rei, respectively) is determined under the *Target problem* column. Some algorithms, such as SVM, are applicable to both classification and regression problems; however, they are mostly used for the former. Table 2 identifies these primary target applications with (✓✓) notation.

### 5.3 Deep Learning

*Convolutional Neural Network (CNN).* The underlying architecture of CNN makes it very efficient at classifying images. CNN benefits from an expedited training process, which paves the way for training deep neural networks consisting of multiple layers. Additionally, running it on a GPU can further accelerate the training process [131]. CNN is established on three fundamental concepts [27]: (i) local receptive fields, which connect each neuron to only a local region of the input data, (ii) shared weights, which use the same weight vector for the convolution, and (iii) pooling, which is used for input sub-sampling.

CNN and its variants are mainly used for image recognition tasks. The study carried out in Reference [110] proposes an accurate and fast cloud-based License Plate Recognition System (LPRS). A deep CNN is executed in the cloud to extract features of the plates, including plate localization, character detection, and segmentation. To determine background features along with extracting the character/number features on a plate, ReLU and Conv layers are added at the plate recognition layer. As the ReLU layer determines the feature count, the Conv layer identifies the background

Table 2. Overview of ML Algorithms Used in Smart City Applications

| ML Algorithms | Target Problem Cla. | Reg. | Clu. | Rei. | Example Application | Advantages and Disadvantages |
|---|---|---|---|---|---|---|
| Support Vector Machine | ✓✓ | ✓ | ✗ | ✗ | Smart Healthcare [7]/ Information Security [81] | ⇑ High Accuracy<br>⇑ Applicability to Classification and Regression<br>⇓ Extended Training Time for Large Datasets<br>⇓ Adverse Effect of Noise on Accuracy |
| $k$-Nearest Neighbor | ✓✓ | ✓ | ✗ | ✗ | Energy Management [91] | ⇑ Easy Implementation, Versatile<br>⇑ No Special Requirements for Dataset<br>⇓ High Memory Usage, Extended Training Time |
| Linear Regression | ✗ | ✓ | ✗ | ✗ | Smart Transportation [145] | ⇑ Optimal Performance for Select Datasets<br>⇓ Applicable to Only Linear Datasets<br>⇓ Prediction of Only Numerical Value |
| $k$-means | ✗ | ✗ | ✓ | ✗ | Advanced Metering Infrastructure [6] | ⇑ Computationally Efficient $O(k \cdot n \cdot d)$<br>⇑ Easy Deployment, Applicable to Many Datasets<br>⇓ Initialization of $k$, Sensitivity to Outliers<br>⇓ Unsuitability for Nonuniform Clusters |
| DBSCAN | ✗ | ✗ | ✓ | ✗ | Smart Grid [68, 94] | ⇑ No Prior Knowledge of Clusters Count<br>⇑ Deterministic, Robust to Noise<br>⇑ Capable of Creating Non-Uniform Clusters<br>⇓ Requirement for Connected High Density Regions<br>⇓ Poor Applicability to Datasets with Variable Density |
| Decision Tree | ✓ | ✓ | ✗ | ✗ | Smart Grid [126] | ⇑ Applicable to Noisy/Incomplete Datasets<br>⇑ Extraction of Most Discriminatory Features<br>⇓ Reliance on Large Training Set<br>⇓ Computationally Demanding |
| Adaptive Boost (AdaBoost) | ✓ | ✗ | ✗ | ✗ | Smart Healthcare [25] | ⇑ Easy Deployment Only one Initial Value, T<br>⇑ No Prior Knowledge of Weak learners Required<br>⇓ Susceptibility to Overfitting, Low Margins<br>⇓ Sensitivity to Uniform Noise |
| Random Forest | ✓✓ | ✓ | ✗ | ✗ | Smart Lighting [148] | ⇑ Applicable to Large Datasets, Accurate<br>⇑ Applicable to Incomplete Datasets<br>⇓ Susceptibility to Overfitting for Noisy Data<br>⇓ Only Machine Interpretable Classification |
| Q-Learning | ✗ | ✗ | ✗ | ✓ | Smart Transportation [55]/ Resource Allocation [82] | ⇑ Model Free<br>⇓ Unsuitable for Noisy Values |

*Target Problems* are categorized into classification (Cla.), Regression (Reg.), Clustering (Clu), an Reinforcement (Rei.). The primary target problem is identified by (✓✓) notation.

using a feature-weighting algorithm. This approach can effectively address various challenges LPRSs face including the existence of multiple plates in an image when traffic load increases, high brightness, extra information, misinterpretation due to plate damage, and image distortion induced by bad weather conditions. Bare-metal cloud servers with kernels improved for NVIDIA GPUs are used to further accelerate the training process. The developed LPDS algorithm consists of three phases: (a) the first phase is plate preprocessing. The operation of this phase affects the

overall performance significantly. (b) In the second phase, the heuristic convolution image manipulation technique is used for plate detection. (c) The third phase involves the application of CNN for extracting plate features and Character Recognition.

The study in Reference [110] provides a comparison of existing license plate recognition algorithms. The edge-based methods are simple and fast, nonetheless, they require constant availability of edge information. For low-resolution videos, Connected Component Analysis (CCA) is an acceptable method. Genetic algorithms can also be used as a viable option for neural networks, but the ratio of brightness over color density remains crucial when evaluating the relationship between width and height of the localization region. Geovision is another method for license plate recognition that employs an advanced neural network technology to analyze pictures of license plates.

*Restricted Boltzmann Machine (RBM).* RBM finds patterns in data by reconstructing input. The algorithm leverages a two-layer architecture consisting of a *visible* and a *hidden* layer (therefore, making it a shallow-net classification as opposed to a deep one). In this architecture, all nodes at the input and output layers are fully connected. However, no connection can be established between nodes at the input layer; hence the term *restricted*. Training an RBM for reconstructing the given input is achieved through several forward and backward phases. At the end of each phase, a parameter called *KL* is monitored to check the convergence. The *KL* parameter quantifies how the newly created and actual inputs are related. RBM does not require labeled inputs, which makes it suitable for real-world datasets, such as voices, photos, videos, and sensor readings. Another powerful aspect of RBM is that it can automatically detect the most important features during the pattern formation process. The algorithm then reconstructs the input data based on the extracted features.

An application of RBM is discussed in Reference [89]. The proposed system applies RBM and a Recurrent Neural Network (RNN) to a large-scale transportation network analysis. It leverages an architecture composed of these neural networks and uses it to model and estimate evolution of traffic congestion by relying on the GPS data from taxis. Most studies conducted on this subject consider congestion areas separately in only a small-scale network and approach the congestion problem using analytical methods that fail to model the involvement of human activity factors. This yields sub-optimal results. Leveraging the vast amount of constantly updated data generated by traffic sensors on existing freeway networks, the system analyzes traffic flows on a large-scale highway network and applies data-mining techniques to examine changing patterns of traffic congestion. However, the curse of dimensionality poses multiple challenges for applying traditional data-mining approaches to ever-changing traffic conditions, as the increasing dimensionality of the input leads to exponential growth of the required training data. This problem can be overcome by using deep learning techniques. Therefore, the system combines RBM and RNN to learn multi-dimensional patterns of traffic congestion. The authors implement the system atop the NVIDIA Compute Unified Device Architecture (CUDA) environment to expedite the learning process and fully utilize the power of GPU-based parallel computing architecture [131].

*Deep Belief Network (DBN).* RBMs suffer from the vanishing gradient problem, which causes neurons in the earlier layers to learn more slowly than neurons in the farther layers. The problem is caused by the backward shrinking of gradient sizes through the hidden layers. This situation eventually decreases accuracy. By combining multiple RBMs in a way that hidden layer of one RBM is the visible layer of the one ahead of it, and training them together, we obtain a neural network called DBN, which is capable of addressing the vanishing gradient problem. When training a DBN, input layer at the very first RBM constructs the input values for the second RBM. This process continues until the output at the very last RBM is constructed. Afterward, the fine-tuning process starts, which involves labeling extracted patterns at each RBM using a supervised learning technique. Advantages of DBN over shallow networks are (i) only a small set of labeled

data is necessary, which makes it suitable for real-world applications; (ii) when executed by GPU, the training time can be significantly reduced; and (iii) it solves the vanishing gradient problem, which in turn improves the accuracy.

