IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON EDGE INTELLIGENCE FOR INTERNET OF THINGS

Received December 20, 2020, accepted January 4, 2021, date of publication January 19, 2021, date of current version January 28, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3052759

Towards Efficient and Intelligent Internet of Things Search Engine

WILLIAM GRANT HATCHER[1], CHENG QIAN[1], WEICHAO GAO[1], FAN LIANG[1], KUN HUA[2], AND WEI YU[1]

[1]Department of Computer and Information Sciences, Towson University, Towson, MD 21252, USA
[2]Department of Electrical and Computer Engineering, Lawrence Technological University, Southfield, MI 48075, USA

Corresponding author: Wei Yu (wyu@towson.edu)

This work was supported in part by the US National Science Foundation (NSF) under grants: CNS 1350145. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding agency.

ABSTRACT The Internet of Things (IoT) has created a novel ecosystem for sensing and actuation throughout our world, enabling intelligently controlled autonomous systems to conserve energy, water crops, manage factories, and provide situation awareness on an unprecedented scale. As IoT progresses, the interest in IoT search engines, that is, search engines to find IoT devices and retrieve IoT data, has grown. While basic examples of IoT search engines exist, considerable challenges prevent the full realization of an efficient and intelligent IoT search engine that provides universal data service, scalable data communication and retrieval, and efficient querying of massively distributed heterogeneous devices and data. In this article, we first propose a generic framework for the IoT search engine, and then present a naming service for the IoT system, an essential component for an effective IoT search engine. We also outline some research challenges and possible solutions for building efficiency and intelligence in the IoT search engine. Further, we present a case study and seek to address a particular aspect of the query process for IoT search, namely efficient and timely query processing. Given the now obvious advances in machine learning, the potential for deep learning-based prediction to improve resource use, and thus query retrieval, is clear. In detail, we utilize Long-Short-Term Memory (LSTM) neural network architecture to predict aggregated query volumes to be preemptively applied and stored for immediate response. Combining several realistic IoT datasets, we explore the efficacy of simultaneously predicting multiple targets for predictive query retrieval.

INDEX TERMS Machine learning, Internet of Things, search engine, edge intelligence, applications.

I. INTRODUCTION

The Internet of Things (IoT) has become integral to improving situation awareness [1]–[6], remote automation and actuation [7]–[9], and data collection [10]–[13], with implications for nearly every industry [14], [15]. The key to this success has been the proliferation of low-cost networked computing devices (i.e., IoT devices) that can be deployed in massive quantities, remotely managed, and power-cycled, to achieve unprecedented data collection and transmission, and the ability to act on remote environments. Such devices, while often limited in computing capabilities, can nonetheless complete complex tasks such as object detection and facial recognition [16], [17], can be equipped with critical features [18] and chips such as temperature and power sensors, and can be embedded in remote locations where resources are scarce but their proper functioning remains highly critical [19], [20]. Moreover, IoT has been envisioned and applied in critical infrastructures to enable cyber-physical systems (CPS) like smart electrical power generation, in consumer devices for the smart home (e.g., smart thermostats, smart wall outlets, and smart light bulbs) to reduce personal energy consumption, in self-driving vehicles for smart transportation for improving traffic efficiency, and in commercial and private industries for efficient and automated manufacturing and production, ushering in the next great industrial revolution, otherwise known as Industry 4.0 [21]–[23].

Despite these achievements, significant work is left to be done to fully realize all the promises of IoT. Remaining challenges include the security of IoT systems, which have

The associate editor coordinating the review of this manuscript and approving it for publication was Sherali Zeadally.

been demonstrated time and again to be vulnerable [24]–[27]. This includes highly-needed device security, as well as security evaluation for various network protocols that enable IoT (constrained application protocol (CoAP), etc.) [28]–[30]. In addition, redundant IoT systems (weather, temperature, atmosphere, etc.) maintained by various disparate organizations in private intranets indicate a waste of resources more broadly. Thus, how to increase the use of IoT data while addressing security and privacy of data sharing has been a growing area of research [31]–[35]. The potential to integrate IoT systems in a manner similar to the traditional Internet as it exists today can enable all businesses, organizations, and individuals to search and discover data on an unprecedented scale, and to transform that data for the next generation of technological advancements. To enable such universal data access, a unified and accepted IoT search framework, or IoT search engine, would need to be designed, implemented, and subscribed to.

While the concept of IoT search and search engines may not be totally new [36]–[40], the increasing deployment and redundancy of IoT systems has increased the need for a fully-realized IoT search engine. This can be coupled with advancements in a variety of related technologies, such as big data and deep learning, which can aid in the achievement of this goal. Clearly, the capacity to store and analyze massive volumes of data has increased dramatically, and advanced machine learning enabled by complex neural networks, denoted as deep learning, has advanced analytical capabilities beyond those of individual humans in a variety of areas [21], [41]–[43]. When combined, these technologies can provide the backbone for an advanced data framework, which can leverage existing infrastructures, incentivize participation through data markets and subscription services, and universalize data of all types for ubiquitous use.

The potential for the IoT search engine is real, and early examples exist, yet barriers to a fully satisfactory IoT search engines still persist. These include the heterogeneity of devices, networks, communication, data, and storage; device ownership, participation incentives, and security; and functional implementation aspects such as distributed architectures, communication protocols, query systems, and quality of service (QoS), among others. In this work, we thus first consider an IoT search engine framework, as laid out in [36], which considers the basic structures and heterogeneity of IoT. This framework layers Quality of Service (QoS), Query Engine, Indexing, and Devices, and allows for interchangeable software modules to optimize and facilitate the Query Engine.

Based on the outlined framework, we then investigate the naming service for IoT systems, which is an essential component to enable the IoT search engine. After investigating existing solutions for naming services and understand their limitations, we enhance the existing naming service in the Internet to design a naming service for IoT systems, and provide examples to demonstrate its application to real-world IoT systems. Further, we outline some research challenges, potential solutions, and future research directions with respect to enabling efficient and intelligent IoT search engine. In addition, we carry out a case study and consider mechanisms to optimize query retrieval and QoS through a generic and interchangeable deep learning-based query module. Specifically, we consider the potential for query prediction to improve QoS for high-volume users and real-time applications. Utilizing query aggregation to combine multiple equivalent queries to improve retrieval, we consider the potential for using deep learning techniques to predict aggregated query volume, which can then be leveraged for predictive query retrieval and data caching. To be concrete, the case study presented in this article is to design a functional query prediction module for the IoT search engine that, when combined with query aggregation, can provide more rapid service to users. By applying ahead-of-time querying, particular user groups or subsets can be better served by reducing or removing the query latency they would otherwise experience.

In detail, the contributions of this work are summarized as follows:

- **Framework:** We consider a generic framework for the IoT search engine. This framework includes a variety of inherent mechanisms, as well as interchangeable software modules. These modules can perform a variety of tasks, including: (i) aggregate queries, and (ii) assess historical query data to predict queries ahead of time.
- **Naming Service:** We design a naming service for the IoT search engine. Particularly, we first compare the benefits and limitations of existing solutions: domain name service (DNS) and named data networking (NDN), which is either not designed for IoT data service or not efficient in existing IP-based networks. We then enhance the DNS and design a naming service for IoT systems. We also provide some examples to demonstrate its use in real-world IoT systems.
- **Open Research Problems:** To enable efficient and intelligent IoT search, we outline major research challenges, potential solutions, and future work that is needed. In terms of efficiency in IoT search, we discuss issues related to the design of query processing, data discovery, and data retrieval, as well as improving the scalability of the IoT search engine given limited resources in the system, and conducting joint resource management. In terms of enabling intelligence in IoT search, we discuss the integration of state-of-the-art data mining and machine learning techniques.
- **LSTM-based Query Prediction:** We conduct a case study to demonstrate the efficacy of IoT search based on LSTM neural network model. Particularly, we first identify existing open source IoT data, which we supplement with our own novel collected data. We then use said data as an analog for query volume in evaluating the potential for query prediction. Using deep learning techniques, we train and validate a variety of predictive regression models based on the LSTM. We additionally consider the overhead and complexity to develop and

maintain such models, and draw conclusions as to their viability.

The remainder of this article is as follows. In Section II, we outline the basic framework of the IoT search engine, globally, and consider the target aggregation and query prediction components in detail. In Section III-B, we present the naming service for IoT systems, which is an essential component for IoT search. In Section IV, we outline challenges and research directions of realizing an efficient and intelligent IoT search. In Section V, we carry out a case study toward achieving intelligence in IoT search via LSTM-based query prediction, introducing the datasets used, the preprocessing considerations for our evaluation, our workflow, and the implemented learning modules. In Section VI, we present our evaluation methodology, environment and results, detail the process of training and validating our models, and demonstrate the results of our LSTM-based query prediction. In Section VII, we evaluate the breadth and depth of existing research on IoT search and search engines, generic search engine mechanisms, and machine learning for time-series prediction, in general. Finally, we provide some concluding remarks in Section VIII.

II. FRAMEWORK

In this section, we consider a generic framework for an IoT search engine as the basis for our study. We outline this framework in general, and then detail several Query Engine modules, the first of which is critical to the optimal and efficient functioning of the search engine.

A. IoT SEARCH ENGINE

For the purposes of this work, we adopt the IoT search engine framework as outlined in [36]. Under this framework, we consider a number of factors that are critical to searching and retrieving IoT devices and their data. At the lowest level, we consider that IoT devices are reachable and addressable utilizing an IoT search address protocol, which features gateway addresses for geolocation services, as well as tags for the descriptions of device identifiers, types of services, etc. As demonstrated in [36], using a variety of network communication protocols, a system can be established that allows for open registration of IoT devices and their data for cross-platform sharing.

All IoT devices are registered to their local gateways, which themselves are registered to a higher-level gateway (i.e., global gateway), as well as registering other neighboring gateways. In this way, gateways have domain over their local set, and all devices and gateways are reachable in a hierarchical structure. Note that we consider hierarchical tree as the structure in this article, but other complex structure such as graph can be considered as well. In addition to gateways and IoT devices, components that mirror traditional Internet search include crawlers and databases for continuous automated retrieval and storage, respectively. Moreover, query engine components take as input queries from human and machine users through a publicly available application

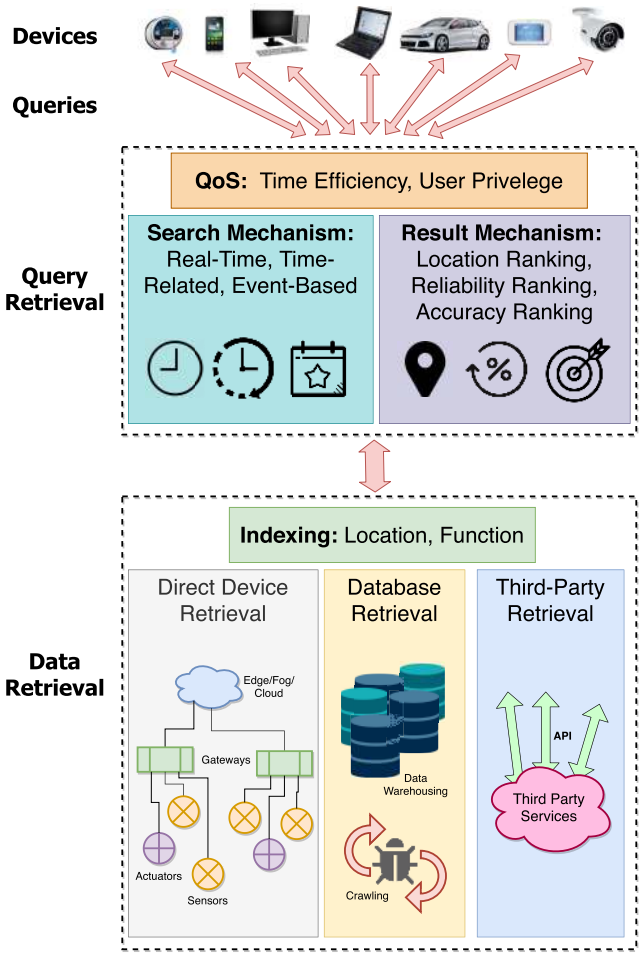

FIGURE 1. A generic framework for the IoT search engine.

programming interface (API), utilize various internal software modules, the crawlers, and the databases, and return as output query results in either finite or ranked format.

Searches can be conducted in a number of ways, including by location and by device function or service type. Query processing should be concerned with the type of resources to retrieve (real-time, periodic, event-based, etc.), the privilege and urgency of the users (safety-critical/real-time need vs. best-effort vs. time-independent), and the ranking mechanism (location, device reachability/reliability, search accuracy, etc.), among many others. These factors should be taken into account in order to provide optimal QoS. Yet, to enable such QoS, additional software modules must be applied to assist the Query Engine in processing the massive volume of data available in an IoT search engine system.

B. MODULES

As mentioned, one of the most critical components for achieving IoT search is the Query Engine. This is what processes queries received from human and machine users and retrieves the requested services. The Query Engine must be able to translate both literal direct queries and human

language queries into the required query language, and must interface with the indexing scheme for IoT devices, crawlers, databases, and third party services. In addition, the Query Engine should be able to provide ranked results based on any number of criteria, including the time properties of the data (real-time, cyclical, event-based), the type of data (streaming, non-streaming), target categories (video, image, weather, temperature, etc.), location, and accuracy of the results, among others.

To achieve the effective operation of the Query Engine, interchangeable software modules are considered. These modules would allow for the rapid adoption of search optimization algorithms, as well as natural language processing of human language queries, aggregation of high-volume queries, and the potential for machine learning-based optimizations, such as query prediction, among others.

1) AGGREGATION

Designed to aid in balancing resource loads, query aggregation can be utilized to combine equivalent queries and reduce the overall load on the system, by reducing the number of query retrievals of the same data. A number of strategies have been considered and implemented to achieve query aggregation [44]–[49], applied to a number of different scenarios and settings, and which depend on a variety of factors, such as QoS, the data to be retrieved, complexity of queries, etc. Another critical function of aggregation is the ability to analyze the volume of equivalent queries, which can in turn be used for additional analytical assessment. Such assessments can be used in time-series prediction to allow for query execution ahead of time, as well as for attack detection against the query engine, among others.

2) QUERY PREDICTION

As the primary target of this research, we assume some query aggregation is in place, and collection of the query volume is likewise. In this way, we can assess the volume of queries deemed ''equivalent'' by the query engine and use this information to attempt to predict the query volume, providing more rapid service to the query user. We consider that automated queries by machine users will likely follow a predictable pattern and have predictable volume based on the number of machine operators generating the queries. Likewise, human users follow predictable patterns as well, such as daily and weekly traffic patterns, eating schedules, etc. In these contexts, whether initiated by human or machine users, we consider the query patterns and volumes predictable globally and within some margin of error. To accomplish query prediction, we can leverage deep recursive neural networks (RNNs) to conduct time-series forecasting. Indeed, a significant amount of work has gone into improving time-series prediction, and the technology is now used in any number of applications, including predicting stock prices [50], [51], network traffic [52], [53], electricity load in power grids [54]–[56], and more.

III. NAMING SERVICE FOR IoT SYSTEMS

In order to provide data-oriented service for IoT systems such as the IoT search engine, an appropriate and effective naming service is essential. In the following, we first review existing network naming frameworks, and then propose our naming service for IoT systems.

A. EXISTING APPROACHES

As the number of IoT devices in IoT systems is large, how to effectively manage them and support IoT search are challenging problems. This calls for designing a naming service for IoT systems. There are two existing approaches to provide naming services. One is to leverage the existing domain name service (DNS) in Internet based on TCP/IP protocols that maps user friendly names to IP addresses [57]. The other is to leverage named data networking [58], [59], which names the devices with content chunks instead of IP addresses. The DNS is one of the fundamental components of the Internet, providing a directory of combined host-names and domain names that match to IP addresses. Note that the combination of host name and domain name comprises the majority of a Universal Resource Locator (URL) [60]. The URL is not only used to find the target device's IP address, but can also locate the resources on a specific device to retrieve data. However, DNS was not originally designed to provide data-oriented services.

The NDN, in contrast to DNS, aims to develop a new Internet architecture, which can leverage the strengths and counteract the weaknesses of the Internet's current host-based communication architecture [58], [59], [61]. With NDN, emerging patterns in communication can be accommodated. Moreover, NDN-related projects have investigated a number of technical challenges, including routing scalability, data fast-forwarding, secure and trustworthy network, data protection and privacy, and fundamental communication theory, and have developed a variety of solutions. Under NDN, a name can be anything: a node, a movie, or an IoT sensor. In addition, users and machine devices retrieve information through the use of Interest packets and Data packets. However, while the NDN has a built-in DNS function and simplifies the process of device and information discovery, exchanging data on the network layer may not be as efficient as the existing IP-based network architecture. Furthermore, according to [59], NDN leverages a name-prefix based routing protocol called Forward Information Base (FIB). Such a routing protocol may be not supported by the existing IP routers. Thus, NDN needs to operate through overlay over existing IP networks.

B. NAMING SERVICE FOR IoT SYSTEMS

Based on RFC 2141 that specifies the Uniform Resources Name (URN) [62], we design a URL-like naming structure specifically for IoT systems. The URN syntax provides a way of encoding character data that can be transmitted in existing protocols. Note that URN is intended to serve as persistent

and location-independent resource identifiers. With URN, we can easily map other namespaces sharing the properties of URN into URN-space.

C. UNIFORM RESOURCE NAME (URN)

First, we note that a URN is a Uniform Resource Identifier (URI) that uses the URN mechanism. URNs are globally unique and persistent identifiers, which are assigned within defined namespaces so that they will be available for a long period of time, even after the resource that they identify does not exist or becomes unavailable. URNs cannot be used to directly locate an item and need not be resolvable. In this case, we always use URL to locate a resource. As an example, URN was proposed as a naming policy for IoT devices [63]. One example of the defined URN naming strategy is as follows: <URN>: := "urn:"<namespace>":"<type>":"<name>":" <value> [":"<vendor-product>":"<version>]. Here, the *namespace* component is used to define the wireless type, which the device is using (Bluetooth, WiFi, etc.). The *type* component determines the kind of device that it belongs to. The *name* represents the device name. The *value* is the UUID (unique ID number) of the target device. The *vendor-product* represents the model name and manufacture name of the device. The *version* tells the product version. As an example, if the target device is an electronic fan, the URN might be *urn:miot-spec-v2:service:fan:00007808*. Here, fan is a WiFi device and its UUID is 00007808. However, using only the URN, it is not easy to identify the device location and which zone this device belongs to. Thus, based on the URL and URN, we need to design a new naming structure for IoT systems.

D. STRUCTURE

To make the IoT systems easy to connect to existing Internet architecture, our proposed IoT naming system structure is based on the DNS inverted tree. While the naming system structure may be formed via complex structures such as graph or its combination with tree via peer-to-peer manner, we focus on the tree structure in this article. Fig. 2 represents the architecture of our IoT naming system. Here, multiple gateways are designed to better fit the operation of IoT devices. In the DNS system, we have a root name server, top-level domain server, authority name server, and local name server. In the IoT naming system, we can have similar server structures. Using a nationwide IoT search engine system as an example, a root gateway is designed to store the IP addresses of the nationwide gateways, as well as the IP address of the root gateway. The state-wide gateways store the IP addresses of the county-wide gateways, as well as the root gateway and the country-wide gateway. The county-wide gateway stores the IP addresses of the city-wide gateways and the state-wide, country-wide, and root gateway. The city-wide gateways store the IP addresses of the local gateways and the upper-level root, country-wide, state-wide, and county-wide gateways. The local gateways store the IP addresses of the IoT sensors and actuators (i.e., typical IoT devices) under

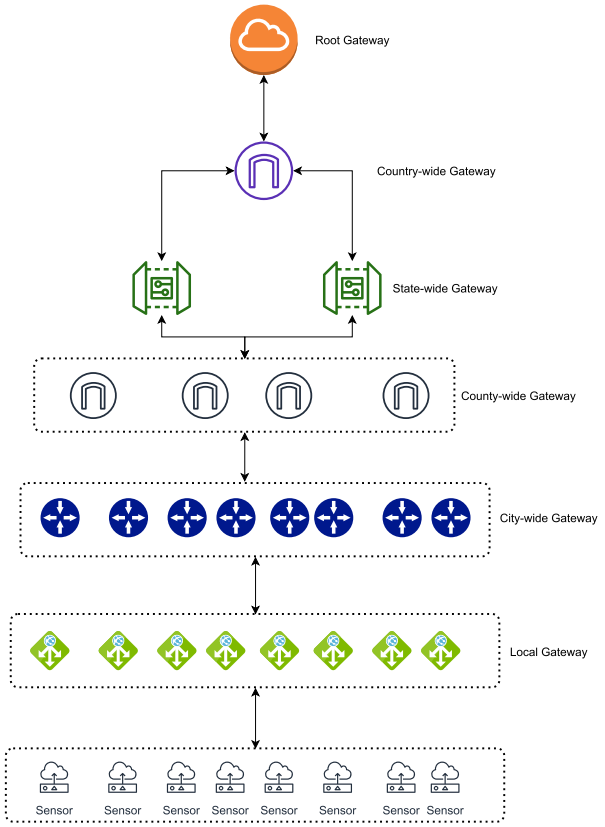

FIGURE 2. Example of IoT naming structure.

their converge, as well as all IP addresses of its upper-level gateways.

The IoT naming system is designed specifically for IoT devices, making the search of target devices easier. As an example, in our prior work [64], the naming structure we developed is as follows: "$< DeviceType >: // < ResourceType > @ < LocalGatewayName > \ldots < City-wideRegionalGateway > \ldots < County-WideRegionalGateway >:< IoTAttribute >:< Pub-Sub >$". Here, the type of IoT resource can be defined before the symbol '@'. The hierarchical architecture of IoT devices can be defined after '@' from local to county. Also, pub-sub field of IoT attribute tells whether the IoT device supports the pub-sub mechanism or not.

Similar to DNS, the IoT naming system can use either recursive query or iterative query. The recursive query is often used when the IoT device is initializing a query to the local gateway. If the local gateway does not know the target device's IP address, the local gateway will act as a client and send a query to its upper-level gateway. On the other hand, the iterative query is commonly used when the local gateway sends a query to the root node. When the root node receives the query, it will send the target IP address to the local gateway or guide the local gateway to process the query.

In addition, the publishing and subscription (pub-sub) mechanism can be implemented in this IoT search system.

Under specific circumstances, user devices need streaming data from selected IoT devices. For example, an autonomous vehicle needs to obtain the real-time traffic data on its path to the destination. In Fig. 3, we can see that at each local gateway, we have two extra components to achieve the pub-sub mechanism. One is the caching storage and the other one is the sensor status table. The caching storage is used when the user subscribes a sensor, which will continue delivering its data to the caching area. The data will then be retrieved by the user from the local gateway. Since the IoT data is time sensitive, outdated data will be discarded from the cache immediately. Also, sensor status (sensor availability, sensor type, pub-sub availability, and pub-sub status, among others) will recorded and stored.

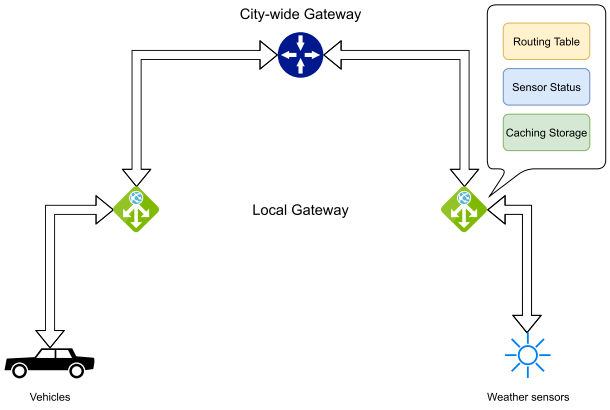

FIGURE 3. Pub-sub of IoT naming system.

E. EXAMPLES

To demonstrate the use of our proposed naming service in IoT systems, we now present three examples in smart transportation, smart parking, and autonomous/self-driving vehicles. The examples we consider are specific to the Towson, Maryland area, but can be generalized to any other locale.