A resilient and self-healing application based on DBN, namely a Microgrid Social Network (MGSN), is proposed in Reference [61]. It combines multiple Microgrids (MGs) by using a shared physical bus and an online social network, through which MGs communicate. An MG is a relatively small, robust, and stable energy infrastructure, which becomes especially useful in case of disasters involving energy outages and grid instabilities. MGs are capable of supplying local loads by taking advantage of Distributed Energy Resources (DERs). Collaboration with other MGs in the cloud further enhances power availability and reliability of an MG. Considering that social media are the main communication platform during emergencies and disasters, this combination of power and communication can expedite the emergency power delivery system with emergency response capability The existence of a positive relationship among individuals in a social network is an underlying assumption in a majority of proposed models. This assumption, however, is not valid in some real-world scenarios. The proposed MGSN architecture is built on top of a DBN-based method, which is capable of characterizing and predicting the complex social relationship of individual MGs in the MGSN. Moreover, facilitating the cooperation among MGs with regard to their own social preferences can be accomplished by applying a game-theoretic paradigm [61].

*Deep Autoencoder (DA).* DAs consist of two DBNs that are symmetrically placed next to each other. Each half has four or five shallow layers that represent the encoding of the first and the second half, respectively. While the encoder gradually decreases the dimensionality of a given input, it steadily increases dimensionality of the created feature vector. Features are extracted by feeding the input into the network and processing them with the decoding and encoding phases. Eventually, the DA learns how to reconstruct a given input by using the extracted features. Since a DA is built using a DBN, its layers are RBMs, excepting minor differences. Unsupervised learning approaches can be used to pre-train a DA, hence addressing the under-fitting problem.

Research presented in Reference [144] constructs a DA-based system for detecting turbine blade breakages using data of a Supervisory Control and Data Acquisition (SCADA) system. The importance of this task can be associated with the repercussions of malfunctioning wind turbines, which range from a significant capital loss to unplanned downtimes and environmental hazards. Existing SCADA features are not capable of presenting irregularities in data before the blade is broken. To overcome this issue, a DA-based method is introduced to monitor the condition of a turbine's blades and detect damages and breakages using the Reconstruction Error (RE) from the SCADA data. SCADA systems are advanced condition monitoring systems (CMSs), which have been extensively integrated into commercial wind farms. Using the data obtained from SCADA, DA derives a signal (RE) and displays the blade breakage trends. Data are obtained from four different SCADA-integrated wind farms: Shandong, Anhui, Tianjin, and Ningxia. SCADA data are input into DA for analysis. In the encoding phase, the feature set of the input is extracted, while dimensionality of the given input gradually reduces and transforms. RBMs are used for mapping models during input transformation. Once the dimensionality of the given input reaches a minimum level, the decoding process is initiated to reconstruct the input iteratively. To minimize the difference between the reconstructed and the actual input, the model is fine-tuned by using a back-propagation method [144].

Another application is discussed in Reference [67], which introduces a new technique to predict the destinations of smart-fare-card users based on the large-scale data provided by Automatic Fare Collection (AFC) system in Seoul, Korea. It includes IDs along with time stamps when passengers are embarking and disembarking at bus stops. A deep-learning model predicts the final stops of passengers based on the entry-only data acquired from smart-cards and land-use characteristics.

Table 3. Overview of Deep Learning (DL) Algorithms That Are Used in Smart City Applications

| Deep Learning Algorithms | Target Problems | Example Applications | Advantages and Disadvantages |
|---|---|---|---|
| Convolutional Neural Network | Image and Video Processing/ Natural Language Processing | License Plate Recognition [110]/ Smart Healthcare [24, 99] | ⇑ Suitable for Image Processing<br>⇓ Slow Training without GPU<br>⇓ Requirements for Large Training Set<br>⇓ Computationally Demanding |
| Restricted Boltzmann Machine | Dimensionality Reduction/ Classification/ Feature Learning | Smart Transportation [89]/ Smart Healthcare [99] | ⇑ Capability to Infer Missing Data<br>⇑ Feature Extractor for Other DL Algorithms<br>⇓ Training Complication (Contrastive Divergence) |
| Deep Belief Network | Classification/ Clustering | Smart Grid [23, 61]/ Smart Healthcare [99] | ⇑ Small Training Set, Real-Time Friendliness<br>⇑ GPU friendliness, High Accuracy<br>⇓ Dependence on Parameter Initialization |
| Deep Autoencoder | Dimensionality/ Reduction | Smart Grid [144] | ⇑ Suitable for Image Search Semantic Hashing<br>⇑ Solid Encoding of Final Model<br>⇓ Difficult Optimization Due to Back-Propagation |

It is possible to obtain missing smart-card destination information by using the same basic logic and preset rules that apply to the transactions. If a smart-card transaction meets these rules, then the destination of the cardholder can be inferred. Limited availability of data for destination validation can be addressed by defining an assumption rate, which corresponds to the proportion of entry-only transactions that meet a certain number of preset rules. This deep learning solution encompasses four layers: input, output, and two hidden layers. The number of features (135) determines the number of nodes at the input layer, whereas the number of nodes at the output layer depends on the maximum number of candidates for the last stop (5). By using ReLU and applying a dropout rate of 30% for the two hidden layers, impressive performance improvement in deep learning is reported. A SoftMax function is adopted for each node at the output layer for classification purposes. The model is trained and tested using Python programming language and Keras 3.0, which is an open-source deep learning library.

The proposed solution surpasses existing research in three aspects. First, it improves the prediction accuracy of the last stop for entry-only-smart-card. Second, it is applicable to not only metro travels but also bus chained-journeys. Third, it is validated using a large dataset. Compared to a reference model, accuracy is enhanced by more than 5% with the contribution of additional feature variables. The precision in determining the estimated destination vs. the observed destination at an individual bus stop is improved to 87.5%, representing an improvement of 6%.

Aside from their many advantages, deep learning solutions cannot replace machine-learning techniques in the smart city ecosystem. Deep learning is particularly suitable for applications that involve high data dimensionality [99]. Particularly, these solutions are becoming more relevant with the diffusion of crowd-sensing and non-dedicated sensing [52], where only deep learning techniques can cope with the astounding volume and velocity of the data [48]. Selecting the most suitable deep learning solution also depends on the type of data (image, video, or signal). Typically, CNN, DBN, RBM, and Autoencoders are suggested for image processing. CNN-based solutions are proposed for both signal and video processing [99]. Table 3 summarizes this section's discussion about deep learning techniques.

### 5.4 Data Visualization

To humans, as the last decision makers in a smart city ecosystem, the big data generated by the sensing plane and analyzed by the data plane is not comprehensible. Data visualization techniques help create a comprehensible representation of complex data, thereby providing assistance with making decisions. Making data visually presentable and appealing shortens the decision-making process, particularly in comparison to the data presented in spreadsheets. Visualization is typically the preferred method of presentation, as it can easily reveal patterns, trends, and correlations that may be missed by humans otherwise. Various forms of graphs, charts, histograms, diagrams, plots, maps, and endless combinations of these formats can be used to visualize data. Traditional visualization techniques can be further enhanced to create an interactive visualization environment.

Accessing the data without having technical knowledge about the underlying IoT resources—such as sensors and data processing components—must be ensured by the platform in the sensing-as-a-service systems. This could be achieved by setting an IoT middleware platform to fetch data. However, opting for an IoT middleware approach to retrieve data imposes a compelling challenge for data consumers, since it demands both technical and domain expertise to be configured. Authors in Reference [107] offer a knowledge-based approach named Context-Aware Sensor Configuration Model (CASCOM) to facilitate the setup stage of IoT middleware platforms, allowing data consumers to conveniently fetch their data. Due to the complexity of data, visualization techniques cannot be typically applied out of the box. The complex interwoven collection of data may contain relevant and non-relevant information depending on the target application and the target user. This implies that irrelevant information must be eliminated to form a relevant dataset. The dataset containing the data of interest is called Key Performance Indicator (KPI).

Big data visualization involves the following considerations. First, a universal visualization technique to meet the requirements of all applications is still nonexistent; therefore, different research cases typically require different visualization approaches tuned for their specific requirements. Second, visualizations must remain as simple as possible by conveying just enough amount of information. Including excessive details can readily eclipse underlying information, while a stark representation of the results may fail to convey crucial details. Finally, visualizations should be hierarchical to allow viewers to adjust the detail level according to their intention.

Major visualization methods are listed as follows:

- **Histograms** are often used for continuous data. They "bin" the input data into multiple predetermined value intervals. Different histograms can be grouped together in one visual presentation.
- **Pie Charts** illustrate numerical proportions of data. It is one dimensional and it contains discrete categories.
- **Stacked Bar Graphs** partition a whole into splits (similar to pie charts). It can be multi-dimensional with multiple discrete categories. Comparing data is easier than pie charts.
- **Venn Diagrams** are useful for showing overlaps in a discrete dataset.
- **Scatter Plots** are capable of representing multiple data groups. Data being observed from each group is continuous. It can fit to the illustrated data to indicate trends. Scatters plots become distracting as data size increases.
- **Line Graphs** are favorable for illustrating discrete data changing in a continuous domain (e.g., temperature vs time). Different discrete data can be compared together.
- **Bar Graphs** represent data by dividing them into categories. Data represented in each category are discrete.

- **Heat Maps** represent data by diving them into matrices and color-coding each value in a matrix. They perform well if there are natural semantics. Understanding the representation can be counterintuitive.
- **Bubble Plots** are a variation of scatter plots. Each bubble represents three-dimensional information, $(x, y)$ coordinates of the bubble, and its size. The way that data are represented is very intuitive. However, the size is not perfectly quantitative.

In Reference [141], authors use a different approach to data visualization. They use various types of glyphs to illustrate the evolution of real-time data streams in a city. Leveraging an object-oriented design, they develop a modular and scalable architecture to address one of the biggest problems of a smart city (transforming daily data streams into information), which eventually leads to strategic and solid decisions using an effective visualization tool.

Defined within the context of public healthcare, MediMap [76] is another example of data mining and decision-support. MediMap employs a combination of data visualization and mining techniques to improve decision-support and facilitate the management of data collected from various Community Health Centers (CHC). MediMap implementation is grounded in two metrics (i) Rate of Accesses to Health Services (RAHS) and (ii) Availability of Health Services for Patients (AHSP). By utilizing these metrics, data are visualized in forms of graphs, charts, and map colorings to give more meaningful insights to experts, who are in charge of planning CHCs. MediMap developers also integrate GIS data into their proposed system, which enables decision-makers to see areas with low accessibility rates. This allows them to take corrective actions by installing new facilities in these regions.

Effective visualization in a smart healthcare applications reduces diagnosis time (from days to minutes). Recording, aggregating, and visualizing data streams from health sensors attached to patient bodies are discussed in Reference [102]. This article proposes an innovative visualization technique to present 24-hour ECG sensor recordings within a single graph [103], thereby helping physicians to diagnose Long QT Syndrome (LQTS) [100]. This data visualization technique is performed in three steps: cleaning, denoising, and plotting data in an easy-to-read way.

A three-layer management system architecture to provide a smart mobility solution to urban bus transportation is proposed in Reference [146]. While the first and second layers are respectively related to big data and data-analysis techniques, the third layer caters to decision support in an interactive visualization environment. Authors implement a map-based visualization dashboard that presents metrics about bus network on a city map and facilitates tracking and assessment of numerous performance metrics interactively, thereby assisting urban planners with decision-making. The visualized map incorporates fundamental information such as bus travel time, bus speed, passenger counts, and so on.

The authors of Reference [137] examine a decision support system, which involves collecting data from local, regional, and statewide geospatial databases. They use several visualization tools to make useful predictions about brownfields redevelopment. To build such a system, innovative methods for accessing and analyzing data and Geographic Information Systems (GIS) visualization models are integrated together. The system is validated by a state-of-the-art resource-modeling application called Smart Places, which enables users with no technical knowledge to interactively analyze land-use scenarios, outline suggested changes, and assess these suggestions against local and regional objectives and constraints.

## 6 OPEN ISSUES AND CHALLENGES

This section investigates (from the standpoint of applications and data plane) some of the unresolved challenges in the literature, as well as emerging invocations that can potentially shape the future of smart cities.

### 6.1 Application Plane

Challenges of power transmission systems can be listed as the lack of adequate transmission capability; a competitive grid operation market; costs associated with the redesign of the power system plan; determination of the type, mix, and implementation of the sensing and control hardware; and the coordination of centralized and decentralized control [9]. In smart homes and smart utilities, power signature analysis remains an open issue, which can help alleviate some of the pressures imposed upon power transmission systems; as stated in Reference [116], extracting power consumption signatures can help locate major power consumers and provide context-aware building automation. However, voltage control is of paramount importance for power systems. In the presence of distributed power generation, control of a steady voltage becomes a challenge. A possible solution is the integration of a communication system with the on-load tap-changer (OLTC) for voltage control [39]. However, optimal control and latency minimization remain unresolved.

Intermittency of renewable resources also introduces some obstacles (such as capacity limits of wind power transmission), whereas distribution issues brought by new flow patterns and unifying interconnection standards further complicate the implementation of smart utility services. Moreover, researchers in this field have to address operational challenges for new energy generation such as wind and solar, such as forecasting and scheduling [65].

Challenges in the field of smart lighting are closely related to free-space optical communication. As reported in Reference [123], the tradeoff among illumination and communication, small spacing between LEDs, mobility and the management of Line of sight (LoS) due to having the scarcely available LSO alignment, attaining a network with capability of detecting angle-of-arrival, and designing solid-state devices with novel approaches for modulation and illumination are the key challenges in smart lighting services.

### 6.2 Data Plane

Cloud-based processing and storage introduce various challenges to smart city applications. For instance, complications brought by cloud-based smart grid include cost-effective provisioning without the replacement of legacy systems; secure integration of new capabilities with existing systems; and, most importantly, the capability of offering a parallel implementation framework for extra capabilities while leveraging software to minimize expenses and latency [65].

Besides clouds, cloudlets and mobile edge computing infrastructures have many advantages. However, they are difficult to deploy for the first time, as there is no standard protocol to facilitate the inclusion of the cloudlet in the network.

Furthermore, in many smart city services, algorithms that determine the best configuration for the deployment of dedicated sensors and the recruitment or non-dedicated sensors remain an open issue. For instance, as mentioned in Reference [22], under a smart transportation use case, some traffic patterns can be useless for traffic flow detection; ignoring useful patterns and including useless data negatively affect system's performance. The number of possible configurations increases exponentially as the number of available sensors grows. While the referred study proposes to use the Taguchi method to find the best possible configuration, new algorithms aiming at various design constraints are needed.

The challenges related to non-dedicated sensing also relate to the data plane, since the state of the art is in need of a mechanism to scale gathered information and somehow extend it as if all non-dedicated sensors are recruited. However, novel scalable solutions are necessary. Another challenge in non-dedicated sensing is the difficulty in providing incentives to the users to participate; while approaches such as gamification have been investigated [111], the trustworthiness of the acquired data from crowd-sensing participants is a big challenge [112, 113]. Novel solutions for user participation are necessary for crowd sensing to become widespread.

The volume of collected data in smart cities also poses a well-known computational challenge. Typically, sampling sifts out a large portion of the collected information to overcome the sheer size of the data. Since the traditional data processing techniques do not scale properly with the size of input data, there is a significant research opportunity for developing alternative solutions. Particularly, it seems that hybrid data processing techniques represent the most expedient substitute. A hybrid solution can be implemented in both architectural and algorithmic levels. The former involves utilization of both fog and cloud computing, where fog computing performs relatively shallow and context-aware analysis, while the cloud is used for deeper evaluations. The algorithmic level aims to exploit the synergetic effects of the existing algorithms. Particularly, fusing deep leaning and reinforcement learning can effectively address some of the drawbacks of RL (e.g., manual feature extraction, scalability, etc.) [93].

Reconciling deep learning with power-limited field devices is also another growing branch of research in the literature, which is expected to advance the proliferation of fog computing. Complicated models still remain beyond the capabilities of most field devices, while simpler ones suffer from degraded accuracy. Fine-tuning this existing tradeoff between performance and accuracy has recently received attention in the literature, where the complexity of models are dynamically adjusted based on the requirements of the application and the context [136].

Dependency on the central platform is a grand challenge for the data plane regardless of the sensing infrastructure. For instance, the study in Reference [120] presents the pros and cons of dedicated and embedded smartphone sensors to detect available parking spaces. The common challenge of both infrastructures has been identified as the requirement of a centralized processing aggregation center. Furthermore, due to the amount of data to process, designing novel parallel applications that are amenable to *massively parallel programming* is necessary [131].