First, Smart Transportation is a typical IoT system that supports real-time traffic information reporting [22], [65]. Assume a vehicle is traveling from Towson University to the nearest local Micro Center (electronics store). There are three local gateways on its route, as shown in Fig. 4. Before the vehicle departs, it will send a request to gateway gw-01 to register itself and request the traffic information of all the possible routes that it might take to reach the destination. Gateway gw-01 will query the city-wide gateway about the IP address of all the corresponding gateways, which are gw-02 and gw-03. After obtaining the IP addresses, gw-01 will query the traffic sensors corresponding to its coverage area. Meanwhile, it will send queries to gw-02 and gw-03 to obtain their traffic information. After gw-01 receives all the traffic information, it will send the combined data packets to the vehicle. Based on the data, the vehicle can determine which is the best route to choose. In addition, before the vehicle leaves the coverage area of gw-01, the gateway will register the vehicle to gateway gw-02. When the vehicle leaves the coverage area of gw-01, it will unsubscribe from the traffic information of gw-01, unregister itself from gw-01, and register to gw-02. The vehicle will continue to follow these steps until it reaches the destination.

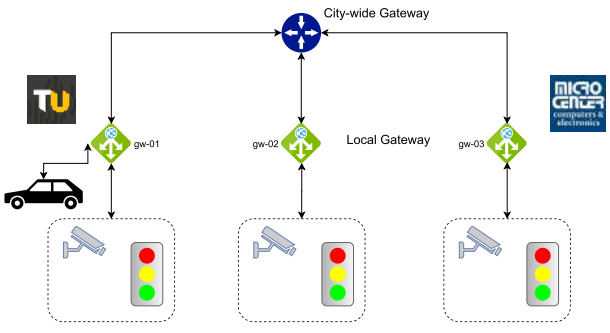

FIGURE 4. Applying naming service in smart transportation.

Second, Smart Parking is an important IoT application in smart cities, enabling efficient utilization of parking resources [66], [67]. Parking lots are considered valuable resources in metropolitan areas, as either public or private lots, they support travel, tourism, and the local economy, generating revenue. Smart parking can improve the driving experience by making it easier to find a parking lot while driving in a city. Generally, smart parking is necessary at the vehicle destination, and thus, for most cases, the vehicle will only need to communicate with one local gateway in the area surrounding the destination. When the vehicle is close to the destination, the local gateway gw-03 will find the nearest available parking lot and send its information to the vehicle, including the number of remaining parking spaces, the parking price, etc. Also, the local gateway gw-03 can predict whether the vehicle will arrive at the parking lot before other competitors based on the historical data of the sensors in the parking lot. If the vehicle will not arrive in time to find a parking space, the gateway gw-03 will recommend another available parking lot for this vehicle in real time.

Third, Autonomous Driving is able to greatly improve traffic efficiency [68], [69] through analysis of a collection of relevant data. To achieve efficient autonomous driving, we need to collect information about traffic, weather, parking lots, date and time, etc. Before the vehicle departs, it will first check the weather and then choose the best drive mode accordingly, such as driving slower and more cautiously in inclement weather. In addition, it will query the traffic information for the optimal route to the destination, taking weather information into consideration to avoid potentially increased traffic and accidents. Moreover, autonomous vehicles may communicate with one another through the local gateway should the sensors on one road stop working. The local gateway can also leverage cameras installed on the autonomous vehicles and traffic intersections to estimate the state of real-time traffic against other the information. The traffic can also be influenced by the state of electrical charge or volume of gasoline in the vehicles, their need to

charge or refuel, the locations of charging and gas stations, and the potential for congestion in these areas, among others. To avoid such problems, the local gateway can aid in finding the best charging station with least waiting time and fastest route to the destination.

IV. OPEN RESEARCH PROBLEMS

As a novel topic, research on the IoT search engine is sparse, and further investigation is necessary. Such research, to fully realize real-time response and quality of service, should include at least the following aspects: building efficiency and intelligence in IoT search. Note that security and privacy issues in IoT search are also important, which is beyond the focus of this article.

A. EFFICIENCY

Consider that, compared with traditional web-content search, it is much more challenging to search for IoT data. On one hand, limited computation capabilities and energy capacity make IoT devices cheap and widely available to deploy globally. On the other hand, from a search perspective, the required resources from IoT systems and the data generated by IoT systems engender redundancies. Thus, how to design methods to enable efficiency in the deployment and use of IoT search is an important issue.

According to the generic IoT search engine framework that we proposed in our prior work [64], the execution of an IoT search engine can be categorized by three major procedures: (i) query processing, (ii) data discovery, and (iii) data retrieval. Generally speaking, query processing is the procedure by which real-time queries are analyzed and redistributed, and the workload of this procedure is determined by the density and complexity of they queries. The data discovery is the process of collecting and identifying information from the IoT devices either in real time or at some other rate. The workload of data discovery is determined by the scale of the IoT systems and their dynamically generated data. Lastly, the data retrieval is the procedure of preparing, aggregating, and formatting the target data to satisfy the query. Despite a variety of algorithms and strategies that are deployed to improve the performance of IoT search, the workload for data retrieval to the system is determined by the query content and the structure of the search network. The query content will determine the location of data and its amounts while the structure of the search network will affect the performance of data delivery.

Thus, addressing the efficiency of the IoT search engine should aim to optimizing the workloads of the above procedures. Furthermore, the optimizations can be achieved from two aspects. The first is the design of algorithms or strategies that enable more efficient processing per-unit workload for each procedure individually, which could minimize the consumption of time and resources. The second is to focus on the scalability of the system, the efficiency of the interactions between the procedures and the underlying technologies, enabling and extending a system with limited resources to deal with increasingly large workloads (queries, data, etc.), and enabling the search engine to process as many queries or cover as many as IoT devices as possible.

In our prior work [64], we proposed a search engine framework, implemented its prototype, and demonstrated its fundamental effectiveness via emulation with real-world datasets. Based on our study, we now discuss potential optimizations for improving the per-unit workload processes of query processing, data discovery, and data retrieval in detail.

- **Query Processing Optimization:** With regard to query processing optimization, the objective is to reduce the processing cost for received queries. One viable method, which can reduce the overhead of redundant queries targeting similar resources or data, so that the average processing cost of each query will be reduced, is query aggregation. For example, if several queries that target the same data are received at the same time or within a specific time window, these queries can be merged to avoid fetching the same data repeatedly. In addition, it is important to design algorithms that are capable of deconstructing complex queries that have multiple targets into smaller sub-queries, which can then be further combined and aggregated to achieve multiple simultaneous aggregated results. Thus, the restructured queries increase the per-unit efficiency of the query processing without changing the response to the user.
- **Data Discovery Optimization:** For data discovery optimization, the optimization goal is the reduction of the overall query latency or response time by proactively preparing the data and making it available as close to the users as possible. Viable strategies include proactive data preparation, query ranking, and data aggregation. The proactive data preparation is to have data ready at the buffer, which is closer to the users, by predicting frequent queries before they arrive. Thus, the respond time to an incoming query is reduced as the data is available immediately. Moreover, the prediction can be enabled and enhanced by applying query ranking, which is based on the performance benchmarks of the data sources. On the other hand, for handing the inquiry for the summary form derived from several groups of raw data, data aggregation can be performed at the nodes, which are more close to the data sources before being sent to respond to the query. By doing this, the overhead in data transmission can be reduced. As an example, if the query is looking for the average value of several groups of data from different data IoT collectors, it is more efficient that each IoT data collector responds to only the average and count of its own data, instead of sending the raw data. For the foreseeable future, methods driven by machine learning and artificial intelligence can be leveraged in developing algorithms and strategies for optimization.
- **Data Retrieval Optimization:** As for data retrieval optimization, the objective is to optimize both the speed and accuracy of locating resources. While it may not be viable for the IoT sensors with constrained

capabilities to continuously send relevant data about their status and identification, it may be possible and more efficient to maintain a list with accurate meta-data and keywords for registered sensors in the data center in a publishing-subscription manner.

In the second aspect, scalability, the key issue is the capability of dealing with increasing workloads by investing resources into the system at a similar or lower rate. The growing workload may include the increase in the number of search targets, the number of independent IoT systems, the scale of data sources, and the number of concurrent queries. A potential solution is to leverage node/server clusters with well-designed load-balancing strategies. More specifically, deploying layers of multiple nodes/servers (e.g., clusters, groups of clusters, inter-groups), in which each single node/server is implemented with the fundamental functions of either query processors or data centers. On the data side, the scale of the covered IoT systems would be handled by the nodes/servers from the clusters of data centers with well-designed indexing strategies. On the query side, overflows of concurrent queries will be handled by the load-balancer that assign nodes/server with vacant capabilities, or forward queries to other clusters if necessary.

Furthermore, unlike traditional content-based web search engines, IoT search is more dynamic. Not only is the data generated by individual sensors updated much more frequently, but the IoT devices also change locations and statuses (wake, sleep, etc.) over time. As a result, the traditional techniques of web crawling and web indexing may not function efficiently in the IoT search environment. Meanwhile, the search space for IoT search engines may also be large due to the large volume of low-cost IoT devices and the hierarchy of the deployed devices in different IoT systems. The computing, networking, and energy resources are also limited in IoT search systems. Thus, how to allocate the limited resources in the IoT search engine is one of the key problems to design cost-effective search engine systems. In detail, some queries to the IoT search engine will be time-sensitive and non-preemptive. Furthermore, limited network resources cannot handle the increasing number of queries. Thus, this calls for designing network resource management schemes so that network resources can be allocated efficiently. Further, due to the complexity of query and IoT data, one query may involve multiple computing resources. Therefore, one important problem in IoT search is how to deploy and schedule computing and network resources to satisfy the performance requirements of queries, which may have different QoS requirements. Moreover, how to conduct joint optimization and design of computing, networking, and energy resource management remains a challenging problem.

B. INTELLIGENCE

Based on the architecture of IoT search that we mentioned above, realizing intelligence in the search process is one of the key requirements in IoT search. To fulfill this requirement, it is important to integrate state-of-the-art data mining and machine learning techniques into IoT search. One of the major purposes of using IoT search is to utilize the massive amount of data generated from a variety of IoT systems. On one hand, data mining could play a critical role in data analysis. Considering the differences between IoT datasets and traditional datasets from a data mining perspective, we see that IoT data is more dynamic and originates from more divergent sources. Whether existing data mining techniques could be applied to IoT search becomes a very interesting problem. In future research for IoT search engines, there is significant potential to apply data mining techniques for the preprocessing, storage, analysis, etc. of massive IoT data. Such data mining approaches need to consider the size and scope of IoT data and will likely include the revision of tested mechanisms, as well as the development of novel data mining methods to meet the requirements of IoT environments. In addition, data mining techniques can be leveraged to improve the security of the IoT search engine through the analysis of queries and anomalies.

On the other hand, machine learning, as a powerful data analysis tool, employs algorithmic analysis of input data to produce statistically generated models, which can be used to perform tasks without explicit instructions. These tasks can include classification, evaluation, segmentation, compression, generation, and organization, among others, and can be implemented in ways that exceed human capabilities at the target task. Moreover, machine learning is a stepping stone to full artificial intelligence. Machine learning models trained on sample data are used to make predictions and decisions on entirely new input, and are used in a wide variety of applications, including spam filtering, computer vision, anomaly detection, and malware analysis, and are especially useful in areas where it is very difficult, very costly, or infeasible to develop a conventional algorithm to perform the complex task [41]. Machine Learning can be applied to a number of areas of the IoT search framework, and could play a critical role in optimizing performance [41], [70], [71]. For example, machine learning could be applied to predict the density of queries such that the system could dynamically adjust the time interval, over which queries are aggregated to avoid large query latency. Meanwhile, applying machine learning in query ranking could help to optimize data preparation and reduce network traffic. With machine learning, we can greatly reduce data size and use trained models to predict traffic over some projected period, validating against real data from sensors to ensure the predicted results are accurate.

Deploying machine learning algorithms to the IoT search engine comes with a variety of challenges. First, machine learning algorithms require high computing capabilities to train on and analyze massive amounts of data, and are thus generally deployed in the cloud server clusters. In addition, to obtain highly accurate results, machine learning algorithms require an appropriately large volume of data to train on. Thus, if machine learning algorithms are deployed in cloud server clusters, the data collection process leads to massive data exchange between servers and IoT devices, which can

occupy the limited network resources and degrade the performance of IoT searches that are latency-sensitive. Although deploying machine learning algorithms to distributed computing environment (e.g., edge computing nodes) is a viable solution, this has its own challenges in terms of the limited computing capacity of distributed computing nodes, the coordination of multiple computing nodes, and security and privacy of data sharing and distributed computing, among others. Indeed, how to optimize machine learning algorithms to reduce computation requirements and deploy the machine learning algorithms to distributed computing nodes are important problems [72].

Machine learning algorithms traditionally require massive amounts of data to be stored and processed to train the learning models in order to satisfy the accuracy requirements of a production system. Nonetheless, the training process could still be very long, making the system unable to satisfy the requirement of timeliness for the IoT search engine. To address this issue, online continuous learning strategies have the potential to reduce training time. In a dynamic IoT search environment, online continuous learning allows the use of dynamically incoming data chunks to continuously train the machine learning model. By doing this, the learning model can be continuously updated to improves the accuracy of the model. Compared to the traditional machine learning pipeline, online continuous learning carries our training process using arriving data, instead of waiting for all data collected [56]. By updating the learning model, the online continuous learning strategy can improve the accuracy of machine learning models over time. By doing this, the online continuous learning strategy can reduce training time and cost while a level of training model accuracy can be achieved, which is particular important to latency sensitive systems.

As an example, the IoT naming system can assist an autonomous vehicle in finding the resources that it needs. In such an system, the autonomous vehicle must interact with drivers or passengers and understand their destination and goals. This can be achieved through the use of nature language processing (NLP), which can assist the IoT search engine in automatically translating a URL from human language. Generally speaking, NLP is a technique that can help computers understand, interpret, and manipulate human language [73] through tasks such as translation, summarization, named entity recognition, relation extraction, speech recognition, and topic segmentation, among others. Moreover, a variety of tools have been designed and implemented to carry out natural language processing, such as spaCy and Nature Language Toolkit (NLTK) [14], [74]. Meanwhile, multiple pre-trained models like BERT, Roberta, and GPT-3, among others act as libraries in spaCy called ''spacy-transformers'', which improves the accuracy for NLP [75].

To demonstrate the use of NLP in the IoT search engine, we return to our smart parking as an example, as shown in Fig. 5, a vehicle is arriving to its destination (Micro Center) needs to find a parking lot. Since the autonomous vehicles are manufactured by different

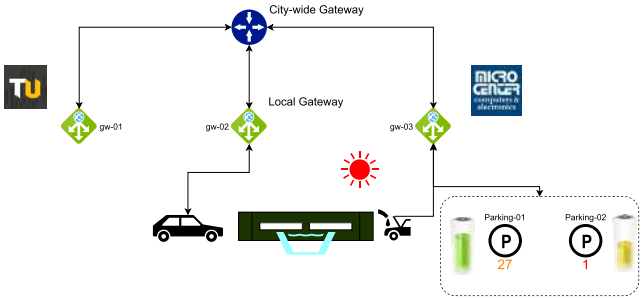

FIGURE 5. Smart parking.

producers, they might send out the request in unformatted machine message or leverage the human language directly. Taking the human language sent by the vehicle as an example, the vehicle may send a message to gw-03 with the content, ''available parking lots near Micro Center in 1957E Joppa Road, Towson MD''. When gw-03 receives the message, it will send it to the NLP module. The NLP module will first use a convolutional neural network (CNN) to classify the geo-location information in this sentence and ''1957E Joppa Road, Towson, MD'' will be automatically extracted from the entire sentence. Then, the NLP module will extract words with noun tagging, which are ''parking lot'' to match the resource type database located in the gw-03 storage. When the resource type matches, the request will be transmitted into a URL Sensor:// Garage@ gw-03.towson.baltimore:JoppaRoad:Public:Parking. Then, gw-03 will process the request and find an available parking lot between parking-01 and parking-02 based on the availability of parking spaces. Since the parking01 has more parking spaces, gw-03 will send URL Sensor://Garage@ gw-03.towson.baltimore:JoppaRoad:Public: Parking-01 to the vehicle and guide the vehicle to its destination.

C. OTHERS

Other open research problems in IoT search include the standardization of IoT search policies and protocols, as well as investigation of security and privacy technologies. Compared to web-based search, as mentioned above, the applications that use IoT search engines may be automated, which means that the interactions between users and IoT search engines are more likely to be machine-type communications. To this end, it is critical to develop a standardized application interface and protocols especially for machine-type communication.

In addition, security and privacy are critical. We have already seen significant evidence of vulnerabilities in IoT due in part to the limitation of devices, as well as the lack of security features in consumer devices [15], [24]. It is imperative that user privacy and information not to be leaked or exposed by IoT search engines, and that location analysis and the extraction of search habits be prevented by key security features in the IoT search engine.

V. CASE STUDY: LSTM-BASED QUERY PREDICTION

As there are numerous open research questions that need resolution in order to achieve an efficient and intelligent IoT

search engine. In the following, we present a case study that applies a specific machine learning technique to enable query prediction in an IoT search application. Particularly, we will introduce our approach to predict query volumes in our IoT search engine framework to provide ahead-of-time query and result caching. The following discussions includes the dataset, our workflow, and the detailed steps of our models.

A. DATASET

The data that we consider in this study comes from two primary sources. First, we utilize free open-source data available from the Aarhus, Denmark local government and collected in 2014 [76]. This data provides counts of the numbers of vehicles in 8 separate parking lots located in Denmark. Second, we supplement this data with free open-source data that we collected from the city of Silver Spring, Maryland, USA. This data includes the numbers of available parking spaces in four separate parking garages in the city [77]. Ultimately, we convert this data to a unified format (available parking spaces), and use this data as an analog for real IoT search engine query request volume.

1) SOURCE DATA

In further detail, we note that the Aarhus Dataset has been used in multiple studies and for various purposes [78], [79]. It is representative of a typical IoT application, for which users would desire up-to-date, real-time information when making traveling decisions. At the same time, we note that the dataset in its original time scale is limited, as half-hour data publishing is generally not appropriate for real-time applications.

Considering the data collected from the city of Silver Spring, Maryland, USA, the data is more frequently published and is likewise representative of a realistic IoT application. In this case, we can consider the time scale to be more closely tailored to real-time needs, as the data is published every minute. This especially serves commuters, who would be interested in the availability of parking near their workplaces.

In transforming this data for our application, we consider the volume of parking spaces (filled or vacant) as representative and analogous to the periodic increase and decreased need for a particular resource. Thus, we analogize the parking volume to query volume over time, enabling our evaluation of volume prediction. In addition, we note that the differences in frequency between datasets allow us to consider diverse needs of machine/human users in the IoT search engine.

2) PREPROCESSING

Several steps were taken to preprocess the existing datasets, which are described below.

- First, we deduplicated the data in the Silver Spring dataset, which was collected at a higher frequency than the data was generated.
- Second, the Aarhus dataset was converted to the same format as the Silver Spring dataset. Specifically, the volume of vehicles in the parking lots were converted to available spaces (the format of the Silver Spring dataset). Given that for some lots, there were more vehicles than spaces (vehicles looking for spaces but unable to find them), the conversion required that available spaces be limited to no less than 0.
- Third, the Aarhus dataset was collected every 30 minutes, while the Silver Spring data was collected every 1 minute. This required a choice of how to combine the data for evaluation. In this case, we chose to match the Aarhus dataset to the Silver Spring time scheme to provide contrasting data distributions against which to test.
- Fourth, upon evaluation, the data of the eighth lot in the Aarhus dataset was of a binary nature, inconsistent with the other lots in either dataset. For this reason, we expelled the eighth lot from Aarhus, resulting in eleven total lots/garages.
- Finally, we can see the data, prior to normalization in Figure V-A. Note that the first four curves are garages from the Silver Spring dataset, and the last seven are from the Aarhus dataset. As the last step of preprocessing, we normalize our data for training our learning models. After the models are trained and validated, the output data must be denormalized to extract the actual values predicted.

B. WORKFLOW

In developing our Query Engine module, we have followed a typical data science pipeline of data ingest, preprocessing, training, validation, and postprocessing. Our work flow is generic, using Python and Tensorflow, and can be generalized to other languages (Java, etc.) and other deep learning libraries/frameworks (PyTorch, DeepLearning4J, etc.). Data is ingested from .csv files as dataframes for ease of bulk data manipulation, before being conversion to numpy arrays for input into out LSTM model.

Our deep learning model (LSTM) design is generic, and performance is only moderately tuned, meaning additional accuracy is achievable. More specifically, the design of the model should be tailored to a real-world system implementation based on a number of factors.

- First, the model training of LSTM, as a typical recurrent neural network (RNN), is computationally expensive on consumer hardware, generally requiring Graphics Processing Unit (GPU)/Tensor Processing Unit (TPU) acceleration. The hardware dedicated to a distributed hierarchical IoT search engine may be variable, and thus the number of models trainable and deployable to the IoT search engine must be determined with a variety of trade-offs in mind (power consumption, number of CPUs/GPUs/TPUs and their power, cost, etc.).
- Second, the query space may be potentially unlimited, and thus the need to constrain the number predicted queries will likely arise. The training of multiple deep recursive neural networks simultaneously for either

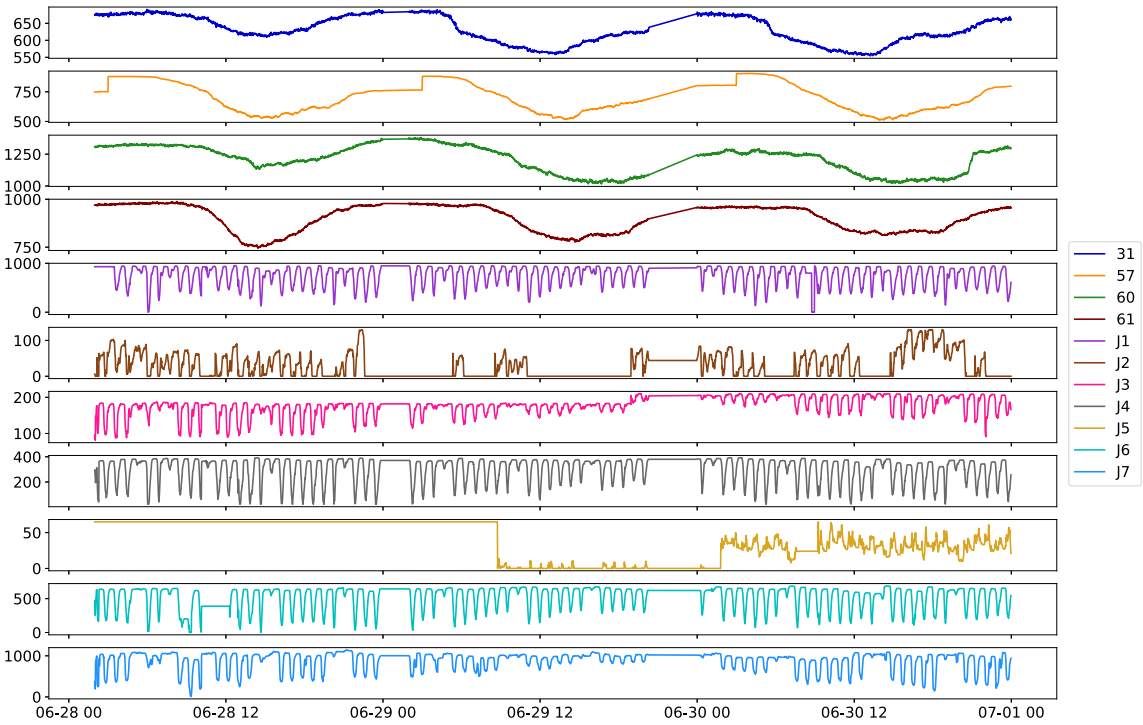

FIGURE 6. Full dataset.

single value or multi-value prediction makes this problem prohibitively expensive and complex. To combat this, we evaluate the prediction of multiple singular time series as well as the combined prediction of all time series, the details of which are outlined in the next subsection.

Training of our LSTM models is conducted to convergence, and accuracy validated. With the model trained, postprocessing extracts the actual prediction values from the normalized output (i.e., denormalizing occurs) for each individual query.