Visualization of the acquired sensor data presents an interesting challenge in that although the amount of data is explosive, the information content is disproportionately low. For example, in the Smart Health application described in Reference [102], authors describe a visualization methodology for a patient's remotely monitored ECG recordings [31, 103]. This methodology allows a doctor to browse through 20–30 patient's ECG data, totaling ≈300MB, within a few seconds. Using this visualization method, their study determines a concealed "long QT cardiac condition," while discovering the same hazard is very challenging in a regular hospital setting [100]. Furthermore, the authors investigate using machine learning algorithms, such as Random Forests and SVMs, to automatically detect certain cardiac hazards in the acquired data [62].

### 6.3 Smart Sustainable Cities

The concept of smart sustainable cities emerges from a conflation of smart city and sustainable city platforms. In the absence of a universal definition, the former can be liberally viewed as ICT services that aim to enhance the efficiency of cities resource management, while the latter generally plans to maintain the long-run sustenance of cities by protecting their resources from depletion. These liberal definitions do not necessarily imply an inherent disparity between the two concepts, yet existing works show that the mainstream smart city research mostly focuses on immediate social and economic benefits while deprioritizing environmental factors (e.g., $CO_2$ emission, and waste, water, and energy management) [4]. In fact, urban planners and smart city service providers hope that these ICT improvements eventually turn into an engine that gradually, yet steadily, advances modern cities toward a more sustainable future. Although the economic returns of these developments are clear, the exact avenue that links the existing technology with a future of more sustainable and more resilient cities remains mostly under-explored.

The promulgation of smart city services brings about multiple negative side effects. Chief among these are the security and privacy threats, which are a direct implication of the complex attack

surface. The literature is well aware of this problem, but there is no indication that a reliable universal solution is likely to be available in the near future. The concept of *security* expands well beyond its most immediate interpretation. The *cyberization* of the most fundamental resources in a city (fresh water, food distribution, electricity and energy, etc.) adds a new dimension to the security and provides ample research opportunities regarding the vulnerabilities, possible malfunctions (intentional or not), and their ramifications. Smart city services typically discriminate in favor of the tech-savvy population, which can exacerbate social inequality and further the clout of technology giants. Most of the less direct implications of smart city technologies are also under-discussed in the literature. It is expected these services reduce person-to-person interactions among citizens, increase screen time, and can further distance citizens from nature (refer to Reference [28] for a short yet interesting elaboration on these topics).

## 7 SUMMARY AND CONCLUDING REMARKS

The IoT and data-analytics concepts have entered into a new era with the rise of smart cities by integrating existing services with computerized intelligence (or, alternatively, *machine intelligence*), which minimizes human intervention. Smart city applications in healthcare, transportation, utility, safety, and environmental health are some of the beneficiaries of this new era, which are the candidates to significantly improve their usefulness by utilizing machine intelligence and the continuous progress in IoT technology.

In light of these developments, our study first details emerging trends in the implementation of the data plane, including cloud-based and edge-based data processing and storage, and their multi-faceted interactions, tradeoffs, advantages, and disadvantages, to establish a premise for our recommendations to researchers and developers based on requirements of their target applications. We then investigate how the algorithmic backbone comprising data analytics, machine-learning, and deep learning components can be utilized in this framework to incorporate machine intelligence in smart city applications. We analyze each of these components, not merely as stand-alone algorithmic solutions but also from the standpoint of their complementary characteristics, arguing that hybrid implementations often result in a superior performance. For each of these components, we also discuss open issues and challenges, where we consider the state-of-the-art research in the field to predict future trends and research opportunities.

## REFERENCES

[1] C. C. Aggarwal, N. Ashish, and A. Sheth. 2013. The internet of things: A survey from the data-centric perspective. In *Managing and Mining Sensor Data*. Springer, Boston, MA, 383–428.

[2] M. Agiwal, A. Roy, and N. Saxena. 2016. Next generation 5G wireless networks: A comprehensive survey. *IEEE Commun. Surv. Tutor.* 18, 3 (2016), 1617–1655.

[3] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman. 2012. A survey of information-centric networking. *IEEE Commun. Mag.* 50, 7 (2012), 26–36.

[4] Hannele Ahvenniemi, Aapo Huovila, Isabel Pinto-Seppä, and Miimu Airaksinen. 2017. What are the differences between sustainable and smart cities? *Cities* 60 (2017), 234–245.

[5] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash. 2015. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Commun. Surv. Tutor.* 17, 4 (2015), 2347–2376.

[6] A. Al-Wakeel and J. Wu. 2016. K-means based cluster analysis of residential smart meter measurements. *Energy Procedia* 88, Suppl. C (2016), 754–760. Cities and Urban Energy 2015-Applied Energy Symposium and Summit 2015: Low carbon cities and urban energy systems.

[7] M. Alhussein. 2017. Monitoring Parkinson's disease in smart cities. *IEEE Access* 5 (2017), 19835–19841.

[8] S. Ames, M. Venkitasubramaniam, A. Page, O. Kocabas, and T. Soyata. 2015. Secure health monitoring in the cloud using homomorphic encryption, a branching-program formulation. In *Enabling Real-Time Mobile Cloud Computing through Emerging Technologies*, T. Soyata (Ed.). IGI Global, Chapter 4, 116–152.

[9] S. M. Amin and B. F. Wollenberg. 2005. Toward a smart grid: Power delivery for the 21st century. *IEEE mpe* 3, 5 (2005), 34–41.