C. LEARNING MODEL

In this work, we utilize the LSTM as our deep learning model. The LSTM has been demonstrated to be one of the top performers in terms of time-series prediction [80]–[105]. This is due to the capacity to retain more information over time through a combination of forget, input, and output gates embedded in the LSTM structure. Generalized in Figure 7, the LSTM structure retains information in the form of cell states C and hidden states h. The forget (first σ from left to right), input (second σ and tanh), and output (third σ and second tanh) gates control what old information is discarded, what new information is inserted, and what is passed along to the next LSTM unit, respectively. Each of these gates operate through use of the sigmoid function, which converts input to a range of 0 to 1, with 0 being unimportant and 1 being important (i.e., keep). Likewise, the hyperbolic

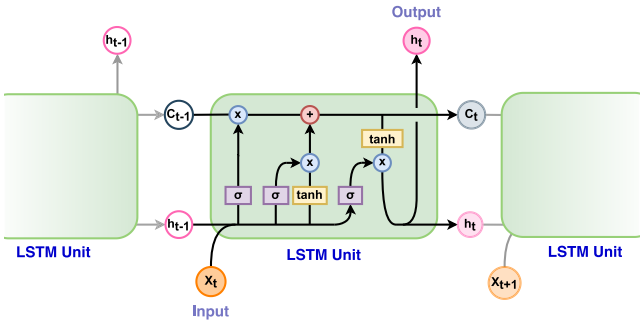

FIGURE 7. Generalized LSTM structure.

tangent (*tanh*) acts in a similar manner, converting input to a range of -1 to 1.

In our learning models, because we are testing several different network shapes, we note simply that we utilize typical LSTM RNN structures with few LSTM layers having approximately 32 units, and a fully-connected dense output layer to obtain our target output dimensions. This method is generic, and can be implemented in a variety of languages and libraries. Our specific implementation uses the Keras LSTM, which is integrated with Tensorflow 2.

VI. PERFORMANCE EVALUATION OF LSTM-BASED QUERY PREDICTION

Based on our approach described in Section V, we now provide the details of our evaluation. We first detail our

evaluation methodology (environment, metrics, and evaluation criteria) and then show our results.

A. METHODOLOGY

1) ENVIRONMENT

For our evaluation environment, we are using a single consumer-grade custom PC/Linux computer with the following hardware: AMD Ryzen 5 3600X (6 cores, 12 threads, 3.8/4.4 GHz), 32 GB RAM (DDR4-3600), RTX 2070 SUPER (8 GB VRAM), and 1000 GB NVMe M.2 SSD. Our evaluation environment is running CentOS 8 Linux, and experiments were run in a Jupyter Notebook server running in docker (Version 19.03.8), with NVIDIA Container Toolkit, in a custom image built upon the official tensorflow: latest-gpu-py3-jupyter image. The environment includes Python 3.6.9 and Tensorflow 2.1.0.

We note that our environment is constrained in several ways in comparison to a fully realized IoT search engine. Particularly, a real-world IoT search engine environment would be composed of many heterogeneous servers of varying computational power, likely much more effective than the hardware environment used herein. In addition, our goal here is to demonstrate the viability of interchangeable software modules in improving and expanding the performance of an IoT search engine as described. Such modules could be written in other programming languages or combinations of languages, run in containers or virtual machines, and could serve neural network models directly. Given the potential for multiple cooperative Query Engines, we can safely assume that a fully realized IoT search architecture would subdivide available hardware resources for various modules and tasks in a typical modern micro-service architecture. We thus consider that our evaluation conditions are not so far from reality as to be irrelevant.

2) METRICS AND CRITERIA

The primary metric of our evaluation is Mean Absolute Error (MAE). This is a typical evaluation metric for time-series prediction [103]–[105] that measures the average magnitude of errors in a set of predictions. MAE can also be used as the training metric which is improved upon with every epoch. The formula for MAE is as follows:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_{f,i} - y_{o,i}|, \tag{1}$$

where $y_{f,i}$ is the i^{th} forecast, and $y_{o,i}$ is the i^{th} observation. That is, $|y_{f,i} - y_{o,i}|$ is absolute error, and the mean of absolute error is take over all predictions/observations.

B. RESULTS

We now detail the results of our experiment. In predicting the query volumes for our simulated IoT search engine, we are ultimately interested in accurately predicting the volumes of all queries simultaneously. This is to reduce the load of maintaining multiple LSTM models, one for each query, and running them concurrently, consuming significant resources.

Thus, we now detail our various approaches to achieving prediction of query volume. First, considering quite simplistic models, we evaluate the training and testing of one model for each target query. In this case, using the available parking spaces as the volume, we end up training and validating 11 total LSTM models. We note that these models result in the lowest error of all models. We hypothesized that providing multiple queries as inputs for single-output and multi-output prediction might improve accuracy (i.e., error reduction), due to interactions between the different queries allowed in the networks. We instead found that the addition of other queries as input actually increased the error, as there are no strong correlations between these data distributions. We note that the addition of alternative data of a different yet relevant domain has been shown in the past to increase accuracy, as in the addition of new data modals for multi-modal learning. These operate well due to the correlation of the added modals. For instance, we can improve prediction on some datasets related to travel or purchasing by adding weather and temperature conditions, because the activities of people that generated the datasets will have been hindered by bad weather and seasonality. In our case, while we found that much of the data follows similar harmonic variation, this did not translate to improved performance.

With this result in mind, we must still make the assertion that training and evaluating one model for each query is highly inefficient and wasteful. Thus, we need to consider the training and testing of multiple input data for the simultaneous prediction of all queries in the next time step. To achieve this, we can modify our input shape and our output neurons to match. Similar to the implementation of prediction over multiple time steps, we can simultaneously predict multiple output targets. Ultimately, we implemented first two separate models to predict our separate datasets that represent queries in this context, and finally we implemented a single model to take 11 sequential sets and predict the next datapoint for each. Note that the ''G'' represents the Silver Spring dataset, while the ''J'' represents the Aarhus dataset.

The results of all of out outputs can be seen in Table 1. Here, we provide the MAE for training and validation of our models. Clearly, we encountered significantly higher error in the Aarhus dataset compared to the Silver Spring dataset. We can consider that the error we calculate is an aggregate of all output targets, and thus would approximate the combination of each individual model. However, we note that the model with all four inputs/outputs from the Silver Spring dataset shows higher error in testing than each of the individual models. Thus, it seems apparent that the interactions among the models likely contribute to increased error, instead of reduced error. Ultimately, we must consider the trade-offs between marginally lower error and computation time/complexity. In the case of training each individual model, while a single input model is faster than a multi-input model, the multi-input model can be run as a single unit when

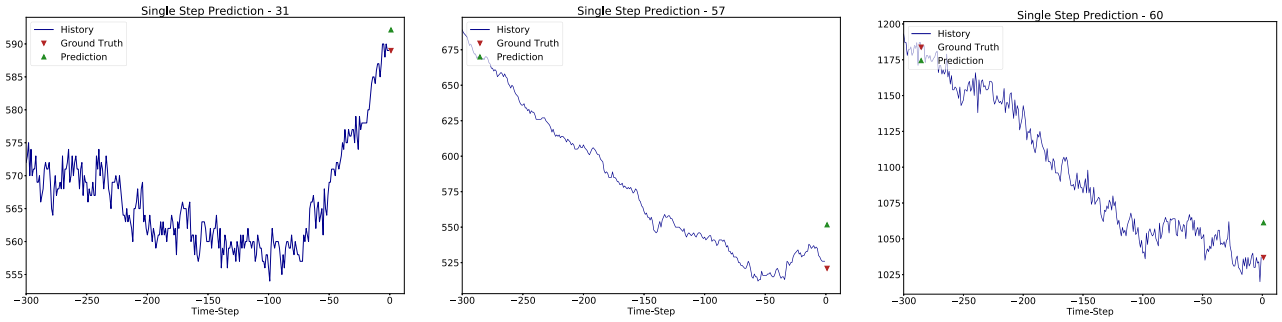

FIGURE 8. Individual predictions for queries represented by G31, G57, and G60.

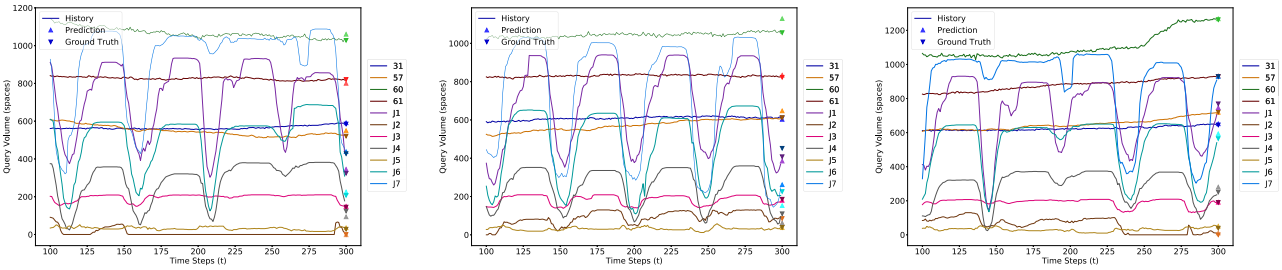

FIGURE 9. Multiple simultaneous predictions for all 11 queries represented by G31-J7. The three figures show three different predictions based on different time windows (all of length 300). The figure shorten the time window for clarity.

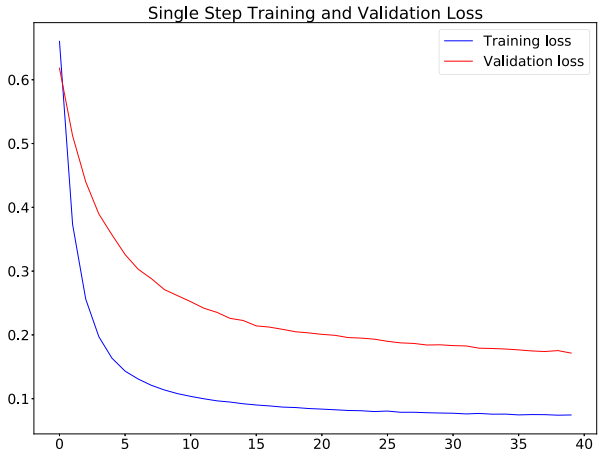

FIGURE 10. Training and validation loss of final model consisting of 11 simultaneous sequential datasets as input, and 11 output predictions.

served to a query engine. In contrast, multiple single-input models (11 in this case) would have to run concurrently to achieve the same output.

VII. RELATED WORK

In this section, we consider works related to the topics of IoT search and search engines, as well as generic search and search engines, and advancements in deep learning.

A. IoT SEARCH AND SEARCH ENGINE

Search engines for IoT and general search of IoT systems and devices have been considered for some time. However,

TABLE 1. Mean absolute error (MAE) results for training and validation of neural network models.

Input/Output	Training	Validation
Single G31	0.0528	0.0553
Single G57	0.0213	0.0180
Single G60	0.0435	0.0503
Single G61	0.0333	0.0315
Group G31-G61	0.0394	0.0657
Group J1-J7	0.0960	0.1615
All G31-J7	0.0703	0.1630

research dedicated to framework design and practical implementation of IoT search systems has been relatively sparse. For instance, as far back as 2012, Ding *et al.* [38] proposed a hybrid search engine framework, which they denoted their IoT-SVK. Here, SVK stands for Spatial-Temporal, Value-based, and Keyword-based search conditions. In their framework, they proposed a three-layer architecture of devices, storage, and indexing. In addition, they provided some algorithmic optimizations for optimizing search over their indexing method using various B^+- and R- tree structures. Lunardi *et al.* [39], in 2015, constructed a different framework for IoT search, which they named COBASEN (COntext BAsed Search ENgine), based upon COMPaaS (Cooperative Middleware Platform as a Service). Their concept was based around a service-oriented architecture approach, and was composed of two main constituents: context module and search engine. In their work, they developed a prototype system, and evaluated its capacity to scale with device volume.