[10] O. Arias, J. Wurm, K. Hoang, and Y. Jin. 2015. Privacy and security in internet of things and wearable devices. *IEEE Trans. Mobile Comput. Sci.* 1, 2 (Apr. 2015), 99–109.
[11] A. Basalamah. 2016. Sensing the crowds using Bluetooth low energy tags. *IEEE Access* 4 (2016), 4225–4233.
[12] Mariana Belgiu and Lucian Drăguţ. 2016. Random forest in remote sensing: A review of applications and future directions. *ISPRS J. Photogram. Remote Sens.* 114 (2016), 24–31.
[13] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. 2012. Fog computing and its role in the internet of things. In *Proceedings of the 1st MCC Workshop on Mobile Cloud Computing*. ACM, New York, NY, 13–16.
[14] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.* 3, 1 (2011), 1–122.
[15] Bruno M. Brentan, Edevar Luvizotto Jr., Manuel Herrera, Joaquín Izquierdo, and Rafael Pérez-García. 2017. Hybrid regression model for near real-time urban water demand forecasting. *J. Comput. Appl. Math.* 309 (2017), 532–541.
[16] R. Buyya, H. Stockinger, J. Giddy, and D. Abramson. 2001. Economic models for management of resources in peer-to-peer and grid computing. In *Commercial Applications for High-Performance Computing. Proc. SPIE* 4528 (2001), 13–26.
[17] K. Cabaj and W. Mazurczyk. 2016. Using software-defined networking for ransomware mitigation: The case of CryptoWall. *IEEE Netw.* 30, 6 (Nov. 2016), 14–20.
[18] F. Calabrese, M. Colonna, P. Lovisolo, D. Parata, and C. Ratti. 2011. Real-time urban monitoring using cell phones: A case study in Rome. *IEEE Trans. Intell. Transport. Syst.* 12, 1 (Mar. 2011), 141–151.
[19] Luca Calderoni, Matteo Ferrara, Annalisa Franco, and Dario Maio. 2015. Indoor localization in a hospital environment using random forest classifiers. *Expert Syst. Appl.* 42, 1 (2015), 125–134.
[20] O. Canovas, P. E. Lopez de Teruel, and A. Ruiz. 2017. Detecting indoor/outdoor places using WiFi signals and AdaBoost. *IEEE Sens. J.* 17, 5 (Mar. 2017), 1443–1453.
[21] L. Catarinucci, D. de Donno, L. Mainetti, L. Palano, L. Patrono, M. L. Stefanizzi, and L. Tarricone. 2015. An IoT-aware architecture for smart healthcare systems. *IEEE IoT J.* 2, 6 (Dec. 2015), 515–526.
[22] K. Y. Chan and T. S. Dillon. 2013. On-road sensor configuration design for traffic flow prediction using fuzzy neural networks and taguchi method. *IEEE Trans. Instrum. Meas.* 62, 1 (Jan. 2013), 50–59.
[23] Gary W. Chang and H. J. Lu. 2018. Integrating grey data preprocessor and deep belief network for day-ahead PV power output forecast. *IEEE Trans. Sust. Energy* (2018), 1–1.
[24] Min Chen, Jun Yang, Jiehan Zhou, Yixue Hao, Jing Zhang, and Chan-Hyun Youn. 2018. 5G-smart diabetes: Towards personalized diabetes diagnosis with healthcare big data clouds. *IEEE Commun. Mag.* 56, 4 (2018), 16–23.
[25] X. Chen and A. L. Yuille. 2005. A time-efficient cascade for real-time object detection: With applications for the visually impaired. in *Proceedings of the Computer Society Conference on Computer Vision and Pattern Recognition* (2005), 28–28.
[26] W. C. Cheng and D. M. Jhan. 2013. Triaxial accelerometer-based fall detection method using a self-constructing Cascade-AdaBoost-SVM classifier. *IEEE J. Biomed. Health Inform.* 17, 2 (Mar. 2013), 411–419.
[27] Igor M. Coelho, Vitor N. Coelho, Eduardo J. da S. Luz, Luiz S. Ochi, Frederico G. Guimarães, and Eyder Rios. 2017. A GPU deep learning metaheuristic based model for time series forecasting. *Appl. Energy* 201 (2017), 412–418.
[28] Johan Colding and Stephan Barthel. 2017. An urban ecology critique on the smart city model. *J. Clean. Prod.* 164 (2017), 95–101.
[29] M. Collotta and G. Pau. 2015. A novel energy management approach for smart homes using Bluetooth low energy. *IEEE J. Select. Areas Commun.* 33, 12 (Dec. 2015), 2988–2996.
[30] B. F. Cooper, A. Silberstein, E. Tam, R. Ramakrishnan, and R. Sears. 2010. Benchmarking cloud serving systems with YCSB. In *Proceedings of the 1st ACM Symposium on Cloud Computing*. ACM, New York, NY, 143–154.
[31] J. P. Couderc, M. K. Aktas, T. Soyata, and A. T. Page. 2016. ECG Clock Electrocardiogram Based Diagnostic Device and Method. US Patent App. 15/368,587.
[32] S. Dey, A. Chakraborty, S. Naskar, and P. Misra. 2012. Smart city surveillance: Leveraging benefits of cloud data stores. In *Proceedings of the 37th Annual IEEE Conference on Local Computer Networks Workshops*. IEEE, Los Alamitos, CA, 868–876.
[33] R. Du, C. Chen, B. Yang, N. Lu, X. Guan, and X. Shen. 2015. Effective urban traffic monitoring by vehicular sensor networks. *IEEE Trans. Vehic. Technol.* 64, 1 (Jan. 2015), 273–286.
[34] M. Erol-Kantarci and M. Uysal. 2016. *Multiple Access in Visible Light Communication Networks*. Springer International Publishing, Cham, 451–461.
[35] A. Fahad, T. Soyata, T. Wang, G. Sharma, W. Heinzelman, and K. Shen. 2012. SOLARCAP: Super capacitor buffering of solar energy for self-sustainable field systems. In *Proceedings of the 25th IEEE International System-on-Chip Conference*. IEEE, Los Alamitos, CA, 236–241.
[36] Eleni Fotopoulou, Anastasios Zafeiropoulos, Fernando Terroso-Sáenz, Umutcan Şimşek, Aurora González-Vidal, George Tsiolis, Panagiotis Gouvas, Paris Liapis, Anna Fensel, and Antonio Skarmeta. 2017. Providing personalized energy management and awareness services for energy efficiency in smart buildings. *Sensors* 17, 9 (2017), 2054.

[37] The OpenStack Foundation. 2010. Openstack Storage. Retrieved from https://www.openstack.org/software/.
[38] R. K. Ganti, F. Ye, and H. Lei. 2011. Mobile crowdsensing: Current state and future challenges. *IEEE Commun. Mag.* 49, 11 (Nov. 2011), 32–39.
[39] C. Gao and M. Redfern. 2011. A review of voltage control in smart grid and smart metering technologies on distribution networks. In *Proceedings of the International Universities Power Engineering Conference (UPEC'11)*. IEEE, Los Alamitos, CA, 1–5.
[40] Y. Geng, J. Chen, R. Fu, G. Bao, and K. Pahlavan. 2016. Enlighten wearable physiological monitoring systems: On-body RF characteristics based human motion classification using a support vector machine. *IEEE Technol. Manage. Council* 15, 3 (Mar. 2016), 656–671.
[41] Van Gerwen, Rob, Saskia Jaarsma, and Rob Wilhite. 2006. Smart Metering, KEMA. Retrieved from http://www.idc-online.com/technical_references/pdfs/electrical_engineering/Smart_Metering.pdf (accessed April 2019).
[42] A. Gharaibeh, M. A. Salahuddin, S. J. Hussini, A. Khreishah, I. Khalil, M. Guizani, and A. Al-Fuqaha. 2017. Smart cities: A survey on data management, security and enabling technologies. *IEEE Commun. Surv. Tutor.* 19, 4 (2017), 2456–2501.
[43] M. Gramaglia, M. Calderon, and C. J. Bernardos. 2014. ABEONA monitored traffic: VANET-assisted cooperative traffic congestion forecasting. *IEEE Vehic. Technol. Mag.* 9, 2 (Jun. 2014), 50–57.
[44] T. Guelzim, M. S. Obaidat, and B. Sadoun. 2016. Chapter 1 - Introduction and overview of key enabling technologies for smart cities and homes. In *Smart Cities and Homes*, Mohammad S. Obaidat and Petros Nicopolitidis (Eds.). Morgan Kaufmann, Boston, 1–16.
[45] Bin Guo, Zhu Wang, Zhiwen Yu, Yu Wang, Neil Y. Yen, Runhe Huang, and Xingshe Zhou. 2015. Mobile crowd sensing and computing: The review of an emerging human-powered sensing paradigm. *ACM Comput. Surv.* 48, 1, Article 7 (Aug. 2015), 31 pages.
[46] S. Gupta, R. Kambli, S. Wagh, and F. Kazi. 2015. Support-vector-machine-based proactive cascade prediction in smart grid using probabilistic framework. *IEEE Trans. Industr. Electron.* 62, 4 (Apr. 2015), 2478–2486.
[47] Hadi Habibzadeh, Tolga Soyata, Burak Kantarci, Azzedine Boukerche, and Cem Kaptan. 2018. Sensing, communication and security planes: A new challenge for a smart city system design. *Comput. Netw.* 144 (2018), 163–200. DOI: https://doi.org/10.1016/j.comnet.2018.08.001
[48] M. Habibzadeh, A. Boggio-Dandry, Z. Qin, T. Soyata, B. Kantarci, and H. Mouftah. 2018. Soft sensing in smart cities: Handling 3Vs using recommender systems, machine intelligence, and data analytics. *IEEE Commun. Mag.* 56, 2 (Feb. 2018), 78–86.
[49] M. Habibzadeh, M. Hassanalieragh, A. Ishikawa, T. Soyata, and G. Sharma. 2017. Hybrid solar-wind energy harvesting for embedded applications: Supercapacitor-based system architectures and design tradeoffs. *IEEE Circuits And Systems Magazine (IEEE MCAS)* 17, 4 (Nov. 2017), 29–63.
[50] M. Habibzadeh, M. Hassanalieragh, T. Soyata, and G. Sharma. 2017. Solar/wind hybrid energy harvesting for supercapacitor-based embedded systems. In *Proceeding of the IEEE Midwest Symposium on Circuits and Systems*. IEEE, Los Alamitos, CA, 329–332.
[51] M. Habibzadeh, M. Hassanalieragh, T. Soyata, and G. Sharma. 2017. Supercapacitor-based embedded hybrid solar/wind harvesting system architectures. In *Proceedings of the 30th IEEE International System-on-Chip Conference*. IEEE, Los Alamitos, CA.
[52] M. Habibzadeh, Z. Qin, T. Soyata, and B. Kantarci. 2017. Large scale distributed dedicated- and non-dedicated smart city sensing systems. *IEEE Sens. J.* 17, 23 (Dec. 2017), 7649–7658.
[53] M. U. Hameed, S. A. Haider, and B. Kantarci. 2017. Performance impacts of hybrid cloud storage. *Computing* 99, 12 (01 Dec. 2017), 1207–1229.
[54] F. Harada, T. Ushio, and Y. Nakamoto. 2007. Power-aware optimization of CPU and frequency allocation based on fairness of QoS. *Syst. Comput. Jpn.* 38, 12 (2007), 37–45.
[55] M. Hasan, E. Hossain, and D. Niyato. 2013. Random access for machine-to-machine communication in LTE-advanced networks: Issues and approaches. *IEEE Commun. Mag.* 51, 6 (Jun. 2013), 86–93.
[56] David Hasenfratz, Olga Saukh, Christoph Walser, Christoph Hueglin, Martin Fierz, Tabita Arn, Jan Beutel, and Lothar Thiele. 2015. Deriving high-resolution urban air pollution maps using mobile sensor nodes. *Perv. Mobile Comput.* 16 (2015), 268–285.
[57] Ibrahim Abaker Targio Hashem, Victor Chang, Nor Badrul Anuar, Kayode Adewole, Ibrar Yaqoob, Abdullah Gani, Ejaz Ahmed, and Haruna Chiroma. 2016. The role of big data in smart city. *Int. J. Inf. Manage.* 36, 5 (2016), 748–758.
[58] M. Hassanalieragh, A. Page, T. Soyata, G. Sharma, M. K. Aktas, G. Mateos, B. Kantarci, and S. Andreescu. 2015. Health monitoring and management using internet-of-things (IoT) sensing with cloud-based processing: Opportunities and challenges. In *Proceedings of the 2015 IEEE International Conference on Services Computing*. IEEE, 285–292.
[59] M. Hassanalieragh, T. Soyata, A. Nadeau, and G. Sharma. 2016. UR-SolarCap: An open source intelligent auto-wakeup solar energy harvesting system for supercapacitor based energy buffering. *IEEE Access* 4 (Mar. 2016), 542–557.

[60] G. Hauber-Davidson and E. Idris. 2006. Smart water metering. *Water* 33, 3 (2006), 38–41.
[61] Y. He, G. J. Mendis, Q. Gao, and J. Wei. 2016. Towards smarter cities: A self-healing resilient microgrid social network. In *Proceedings of the Power and Energy Society General Meeting (PESGM'16)*. IEEE, 1–5.
[62] S. Hijazi, A. Page, B. Kantarci, and T. Soyata. 2016. Machine learning in cardiac health monitoring and decision support. *IEEE Comput. Mag.* 49, 11 (Nov. 2016), 38–48.
[63] H. Huang. 2015. Multi-access Edge Computing. Retrieved from http://www.etsi.org/technologies-clusters/technologies/multi-access-edge-computing.
[64] MongoDB Inc. 2009. MongoDB. Retrieved from https://www.mongodb.com/.
[65] A. Ipakchi and F. Albuyeh. 2009. Grid of the future. *IEEE Power Energy Mag.* 7, 2 (2009), 52–62.
[66] Ishwarappa and J. Anuradha. 2015. A brief introduction on big data 5Vs characteristics and hadoop technology. *Proc. Comput. Sci.* 48, Suppl. C (2015), 319–324.
[67] J. Jung and K. Sohn. 2017. Deep-learning architecture to forecast destinations of bus passengers from entry-only smart-card data. *IET Intell. Transport Syst.* 11, 6 (2017), 334–339.
[68] Yu Kang, Xinting Wang, Xiu Cao, Yangfan Zhou, Zhichao Lai, Yuhao Li, Xuqi Zhang, and Wei Geng. 2018. Detecting anomalous users via streaming data processing in smart grid. In *Proceedings of the 2018 International Conference on Minign Software Repositories (MSR'18)*. ACM, 14–20.
[69] C. Kaptan, B. Kantarci, T. Soyata, and A. Boukerche. 2018. Emulating smart city sensors using soft sensing and machine intelligence: A case study in public transportation. In *Proceedings of the IEEE International Conference on Communications (ICC'18)*. IEEE.
[70] R. Khatoun and S. Zeadally. 2016. Smart cities: Concepts, architectures, research opportunities. *Commun. ACM* 59, 8 (2016), 46–57.
[71] Y. Kim, T. Soyata, and R. F. Behnagh. 2018. Towards emotionally-aware AI smart classroom: Current issues and directions for engineering and education. *IEEE Access* 6 (2018), 5308–5331.
[72] C. K. Koc, T. Acar, and B. S. Kaliski. 1996. Analyzing and comparing montgomery multiplication algorithms. *IEEE Micro* 16, 3 (1996), 26–33.
[73] O. Kocabas and T. Soyata. 2015. Utilizing homomorphic encryption to implement secure and private medical cloud computing. In *Proceedings of the IEEE 8th International Conference on Cloud Computing*. IEEE, 540–547.
[74] O. Kocabas, T. Soyata, and M. K. Aktas. 2016. Emerging security mechanisms for medical cyber physical systems. *IEEE/ACM Trans. Comput. Biol. Bioinform.* 13, 3 (Jun. 2016), 401–416.
[75] A. O. Kotb, Y. c. Shen, and Y. Huang. 2017. Smart parking guidance, monitoring and reservations: A review. *IEEE Intell. Transport. Syst. Mag.* 9, 2 (2017), 6–16.
[76] N. Lavrač, M. Bohanec, A. Pur, B. Cestnik, M. Debeljak, and A. Kobler. 2007. Data mining and visualization for decision support and modeling of public health-care resources. *J. Biomed. Inf.* 40, 4 (2007), 438–447.
[77] I-G. Lee and M. Kim. 2016. Interference-aware self-optimizing Wi-Fi for high efficiency internet of things in dense networks. *Comput. Commun.* 89, Suppl. C (2016), 60–74.
[78] W. D. Leon-Salas and C. Halmen. 2016. A RFID sensor for corrosion monitoring in concrete. *IEEE Sens. J.* 16, 1 (Jan 2016), 32–42.
[79] Chao Li, Yushu Xue, Jing Wang, Weigong Zhang, and Tao Li. 2018. Edge-oriented computing paradigms: A survey on architecture design and system management. *ACM Comput. Surv.* 51, 2, Article 39 (Apr. 2018), 34 pages.
[80] C.-S. Li and W. Liao. 2013. Software defined networks. *IEEE Commun. Mag.* 51, 2 (2013), 113–113.
[81] I. Li, L. Deng, B. B. Gupta, H. Wang, and C. Choi. 2019. A novel CNN based security guaranteed image watermarking generation scenario for smart city applications. *Information Sciences* 479 (2019) 432–447. DOI : https://doi.org/10.1016/j.ins.2018.02.060
[82] Han Li, Hui Gao, Tiejun Lv, and Yueming Lu. 2018. Deep q-learning based dynamic resource allocation for self-powered ultra-dense networks. In *Proceedings of the IEEE International Conference on Communications Workshops*. IEEE, 1–6.
[83] Ronghua Liang, Yuge Zhu, and Haixia Wang. 2014. Counting crowd flow based on feature points. *Neurocomputing* 133 (2014), 377–384.
[84] K. Liao, Z. Zhao, A. Doupe, and G. J. Ahn. 2016. Behind closed doors: Measurement and analysis of CryptoLocker Ransoms in Bitcoin. In *Proceedings of the 2016 APWG Symposium on Electronic Crime Research (eCrime'16)*. IEEE, 1–13.
[85] F. Lin, A. Wang, Y. Zhuang, M. R. Tomita, and W. Xu. 2016. Smart insole: A wearable sensor device for unobtrusive gait monitoring in daily life. *IEEE Trans. Industr. Inf.* 12, 6 (Dec. 2016), 2281–2291.
[86] T. Lin, H. Rivano, and F. Le Mouël. 2017. A survey of smart parking solutions. *IEEE Trans. Intell. Transport. Syst.* 18, 12 (Dec. 2017), 3229–3253.
[87] T. H. Luan, L. Gao, Z. Li, Y. Xiang, and L. Sun. 2015. Fog computing: Focusing on mobile users at the edge. *CoRR* abs/1502.01815 (2015). arxiv:1502.01815 http://arxiv.org/abs/1502.01815