Barnaghi and Sheth [40], in 2016, presented a collection of requirements and challenges to IoT search. The challenges include the heterogeneity of network interfaces, heterogeneity of devices, heterogeneity of data, and many classical big data challenges (volume, velocity, variety, security, and automation). The requirements are many, and include frequent updates, machine interpretable structures and queries, efficiency and scalability in indexing, and search and discovery of distributed resources, among others. Ma and Liu [37], in 2018, proposed a series of broad IoT search characteristics and challenges, and observed IoT systems from the perspectives of data acquisition, analysis, and utilization. They also provided two case studies: progressive vehicle search and progressive person search, which they evaluated using their developed prototype progressive search system, denoted PROVINS.

B. QUERY AGGREGATION AND PREDICTION

Research targeting the aggregation of queries and the prediction of queries has been relatively prolific, and have sought to improve and optimize the performance of search, index, and storage systems [44], [45], [106]. With respect to query aggregation, An [44] developed a method for query aggregation in wireless sensor networks that combines query simplification and merging. In addition, Sarode and Nandhini [45] developed adaptive pruning and data aggreagation (APDA) for query-based wireless sensor networks. Their work seeks to minimize query response time by establishing Dominator nodes that perform adaptive pruning and aggregation to perform the task of finding the highest-K values (application dependent) and source sensor IDs.

There are several types of queries in query processing and aggregation, including location-based query [107], content-based query [108], and heterogeneous query [109]. Note that the location information of IoT data is important and queries to IoT systems could be interested in data related to certain geographical area. The query can be content-based if the query should be conducted based on the data content [110]. Heterogeneous query includes semantic or ontology-based search and resource or service retrieval [111].

In the case of query prediction, a variety of works have been proposed, though the topic is not as vibrant. For instance, Akdere *et al.* [112] evaluated predictive modeling techniques for Query Performance Prediction (QPP) on via support vector machine (SVM) and Kernel Canonical Correlation Analysis (KCCA) on plan-level modeling, and multiple linear regression (MLR) operator-level modeling. Also, He *et al.* [113] proposed a web recommendation method based on sequential query prediction. In their work, they evaluated N-gram and Variable Markov Models (VMM), and proposed a Mixture Variable Markov (MVMM) Model as well.

C. MACHINE LEARNING ON SEQUENTIAL DATA

Significant work has been done in every variety of applications to apply and improve deep learning. Regarding time-series prediction specifically, a variety of research has been conducted. For instance, Song *et al.* [80] combined LSTM and Kalman Filter models for air quality prediction, achieving lower RMSE than the representative LSTM alone. Hajiaghayi and Vahedi [87] utilized LSTM models prediction and pattern extraction of code failure. In addition, Shu *et al.* [82] proposed a multi-LSTM or LSTM-in-LSTM mechanism for group activity recognition. Denoted as GLIL (Graph LSTM-In-LSTM), they utilized LSTMs for person-level activity recognition, which reside in a global graph LSTM for group-level recognition.

Heryadi and Warnars [84] utilized a variety of CNN and LSTM models for fraudulent transaction detection through learning short- and long-term representations of transaction history. Their data was highly imbalanced, with binary labels (fraud/not fraud), and their results showed the CNN result in the best AUC/ROC characteristics and the highest testing accuracy, while the LSTM and hybrid CNN-LSTM models achieved the highest training accuracy, despite poorer testing performance. Li *et al.* [89] utilized LSTM to predict channel state information, accelerating performance through the use of stochastic computing. They developed stochastic implementations for input processing, a universal gate (which can implement forget, inputs, and output gates), multiplier, adder, and output processing. In addition, experiments on a Xilinx FPGA chip platform demonstrate resource consumption of both the traditional and stochastic LSTM implementations, with the stochastic method consuming significantly less resources. Li and Lu [85] developed a distributed denial of service attack detection mechanism based upon LSTM and Bayes.

Xie and Wen [90] proposed LSTM-MA, an LSTM model that includes multi-modality and adjacency constraints for segmentation of human brain MRI images. Their model uses pixel-wise and super-pixel-wise constraints, and has been implemented in LSTM and BiLSTM (bidirectional), both of which demonstrated superior performance over a variety of traditional algorithms, though other neural network models were not compared. Jo *et al.* [88] designed an LSTM implementation scheme to reduce power consumption. Specifically, taking the DeepSpeech network as a baseline model, they selectively implemented the LSTM nodes as strong and weak in the bidiractional LSTM structure, where strong nodes use 19-bit fixed point encoding, and weak nodes use 7-bit fixed point encoding. Kumar and Subha [83] applied LSTM to predict periods of depression in patient EEG signals. Akandeh and Salem [81] developed ''slim'' LSTM model designs, which require less computational resources through the implementation of constant gate signals and reduced adaptive parameters. Likewise, Chakraborty *et al.* [86] studied the effect of the number of LSTM cells used to construct a model on performance in character prediction. They found that convolutional LSTM models are generally much more stable (less prone to over-fitting) than traditional LSTM models, and that the traditional LSTM models they use clearly begin to overfit at approximately 100 nodes.

VIII. FINAL REMARKS

In this article, we have considered a generic framework for the IoT search engine, and studied potential modules that would improve performance, enabling the IoT search engine to cope with the massive volume of requests expected from human and machine users. We have also proposed the naming service for IoT search engine, and presented challenges and potential research directions for designing efficient and intelligent IoT search engine. Further, as a case study, we have designed the LSTM-based query prediction for improving the IoT search engine. Particularly, reducing the load caused by massively replicated equivalent queries can be achieved through the use of query aggregation in combination with predictive query execution. To achieve such a query prediction, we have implemented LSTM RNNs to simultaneously predict multiple queries. While less accurate than individual networks for each predicted query, simultaneous prediction reduces computation overhead, while still affording relatively low error.

ACKNOWLEDGMENT

Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding agency.

REFERENCES

[1] Y. Krytska, I. Skarga-Bandurova, and A. Velykzhanin, "Iot-based situation awareness support system for real-time emergency management," in *Proc. 9th IEEE Int. Conf. Intell. Data Acquisition Adv. Comput. Syst., Technol. Appl. (IDAACS)*, vol. 2, Dec. 2017, pp. 955–960.

[2] S. Alamgir Hossain, M. Anisur Rahman, and M. A. Hossain, "Edge computing framework for enabling situation awareness in IoT based smart city," *J. Parallel Distrib. Comput.*, vol. 122, pp. 226–237, Dec. 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0743731518306130

[3] M. Chernyshev, Z. Baig, O. Bello, and S. Zeadally, "Internet of Things (IoT): Research, simulators, and testbeds," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 1637–1647, Jun. 2018.

[4] G. Xu, Y. Cao, Y. Ren, X. Li, and Z. Feng, "Network security situation awareness based on semantic ontology and user-defined rules for Internet of Things," *IEEE Access*, vol. 5, pp. 21046–21056, 2017.

[5] B. Hamdaoui, M. Alkalbani, A. Rayes, and N. Zorba, "IoT share: A blockchain-enabled IoT resource sharing on-demand protocol for smart city situation-awareness applications," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 10548–10561, Oct. 2020.

[6] B. Cheng, M. Wang, S. Zhao, Z. Zhai, D. Zhu, and J. Chen, "Situation-aware dynamic service coordination in an IoT environment," *IEEE/ACM Trans. Netw.*, vol. 25, no. 4, pp. 2082–2095, Aug. 2017, doi: 10.1109/TNET.2017.2705239.

[7] A. Parkhomenko, O. Bilov, A. Tulenkov, A. Sokolyanskii, Y. Zalyubovskiy, K. Henke, and H.-D. Wuttke, "Virtual model for remote laboratory smart house & IoT," in *Proc. 10th IEEE Int. Conf. Intell. Data Acquisition Adv. Comput. Syst., Technol. Appl. (IDAACS)*, Sep. 2019, pp. 985–990.

[8] J. Saha, A. K. Saha, A. Chatterjee, S. Agrawal, A. Saha, A. Kar, and H. N. Saha, "Advanced IOT based combined remote health monitoring, home automation and alarm system," in *Proc. IEEE 8th Annu. Comput. Commun. Workshop Conf. (CCWC)*, Jan. 2018, pp. 602–606.

[9] N. Fujii and N. Koike, "IoT remote group experiments in the cyber laboratory: A FPGA-based remote laboratory in the hybrid cloud," in *Proc. Int. Conf. Cyberworlds (CW)*, Sep. 2017, pp. 162–165.

[10] W. Wang, P. Xu, and L. T. Yang, "Secure data collection, storage and access in cloud-assisted IoT," *IEEE Cloud Comput.*, vol. 5, no. 4, pp. 77–88, Jul. 2018.

[11] N. Fotiou, V. A. Siris, A. Mertzianis, and G. C. Polyzos, "Smart IoT data collection," in *Proc. IEEE 19th Int. Symp. (WoWMoM)*, Jun. 2018, pp. 588–599.

[12] A. Roukounaki, S. Efremidis, J. Soldatos, J. Neises, T. Walloschke, and N. Kefalakis, "Scalable and configurable End-to-End collection and analysis of IoT security data : Towards End-to-End security in IoT systems," in *Proc. Global IoT Summit (GIoTS)*, Jun. 2019, pp. 1–6.

[13] A. P. Plageras, K. E. Psannis, C. Stergiou, H. Wang, and B. B. Gupta, "Efficient IoT-based sensor BIG data collection–processing and analysis in smart buildings," *Future Gener. Comput. Syst.*, vol. 82, pp. 349–357, May 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S016773917314127

[14] W. Yu, F. Liang, X. He, W. Grant Hatcher, C. Lu, J. Lin, and X. Yang, "A survey on the edge computing for the Internet of Things," *IEEE Access*, vol. 6, pp. 6900–6919, 2018.

[15] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, "A survey on Internet of Things: Architecture, enabling technologies, security and privacy, and applications," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1125–1142, Oct. 2017.

[16] P. Bhatia, S. Rajput, S. Pathak, and S. Prasad, "IOT based facial recognition system for home security using LBPH algorithm," in *Proc. 3rd Int. Conf. Inventive Comput. Technol. (ICICT)*, Nov. 2018, pp. 191–193.

[17] W.-J. Chang, M. Schmelzer, F. Kopp, C.-H. Hsu, J.-P. Su, L.-B. Chen, and M.-C. Chen, "A deep learning facial expression recognition based scoring system for restaurants," in *Proc. Int. Conf. Artif. Intell. Inf. Commun. (ICAIIC)*, Feb. 2019, pp. 251–254.

[18] X. Ge, R. Zhou, and Q. Li, "5g nfv-based tactile Internet for mission-critical iot services," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6150–6163, Dec. 2020.

[19] M. Tang, S. Cai, and V. K. N. Lau, "Remote state estimation with asynchronous mission-critical IoT sensors," *IEEE J. Sel. Areas Commun.*, early access, Aug. 24, 2020, doi: 10.1109/JSAC.2020.3018800.

[20] X. Carpent, K. Eldefrawy, N. Rattanavipanon, A.-R. Sadeghi, and G. Tsudik, "Invited: Reconciling remote attestation and safety-critical operation on simple IoT devices," in *Proc. 55th ACM/ESDA/IEEE Design Autom. Conf. (DAC)*, Jun. 2018, pp. 1–6.

[21] H. Xu, W. Yu, D. Griffith, and N. Golmie, "A survey on industrial Internet of Things: A cyber-physical systems perspective," *IEEE Access*, vol. 6, pp. 78238–78259, 2018.

[22] J. Lin, W. Yu, X. Yang, Q. Yang, X. Fu, and W. Zhao, "A novel dynamic en-route decision real-time route guidance scheme in intelligent transportation systems," in *Proc. IEEE 35th Int. Conf. Distrib. Comput. Syst.*, Jun. 2015, pp. 61–72.

[23] R. Hussain and S. Zeadally, "Autonomous cars: Research results, issues, and future challenges," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 2, pp. 1275–1313, 2nd Quart., 2019.

[24] X. Liu, C. Qian, W. G. Hatcher, H. Xu, W. Liao, and W. Yu, "Secure Internet of Things (IoT)-based smart-world critical infrastructures: Survey, case study and research opportunities," *IEEE Access*, vol. 7, pp. 79523–79544, 2019.

[25] Q. Yang, J. Yang, W. Yu, D. An, N. Zhang, and W. Zhao, "On false data-injection attacks against power system state estimation: Modeling and countermeasures," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 3, pp. 717–729, Mar. 2014.

[26] J. Lin, W. Yu, and X. Yang, "Towards multistep electricity prices in smart grid electricity markets," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 1, pp. 286–302, Jan. 2016.

[27] J. Lin, W. Yu, N. Zhang, X. Yang, and L. Ge, "Data integrity attacks against dynamic route guidance in transportation-based cyber-physical systems: Modeling, analysis, and defense," *IEEE Trans. Veh. Technol.*, vol. 67, no. 9, pp. 8738–8753, Sep. 2018.

[28] N. Neshenko, E. Bou-Harb, J. Crichigno, G. Kaddoum, and N. Ghani, "Demystifying IoT security: An exhaustive survey on IoT vulnerabilities and a first empirical look on Internet-scale IoT exploitations," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2702–2733, 3rd Quart., 2019.

[29] R. Williams, E. Mcmahon, S. Samtani, M. Patton, and H. Chen, "Identifying vulnerabilities of consumer Internet of Things (IoT) devices: A scalable approach," in *Proc. IEEE Int. Conf. Intell. Secur. Informat. (ISI)*, Jul. 2017, pp. 179–181.

[30] W. Gao, J. H. Nguyen, W. Yu, C. Lu, D. T. Ku, and W. G. Hatcher, "Toward emulation-based performance assessment of constrained application protocol in dynamic networks," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1597–1610, Oct. 2017.

[31] F. Liang, W. Yu, D. An, Q. Yang, X. Fu, and W. Zhao, "A survey on big data market: Pricing, trading and protection," *IEEE Access*, vol. 6, pp. 15132–15154, 2018.

[32] Z. Cai and Z. He, "Trading private range counting over big IoT data," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2019, pp. 144–153.

[33] W. Gao, W. Yu, F. Liang, W. G. Hatcher, and C. Lu, "Privacy-preserving auction for big data trading using homomorphic encryption," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 2, pp. 776–791, Apr. 2020.

[34] X. Zheng and Z. Cai, "Privacy-preserved data sharing towards multiple parties in industrial IoTs," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 5, pp. 968–979, May 2020.

[35] X. Yang, T. Wang, X. Ren, and W. Yu, "Survey on improving data utility in differentially private sequential data publishing," *IEEE Trans. Big Data*, early access, Jun. 15, 2017, doi: 10.1109/TBDATA.2017.2715334.

[36] F. Liang, C. Qian, W. G. Hatcher, and W. Yu, "Search engine for the Internet of Things: Lessons from Web search, vision, and opportunities," *IEEE Access*, vol. 7, pp. 104673–104691, 2019.

[37] H. Ma and W. Liu, "A progressive search paradigm for the Internet of Things," *IEEE Multimedia Mag.*, vol. 25, no. 1, pp. 76–86, Jan. 2018.

[38] Z. Ding, X. Gao, L. Guo, and Q. Yang, "A hybrid search engine framework for the Internet of Things based on spatial-temporal, value-based, and keyword-based conditions," in *Proc. IEEE Int. Conf. Green Comput. Commun.*, Nov. 2012, pp. 17–25.

[39] W. T. Lunardi, E. de Matos, R. Tiburski, L. A. Amaral, S. Marczak, and F. Hessel, "Context-based search engine for industrial IoT: Discovery, search, selection, and usage of devices," in *Proc. IEEE 20th Conf. Emerg. Technol. Factory Autom. (ETFA)*, Sep. 2015, pp. 1–8.

[40] P. Barnaghi and A. Sheth, "On searching the Internet of Things: Requirements and challenges," *IEEE Intell. Syst.*, vol. 31, no. 6, pp. 71–75, Nov. 2016.

[41] W. G. Hatcher and W. Yu, "A survey of deep learning: Platforms, applications and emerging research trends," *IEEE Access*, vol. 6, pp. 24411–24432, 2018.

[42] M. Mohammadi, A. Al-Fuqaha, S. Sorour, and M. Guizani, "Deep learning for IoT big data and streaming analytics: A survey," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 2923–2960, 4th Quart., 2018.

[43] S. Zeadally, E. Adi, Z. Baig, and I. A. Khan, "Harnessing artificial intelligence capabilities to improve cybersecurity," *IEEE Access*, vol. 8, pp. 23817–23837, 2020.

[44] Z. An, "Query aggregation algorithm in wireless sensor networks," in *Proc. IEEE Symp. Electr. Electron. Eng. (EEESYM)*, Jun. 2012, pp. 651–654.

[45] P. Sarode and R. Nandhini, "APDA: Adaptive pruning & data aggregation algorithms for Query based wireless sensor networks," in *Proc. Int. Conf. Global Trends Signal Process., Inf. Comput. Commun. (ICGTSPICC)*, Dec. 2016, pp. 219–224.

[46] H. Yan, N. Al-Hoqani, and S.-H. Yang, "In-network multi-sensors Query aggregation algorithm for wireless sensor networks database," in *Proc. IEEE 15th Int. Conf. Netw., Sens. Control (ICNSC)*, Mar. 2018, pp. 1–8.

[47] R. Deepika, A. Latha, and S. Jayashri, "Secure pattern based aggregation using Query processing algorithm in wireless sensor networks," in *Proc. Int. Conf. Commun. Signal Process.*, Apr. 2013, pp. 1064–1068.

[48] L. Liu, X. Qin, B. Li, Y. Liu, and G. Zheng, "Energy-efficient and robust spatial window aggregation Query processing algorithm in wireless sensor networks," in *Proc. 32nd Int. Conf. Distrib. Comput. Syst. Workshops*, Jun. 2012, pp. 169–177.

[49] Y. Zhao, Z. Shi, J. Zhang, D. Chen, and L. Gu, "A novel active learning framework for classification: Using weighted rank aggregation to achieve multiple Query criteria," *Pattern Recognit.*, vol. 93, pp. 581–602, Sep. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320319301372

[50] J. Du, Q. Liu, K. Chen, and J. Wang, "Forecasting stock prices in two ways based on LSTM neural network," in *Proc. IEEE 3rd Inf. Technol., Netw., Electron. Autom. Control Conf. (ITNEC)*, Mar. 2019, pp. 1083–1086.

[51] M. S. Hegde, G. Krishna, and R. Srinath, "An ensemble stock predictor and recommender system," in *Proc. Int. Conf. Adv. Comput., Commun. Informat. (ICACCI)*, Sep. 2018, pp. 1981–1985.

[52] Z. Liu, Z. Wang, X. Yin, X. Shi, Y. Guo, and Y. Tian, "Traffic matrix prediction based on deep learning for dynamic traffic engineering," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, Jun. 2019, pp. 1–7.

[53] R. Madan and P. S. Mangipudi, "Predicting computer network traffic: A time series forecasting approach using DWT, ARIMA and RNN," in *Proc. 11th Int. Conf. Contemp. Comput. (IC3)*, Aug. 2018, pp. 1–5.

[54] N. Kim, M. Kim, and J. K. Choi, "LSTM based short-term electricity consumption forecast with daily load profile sequences," in *Proc. IEEE 7th Global Conf. Consum. Electron. (GCCE)*, Oct. 2018, pp. 136–137.

[55] A. Tokgoz and G. Unal, "A RNN based time series approach for forecasting turkish electricity load," in *Proc. 26th Signal Process. Commun. Appl. Conf. (SIU)*, May 2018, pp. 1–4.

[56] F. Liang, W. G. Hatcher, G. Xu, J. Nguyen, W. Liao, and W. Yu, "Towards online deep learning-based energy forecasting," in *Proc. 28th Int. Conf. Comput. Commun. Netw. (ICCCN)*, Jul. 2019, pp. 1–9.

[57] P. Mockapetris and K. J. Dunlap, "Development of the domain name system," in *Proc. Symp. Proc. Commun. Archit. Protocols*, 1988, pp. 123–133.

[58] A. Afanasyev, J. Burke, T. Refaei, L. Wang, B. Zhang, and L. Zhang, "A brief introduction to named data networking," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, Oct. 2018, pp. 1–6.

[59] L. Zhang, A. Afanasyev, J. Burke, and V. Jacobson, "Named data networking," *SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 3, pp. 66–73, Jul. 2014.

[60] T. Berners-Lee. *Uniform Resource Identifier (URI): Generic Syntax*. Accessed: Oct. 2020. [Online]. Available: https://tools.ietf.org/html/rfc3986

[61] C. Ghasemi, H. Yousefi, K. G. Shin, and B. Zhang, "A fast and memory-efficient trie structure for name-based packet forwarding," in *Proc. IEEE 26th Int. Conf. Netw. Protocols (ICNP)*, Sep. 2018, pp. 302–312.

[62] R. Moats. *RFC 2141 URN*. Accessed: Oct. 2020. [Online]. Available: https://tools.ietf.org/html/rfc2141

[63] Xiaomi. *Xiaomi URN*. Accessed: Oct. 2020. [Online]. Available: https://gist.github.com/marcelrv/20509106e97b4c7dd8c9056aefda2332

[64] C. Qian, W. Gao, W. G. Hatcher, W. Liao, C. Lu, and W. Yu, "Search engine for heterogeneous Internet of Things systems and optimization," in *Proc. IEEE Intl Conf Dependable, Autonomic Secure Comput., Intl Conf Pervas. Intell. Comput., Intl Conf Cloud Big Data Comput., Intl Conf Cyber Sci. Technol. Congr. (DASC/PiCom/CBDCom/CyberSciTech)*, Aug. 2020, pp. 475–482.

[65] N. Ekedebe, C. Lu, and W. Yu, "Towards experimental evaluation of intelligent transportation system safety and traffic efficiency," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2015, pp. 3757–3762.

[66] T. Lin, H. Rivano, and F. Le Mouel, "A survey of smart parking solutions," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 12, pp. 3229–3253, Dec. 2017.

[67] D. An, Q. Yang, D. Li, W. Yu, W. Zhao, and C.-B. Yan, "Where am i parking: Incentive online parking-space sharing mechanism with privacy protection," *IEEE Trans. Autom. Sci. Eng.*, early access, Oct. 7, 2020, doi: 10.1109/TASE.2020.3024835.

[68] J. Wang, J. Liu, and N. Kato, "Networking and communications in autonomous driving: A survey," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 2, pp. 1243–1274, 2nd Quart., 2019.

[69] Q. Chen, X. Ma, S. Tang, J. Guo, Q. Yang, and S. Fu, "F-cooper: Feature based cooperative perception for autonomous vehicle edge computing system using 3D point clouds," in *Proc. 4th ACM/IEEE Symp. Edge Comput.*, Nov. 2019, pp. 88–100, doi: 10.1145/3318216.3363300.

[70] H. Xu, X. Liu, W. Yu, D. Griffith, and N. Golmie, "Reinforcement learning-based control and networking co-design for industrial Internet of Things," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 5, pp. 885–898, May 2020.

[71] F. Liang, W. G. Hatcher, W. Liao, W. Gao, and W. Yu, "Machine learning for security and the Internet of Things: The good, the bad, and the ugly," *IEEE Access*, vol. 7, pp. 158126–158147, 2019.

[72] F. Liang, W. Yu, X. Liu, D. Griffith, and N. Golmie, "Toward edge-based deep learning in industrial Internet of Things," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4329–4341, May 2020.

[73] A. Torfi, R. A. Shirvani, Y. Keneshloo, N. Tavaf, and E. A. Fox, "Natural language processing advancements by deep learning: A survey," 2020, *arXiv:2003.01200*. [Online]. Available: https://arxiv.org/abs/2003.01200

[74] F. N. A. Al Omran and C. Treude, "Choosing an NLP library for analyzing software documentation: A systematic literature review and a series of experiments," in *Proc. IEEE/ACM 14th Int. Conf. Mining Softw. Repositories (MSR)*, May 2017, pp. 187–197.