[88] I. Lujic, V. D. Maio, and I. Brandic. 2017. Efficient edge storage management based on near real-time forecasts. In *Proceedings of the IEEE International Conference on Fog and Edge Computing (ICFEC'17)*. IEEE, 21–30.
[89] X. Ma, H. Yu, Y. Wang, and Y. Wang. 2015. Large-scale transportation network congestion evolution prediction using deep learning theory. *PLoS ONE* 10, 3 (Mar. 2015), 1–17.
[90] P. Mannion, J. Duggan, and E. Howley. 2016. *An Experimental Review of Reinforcement Learning Algorithms for Adaptive Traffic Signal Control*. Springer International Publishing, Cham, 47–66.
[91] L. J. V. Miranda, M. J. S. Gutierrez, S. M. G. Dumlao, and R. S. Reyes. 2016. Appliance recognition using hall effect sensors and k-nearest neighbors for power management systems. In *Proceedings of the IEEE Region Ten Conference (TENCON'16)*. IEEE, 6–9.
[92] P. Mirchandani and Fei-Yue Wang. 2005. RHODES to intelligent transportation systems. *IEEE Intelligent Systems* 20, 1 (Jan. 2005), 10–15.
[93] M. Mohammadi and A. Al-Fuqaha. 2018. Enabling cognitive smart cities using big data and machine learning: Approaches and challenges. *IEEE Commun. Mag.* 56, 2 (Feb. 2018), 94–101.
[94] A. Molina-Markham, P. Shenoy, K. Fu, E. Cecchet, and D. Irwin. 2010. Private memoirs of a smart meter. In *Proceedings of the International Conference on Embedded Systems for Energy-Efficient Buildings (BuildSys'10)*. ACM, New York, NY, 61–66.
[95] M. Muja and D. G. Lowe. 2014. Scalable nearest neighbor algorithms for high dimensional data. *IEEE Trans. Pattern Anal. Mach. Intell.* 36, 11 (Nov. 2014), 2227–2240.
[96] A. Nadeau, M. Hassanalieragh, G. Sharma, and T. Soyata. 2015. Energy awareness for supercapacitors using kalman filter state-of-charge tracking. *J. Power Sources* 296 (Nov. 2015), 383–391.
[97] D. Neumann, C. Bodenstein, O. F. Rana, and R. Krishnaswamy. 2011. STACEE: Enhancing storage clouds using edge devices. In *Proceedings of the 1st ACM/IEEE Workshop on Autonomic Computing in Economics (ACE'11)*. ACM, New York, NY, 19–26.
[98] T. T. T. Nguyen and G. Armitage. 2008. A survey of techniques for internet traffic classification using machine learning. *IEEE Commun. Surv. Tutor.* 10, 4 (2008), 56–76.
[99] A. A. Obinikpo and B. Kantarci. 2017. Big sensed data meets deep learning for smarter health care in smart cities. *J. Sens. Actuator Netw.* 6, 4 (2017).
[100] A. Page, M. K. Aktas, T. Soyata, W. Zareba, and J. Couderc. 2016. QT clock to improve detection of QT prolongation in long QT syndrome patients. *Heart Rhythm* 13, 1 (Jan. 2016), 190–198.
[101] A. Page, O. Kocabas, T. Soyata, M. K. Aktas, and J. Couderc. 2014. Cloud-based privacy-preserving remote ECG monitoring and surveillance. *Ann. Noninvas. Electrocardiol.* 20, 4 (2014), 328–337.
[102] A. Page, T. Soyata, J. Couderc, M. Aktas, B. Kantarci, and S. Andreescu. 2015. Visualization of health monitoring data acquired from distributed sensors for multiple patients. In *Proceedings of the IEEE Global Communications Conference (GLOBECOM'15)*. IEEE, 1–7.
[103] A. Page, T. Soyata, J. Couderc, and M. K. Aktas. 2015. An open source ECG clock generator for visualization of long-term cardiac monitoring data. *IEEE Access* 3 (Dec. 2015), 2704–2714.
[104] M. R. Palattella, M. Dohler, A. Grieco, G. Rizzo, J. Torsner, T. Engel, and L. Ladid. 2016. Internet of things in the 5G era: Enablers, architecture, and business models. *IEEE J. Select. Areas Commun.* 34, 3 (Ma. 2016), 510–527.
[105] Veljko Pejovic and Mirco Musolesi. 2015. Anticipatory mobile computing: A survey of the state of the art and research challenges. *ACM Comput. Surv.* 47, 3 (2015), 47.
[106] Gang Peng. 2004. CDN: Content distribution network. *CoRR* cs.NI/0411069 (2004). http://arxiv.org/abs/cs.NI/0411069
[107] Charith Perera and Athanasios V. Vasilakos. 2016. A knowledge-based resource discovery for internet of things. *Knowl.-Based Syst.* 109 (2016), 122–136.
[108] Gianluigi Pillonetto, Francesco Dinuzzo, Tianshi Chen, Giuseppe De Nicolao, and Lennart Ljung. 2014. Kernel methods in system identification, machine learning and function estimation: A survey. *Automatica* 50, 3 (2014), 657–682.
[109] J. S. Plank. 2013. Erasure codes for storage systems: A brief primer. *USENIX Mag.* 38, 6 (2013), 44–50.
[110] R. Polishetty, M. Roopaei, and P. Rad. 2016. A next-generation secure cloud-based deep learning license plate recognition for smart cities. In *Proceedings of the International Conference on Machine Learning and Applications (ICMLA'16)*. IEEE, 286–293.
[111] M. Pouryazdan, C. Fiandrino, B. Kantarci, T. Soyata, D. Kliazovich, and P. Bouvry. 2017. Intelligent gaming for mobile crowd-sensing participants to acquire trustworthy big data in the internet of things. *IEEE Access* 5, 1 (Dec. 2017), 22209–22223.
[112] M. Pouryazdan, B. Kantarci, T. Soyata, L. Foschini, and H. Song. 2017. Quantifying user reputation scores, data trustworthiness, and user incentives in mobile crowd-sensing. *IEEE Access* 5 (Jan. 2017), 1382–1397.
[113] M. Pouryazdan, B. Kantarci, T. Soyata, and H. Song. 2016. Anchor-assisted and vote-based trustworthiness assurance in smart city crowdsensing. *IEEE Access* 4 (Mar. 2016), 529–541.

[114] J. Qin, W. Fu, H. Gao, and W. X. Zheng. 2017. Distributed $k$-means algorithm and fuzzy $c$-means algorithm for sensor networks based on multiagent consensus theory. *IEEE Trans. Cybernet.* 47, 3 (Mar. 2017), 772–783.
[115] J. R. Lin, T. Talty, and O. K. Tonguz. 2015. On the potential of Bluetooth low energy technology for vehicular applications. *IEEE Commun. Mag.* 53, 1 (Jan. 2015), 267–275.
[116] A. Reinhardt, D. Burkhardt, P. S. Mogre, M. Zaheer, and R. Steinmetz. 2011. SmartMeter.KOM: A low-cost wireless sensor for distributed power metering. In *Proceedings of the 2011 IEEE 36th Conference on Local Computer Networks (LCN'11)*. IEEE, 1032–1039.
[117] RIEGL. 2017. RIEGL VMX 450. Retrieved from http://www.riegl.com/uploads/tx_pxpriegldownloads/RIEGL_VMX-450-RAIL_2015-08-24.pdf.
[118] E. S. Rigas, S. D. Ramchurn, and N. Bassiliades. 2015. Managing electric vehicles in the smart grid using artificial intelligence: A survey. *IEEE Trans. Intell. Transport. Syst.* 16, 4 (Aug. 2015), 1619–1635.
[119] J. Salamon and J. P. Bello. 2015. Unsupervised feature learning for urban sound classification. In *Proceedings of the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'15)*. IEEE, 171–175.
[120] R. Salpietro, L. Bedogni, M. Di Felice, and L. Bononi. 2015. Park here! A smart parking system based on smartphones' embedded sensors and short range communication technologies. In *Proceedings of the World Forum on Internet of Things (WF-IoT'15)*. IEEE, 18–23.
[121] E. F. Z. Santana, A. P. Chaves, M. A. Gerosa, F. Kon, and D. S. Milojicic. 2017. Software platforms for smart cities: Concepts, requirements, challenges, and a unified reference architecture. *ACM Comput. Surv.* 50, 6, Article 78 (Nov. 2017), 37 pages.
[122] R. Schollmeier. 2001. A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications. In *Proceedings of the 1st International Conference on Peer-to-Peer Computing*. IEEE, 101–102.
[123] A. Sevincer, A. Bhattarai, M. Bilgi, M. Yuksel, and N. Pala. 2013. LIGHTNETs: Smart LIGHTing and mobile optical wireless NETworks—A survey. *IEEE Commun. Surv. Tutor.* 15, 4 (2013), 1620–1641.
[124] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu. 2016. Edge computing: Vision and challenges. *IEEE IoT J.* 3, 5 (Oct. 2016), 637–646.
[125] H. Y. Shwe, T. K. Jet, and P. H. J. Chong. 2016. An IoT-oriented data storage framework in smart city applications. In *Proceedings of the International Conference on ICT Convergence (ICTC'16)*. IEEE, 106–108.
[126] Y. Simmhan, S. Aman, A. Kumbhare, R. Liu, S. Stevens, Q. Zhou, and V. Prasanna. 2013. Cloud-based software platform for big data analytics in smart grids. *Comput. Sci. Eng.* 15, 4 (Jul. 2013), 38–47.
[127] Eugene Siow, Thanassis Tiropanis, and Wendy Hall. 2018. Analytics for the internet of things: A survey. *ACM Comput. Surv.* 51, 4 (2018), 74.
[128] E. Sit, A. Haeberlen, F. Dabek, B.-G. Chun, H. Weatherspoon, R. Morris, M. F. Kaashoek, and J. Kubiatowicz. 2006. Proactive replication for data durability. In *Proceedings of the International Workshop on Peer-to-Peer Systems (IPTPS'06)*.
[129] A. Solanas, C. Patsakis, M. Conti, I. S. Vlachos, V. Ramos, F. Falcone, O. Postolache, P. A. Perez-martinez, R. D. Pietro, D. N. Perrea, and A. Martinez-Balleste. 2014. Smart health: A context-aware health paradigm within smart cities. *IEEE Commun. Mag.* 52, 8 (Aug. 2014), 74–81.
[130] Yunsheng Song, Jiye Liang, Jing Lu, and Xingwang Zhao. 2017. An efficient instance selection algorithm for $k$-nearest neighbor regression. *Neurocomputing* 251 (2017), 26–34.
[131] T. Soyata. 2018. *GPU Parallel Program Development Using CUDA*. Taylor & Francis.
[132] Apache Spark. 2017. Apache Spark—Lightening-Fast Cluster Computing. Retrieved from http://spark.apache.org/.
[133] I. Stojmenovic. 2014. Fog computing: A cloud to the ground support for smart things and machine-to-machine networks. In *Proceedings of the Australian Telecommunication Networks and Applications Conference (ATNAC'14)*. IEEE, 117–122.
[134] J. Stößer, D. Neumann, and C. Weinhardt. 2010. Market-based pricing in grids: On strategic manipulation and computational cost. *Eur. J. Operat. Res.* 203, 2 (2010), 464–475.
[135] M. Taneja and A. Davy. 2016. Poster abstract: Resource aware placement of data stream analytics operators on fog infrastructure for internet of things applications. In *Proceedings of the 2016 IEEE/ACM Symposium on Edge Computing (SEC)*. IEEE, 113–114.
[136] Ben Taylor, Vicent Sanz Marco, Willy Wolff, Yehia Elkhatib, and Zheng Wang. 2018. Adaptive deep learning model selection on embedded systems. *SIGPLAN Not.* 53, 6 (Jun. 2018), 31–43.
[137] M. R. Thomas. 2002. A GIS-based decision support system for brownfield redevelopment. *Landsc. Urban Plan.* 58, 1 (2002), 7–23.
[138] L. Valerio, A. Passarella, and M. Conti. 2016. Hypothesis transfer learning for efficient data computing in smart cities environments. In *Proceedings of the 2016 IEEE International Conference on Smart Computing (SMARTCOMP'16)*. IEEE, 1–8.

[139] P. M. Varela and T. Otsuki Ohtsuki. 2016. Discovering co-located walking groups of people using iBeacon technology. *IEEE Access* 4 (2016), 6591–6601.

[140] F. Viani, A. Polo, P. Garofalo, N. Anselmi, M. Salucci, and E. Giarola. 2017. Evolutionary optimization applied to wireless smart lighting in energy-efficient museums. *IEEE Sens. J.* 17, 5 (Mar. 2017), 1213–1214.

[141] F. J. Villanueva, C. Aguirre, D. Villa, M. J. Santofimia, and J. C. López. 2014. Smart city data stream visualization using glyphs. In *Proceedings of the 2014 8th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*. IEEE, 399–403.

[142] Jaime Vitola, Francesc Pozo, Diego A. Tibaduiza, and Maribel Anaya. 2017. A sensor data fusion system based on k-nearest neighbor pattern classification for structural health monitoring applications. *Sensors* 17, 2 (2017).

[143] Petra Vrablecová, Anna Bou Ezzeddine, Viera Rozinajová, Slavomír Šárik, and Arun Kumar Sangaiah. 2017. Smart grid load forecasting using online support vector regression. *Comput. Electr. Eng.* 65 (2017), 102–117.

[144] L. Wang, Z. Zhang, J. Xu, and R. Liu. 2018. Wind turbine blade breakage monitoring with deep autoencoders. *IEEE Transactions on Smart Grid* 9, 4 (July 2018), 2824–2833.

[145] M. Wang, S. Yang, Y. Sun, and J. Gao. 2016. Predicting human mobility from region functions. In *Proceedings of the IEEE International Conference on Green Computing and Communications (GreenCom'16)*. IEEE, 540–547.

[146] Y. Wang, S. Ram, F. Currim, E. Dantas, and L. A. Sabóia. 2016. A big data approach for smart transportation management on bus network. In *Proceedings of the IEEE International Smart Cities Conference (ISC2'16)*. IEEE, 1–6.

[147] R. A. Waraich, M. D. Galus, C. Dobler, M. Balmer, G. Andersson, and K. W. Axhausen. 2013. Plug-in hybrid electric vehicles and smart grids: Investigations based on a microsimulation. *Transport. Res. C: Emerg. Technol.* 28, Supplement C (2013), 74–86.

[148] F. Wu, C. Wen, Y. Guo, J. Wang, Y. Yu, C. Wang, and J. Li. 2017. Rapid localization and extraction of street light poles in mobile lidar point clouds: A supervoxel-based approach. *IEEE Trans. Intell. Transport. Syst.* 18, 2 (Feb. 2017), 292–305.

[149] G. Wu, J. Chen, W. Bao, X. Zhu, W. Xiao, and J. Wang. 2017. Towards collaborative storage scheduling using alternating direction method of multipliers for mobile edge cloud. *J. Syst. Softw.* 134, Suppl. C (2017), 29–43.

[150] W. Wu and M. Peng. 2017. A data mining approach combining $k$-means clustering with bagging neural network for short-term wind power forecasting. *IEEE IoT J.* 4, 4 (Aug. 2017), 979–986.

[151] S. Yi, C. Li, and Q. Li. 2015. A survey of fog computing: Concepts, applications and issues. In *Proceedings of the 2015 Workshop on Mobile Big Data*. ACM, 37–42.

[152] J. Yin, I. Gorton, and S. Poorva. 2012. Toward real time data analysis for smart grids. In *Proceedings of the 2012 SC Companion: High Performance Computing, Networking Storage and Analysis*. IEEE, 827–832.

[153] Ruiyun Yu, Yu Yang, Leyou Yang, Guangjie Han, and Oguti Ann Move. 2016. RAQ—A random forest approach for predicting air quality in urban sensing systems. *Sensors* 16, 1 (2016).

[154] X. Zhang, Z. Yang, W. Sun, Y. Liu, S. Tang, K. Xing, and X. Mao. 2016. Incentives for mobile crowd sensing: A survey. *IEEE Commun. Surv. Tutor.* 18, 1 (2016), 54–67.

[155] Y. D. Zhang, Z. J. Yang, H. M. Lu, X. X. Zhou, P. Phillips, Q. M. Liu, and S. H. Wang. 2016. Facial emotion recognition based on biorthogonal wavelet entropy, fuzzy support vector machine, and stratified cross validation. *IEEE Access* 4 (2016), 8375–8385.

Received March 2018; revised January 2019; accepted January 2019