[75] G. Shekhar. *Understanding GPT-3: Openai's Latest Language Model*. Accessed: Oct. 2020. [Online]. Available: https://medium.com/swlh/understanding-gpt-3-openais-latest-language-mod el-a3ef89cffac2

[76] M. I. Ali, F. Gao, and A. Mileo, "Citybench: A configurable benchmark to evaluate rsp engines using smart city datasets," in *Proc. 14th Int. Semantic Web Conf.* Bethlehem, PA, USA: W3C, 2015, pp. 374–389.
[77] Montgomery County Government. (2020). *Data Montgomery*. [Online]. Available: https://data.montgomerycountymd.gov/
[78] J. Zenkert, M. Dornhofer, C. Weber, C. Ngoukam, and M. Fathi, "Big data analytics in smart mobility: Modeling and analysis of the aarhus smart city dataset," in *Proc. IEEE Ind. Cyber-Physical Syst. (ICPS)*, May 2018, pp. 363–368.
[79] A. R. Honarvar and A. Sami, "Multi-source dataset for urban computing in a smart city," *Data Brief*, vol. 22, pp. 222–226, Feb. 2019.
[80] X. Song, J. Huang, and D. Song, "Air quality prediction based on LSTM-Kalman model," in *Proc. IEEE 8th Joint Int. Inf. Technol. Artif. Intell. Conf. (ITAIC)*, May 2019, pp. 695–699.
[81] A. Akandeh and F. M. Salem, "Slim lstm networks: Lstm_6 and lstm_c6," in *2019 IEEE 62nd Int. Midwest Symp. Circuits Syst. (MWSCAS)*, 2019, pp. 630–633.
[82] X. Shu, L. Zhang, Y. Sun, and J. Tang, "Host-parasite: Graph LSTM-in-LSTM for group activity recognition," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Apr. 2, 2020, doi: 10.1109/TNNLS.2020.2978942.
[83] S. D. Kumar and D. Subha, "Prediction of depression from EEG signal using long short term Memory(LSTM)," in *Proc. 3rd Int. Conf. Trends Electron. Informat. (ICOEI)*, Apr. 2019, pp. 1248–1253.
[84] Y. Heryadi and H. L. H. S. Warnars, "Learning temporal representation of transaction amount for fraudulent transaction recognition using CNN, stacked LSTM, and CNN-LSTM," in *Proc. IEEE Int. Conf. Cybern. Comput. Intell. (CyberneticsCom)*, Nov. 2017, pp. 84–89.
[85] Y. Li and Y. Lu, "LSTM-BA: DDoS detection approach combining LSTM and bayes," in *Proc. 7th Int. Conf. Adv. Cloud Big Data (CBD)*, Sep. 2019, pp. 180–185.
[86] S. Chakraborty, J. Banik, S. Addhya, and D. Chatterjee, "Study of dependency on number of LSTM units for character based text generation models," in *Proc. Int. Conf. Comput. Sci., Eng. Appl. (ICCSEA)*, Mar. 2020, pp. 1–5.
[87] M. Hajiaghayi and E. Vahedi, "Code failure prediction and pattern extraction using LSTM networks," in *Proc. IEEE 5th Int. Conf. Big Data Comput. Service Appl. (BigDataService)*, Apr. 2019, pp. 55–62.
[88] J. Jo, S. Hwang, S. Lee, and Y. Lee, "Multi-mode LSTM network for energy-efficient speech recognition," in *Proc. Int. SoC Design Conf. (ISOCC)*, Nov. 2018, pp. 133–134.
[89] S. Li, Q. Wang, X. Liu, and J. Chen, "Low cost LSTM implementation based on stochastic computing for channel state information prediction," in *Proc. IEEE Asia Pacific Conf. Circuits Syst. (APCCAS)*, Oct. 2018, pp. 231–234.
[90] K. Xie and Y. Wen, "LSTM-MA: A LSTM method with multi-modality and adjacency constraint for brain image segmentation," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2019, pp. 240–244.
[91] A. Berhich, F.-Z. Belouadha, and M. I. Kabbaj, "LSTM-based models for earthquake prediction," in *Proc. 3rd Int. Conf. Netw., Inf. Syst. Secur.*, Mar. 2020, pp. 1–7, doi: 10.1145/3386723.3387865.
[92] S. Wang, Z. Li, C. Ding, B. Yuan, Q. Qiu, Y. Wang, and Y. Liang, "C-LSTM: Enabling efficient LSTM using structured compression techniques on FPGAs," in *Proc. ACM/SIGDA Int. Symp. Field-Program. Gate Arrays*, Feb. 2018, pp. 11–20, doi: 10.1145/3174243.3174253.
[93] Z. Liu, W. Zhou, and H. Li, "AB-LSTM: Attention-based bidirectional LSTM model for scene text detection," *ACM Trans. Multimedia Comput., Commun., Appl.*, vol. 15, no. 4, pp. 1–23, Jan. 2020, doi: 10.1145/3356728.
[94] Y. Chen, "CT-LSTM: Detection & estimation duplexed system for robust object tracking," in *Proc. 2nd Int. Conf. Comput. Sci. Appl. Eng.*, 2018, pp. 1–7, doi: 10.1145/3207677.3277985.
[95] R. Shi, J. Liu, H. K.-H. So, S. Wang, and Y. Liang, "E-LSTM: Efficient inference of sparse LSTM on embedded heterogeneous system," in *Proc. 56th Annu. Design Autom. Conf.*, Jun. 2019, pp. 1–6, doi: 10.1145/3316781.3317813.
[96] C. Du, P. Shu, and Y. Li, "CA-LSTM," in *Proc. 41st Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jun. 2018, pp. 1101–1104, doi: 10.1145/3209978.3210087.
[97] X. Zheng and W. Chen, "An attention-based bi-LSTM method for visual object classification via EEG," *Biomed. Signal Process. Control*, vol. 63, Jan. 2021, Art. no. 102174. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S174680942030313X
[98] Z. Shi and A. Chehade, "A dual-LSTM framework combining change point detection and remaining useful life prediction," *Rel. Eng. Syst. Saf.*, vol. 205, Jan. 2021, Art. no. 107257. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0951832020307572
[99] F. Shahid, A. Zameer, and M. Muneeb, "Predictions for COVID-19 with deep learning models of LSTM, GRU and bi-LSTM," *Chaos, Solitons Fractals*, vol. 140, Nov. 2020, Art. no. 110212. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0960077920306081
[100] M. Sarkar and A. De Bruyn, "LSTM response models for direct marketing analytics: Replacing feature engineering with deep learning," *J. Interact. Marketing*, vol. 53, pp. 80–95, Feb. 2021. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1094996820301080
[101] M. Sayah, D. Guebli, Z. Al Masry, and N. Zerhouni, "Robustness testing framework for RUL prediction deep LSTM networks," *ISA Trans.*, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0019057820302767, doi: 10.1016/j.isatra.2020.07.003.
[102] H. Yan, Y. Qin, S. Xiang, Y. Wang, and H. Chen, "Long-term gear life prediction based on ordered neurons LSTM neural networks," *Measurement*, vol. 165, Dec. 2020, Art. no. 108205. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0263224120307430
[103] C. Wang and Q. Gao, "High and low prices prediction of soybean futures with LSTM neural network," in *Proc. IEEE 9th Int. Conf. Softw. Eng. Service Sci. (ICSESS)*, Nov. 2018, pp. 140–143.
[104] A. Tongta and K. Chooruang, "Long short-term memory (LSTM) neural networks applied to energy disaggregation," in *Proc. 8th Int. Electr. Eng. Congr. (iEECON)*, Mar. 2020, pp. 1–4.
[105] H. Wang, Z. Guo, and L. Chen, "Financial forecasting based on LSTM and text emotional features," in *Proc. IEEE 8th Joint Int. Inf. Technol. Artif. Intell. Conf. (ITAIC)*, May 2019, pp. 1427–1430.
[106] W. Yu, T. Nam Le, D. Xuan, and W. Zhao, "Query aggregation for providing efficient data services in sensor networks," in *Proc. IEEE Int. Conf. Mobile Ad-hoc Sensor Syst.*, Oct. 2004, pp. 31–40.
[107] W. Wang, S. De, G. Cassar, and K. Moessner, "An experimental study on geospatial indexing for sensor service discovery," *Expert Syst. Appl.*, vol. 42, no. 7, pp. 3528–3538, May 2015.
[108] P. Zhang, Y.-A. Liu, F. Wu, and B. Tang, "Matching state estimation scheme for content-based sensor search in the Web of things," *Int. J. Distrib. Sensor Netw.*, vol. 11, no. 11, Nov. 2015, Art. no. 326780.
[109] S. Pattar, R. Buyya, K. R. Venugopal, S. S. Iyengar, and L. M. Patnaik, "Searching for the IoT resources: Fundamentals, requirements, comprehensive review, and future directions," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 3, pp. 2101–2132, 3rd Quart., 2018.
[110] P. Zhang, Y. Liu, F. Wu, S. Liu, and B. Tang, "Low-overhead and high-precision prediction model for content-based sensor search in the Internet of Things," *IEEE Commun. Lett.*, vol. 20, no. 4, pp. 720–723, Apr. 2016.
[111] L. Nunes, J. Estrella, L. Nakamura, R. de Libardi, C. Ferreira, L. Jorge, C. Perera, and S. Reiff-Marganiec, "A distributed sensor data search platform for Internet of Things environments," 2016, *arXiv:1606.07932*. [Online]. Available: http://arxiv.org/abs/1606.07932
[112] M. Akdere, U. Çetintemel, M. Riondato, E. Upfal, and S. B. Zdonik, "Learning-based Query performance modeling and prediction," in *Proc. IEEE 28th Int. Conf. Data Eng.*, Apr. 2012, pp. 390–401.
[113] Q. He, D. Jiang, Z. Liao, S. C. H. Hoi, K. Chang, E.-P. Lim, and H. Li, "Web Query recommendation via sequential query prediction," in *Proc. IEEE 25th Int. Conf. Data Eng.*, Mar. 2009, pp. 1443–1454.

WILLIAM GRANT HATCHER received the Bachelor of Science degree in materials science and engineering from the University of Maryland, in 2009, and the master's degree in computer science from Towson University, in 2018, where he is currently pursuing the Ph.D. degree. His research interests include mobile computing and security, big data, and machine learning.

CHENG QIAN received the B.S. degree from Jianqiao University, Shanghai, China, in 2018, and the M.S. degree in computer science from Towson University, in December 2020, where he is currently pursuing the Ph.D. degree. His research interests include the Internet of Things, cyberspace security, and computer networks.

WEICHAO GAO received the B.S. degree from Fudan University, China, in 2005, the M.B.A. degree from the University of Michigan, in 2011, and the M.S. degree in computer science and technology from Towson University, in 2017, where he is currently pursuing the Ph.D. degree. His research interests include the Internet of Things, cyberspace security, and computer networks.

FAN LIANG received the bachelor's degree in computer science from Northwestern Polytechnical University, China, in 2005, and the M.S. degree in computer engineering from the University of Massachusetts at Dartmouth, in 2015. He is currently pursuing the Ph.D. degree with Towson University. His research interests include the Internet of Things, data analytics, edge computing, and security.

KUN HUA received the B.Sc. (Hons.) and M.Sc. degrees in electrical and computer engineering from Xi'an Jiaotong University, China, in 1999 and 2004, respectively, and the Ph.D. degree in computer and electronic engineering from the University of Nebraska Lincoln, Nebraska, USA, in 2008. He is currently an Associate Professor with the Department of Electrical and Computer Engineering, Lawrence Technological University, Southfield, MI, USA. He continued his research with the University of Nebraska Lincoln as a Postdoctoral Researcher in 2009. His current research interests include wireless communication and signal processing. He was a recipient of the Best Paper Award of ACM ANSS 2011 and another of his paper is nominated as the Best Paper in IEEE BCGIN 2011. He has served as a Guest Editor for international journals, the Chair, and a Committee Member for several conferences.

WEI YU received the B.S. degree in electrical engineering from the Nanjing University of Technology, Nanjing, China, in 1992, the M.S. degree in electrical engineering from Tongji University, Shanghai, China, in 1995, and the Ph.D. degree in computer engineering from Texas A&M University, in 2008. He is currently a Full Professor with the Department of Computer and Information Sciences, Towson University, MD, USA. His research interests include cybersecurity and privacy, the cyber-physical systems/Internet of Things, and data and machine learning-driven applications. He was a recipient of the 2014 NSF CAREER Award, the 2015 University System of Maryland (USM) Regents' Faculty Award for Excellence in Scholarship, Research, or Creative Activity, and the 2016 USM Wilson H. Elkins Professorship. His research received the Best Paper Awards from ICC 2008, ICC 2013, IPCCC 2016, WASA 2017, IEEE CyberSciTech 2020, and DASC 2020.